Mining Microarray Gene Expression Data using Machine Learning Techniques

Doctor of Philosophy (Engineering)

Thesis Submitted by

Shilpi Bose

School of Education Technology
Faculty Council of Engineering &Technology
Jadavpur University
Kolkata-700032,India
2022

Mining Microarray Gene Expression Data Using Machine Learning Techniques

by

Shilpi Bose

Under the guidance of

Prof. Matangini Chattopadhyay

Professor

School of Education Technology

Jadavpur University

Kolkata, India

&

Dr. Chandra Das

Associate Professor

Department of Computer Sc. & Engg.

Netaji Subhash Engineering. College

Kolkata, India

School of Education Technology

Faculty Council of Engineering &Technology

Jadavpur University

Kolkata-700032,India

2022

List of Publications

Journal Papers:

1. **Shilpi Bose**, Chandra Das, Aishwarya Barik, Kuntal Ghosh, Matangini Chattopadhyay, Samiran Chattopadhyay, “HCFPC: A New Hybrid Clustering Framework using Partition-Based Clustering Algorithms to Group Functionally similar Genes from Microarray Gene Expression Data”, Springer Nature Computer Science, (communicated).

2. **Shilpi Bose**, Chandra Das, Abhik Banerjee, Kuntal Ghosh, Matangini Chattopadhyay, Samiran Chattopadhyay, Aishwarya Barik, “Multi-Filtering and Supervised Gene Clustering Algorithm based Ensemble Classification Model for Microarray Gene Expression Data”, PEERJ, 2021. Major Index: SCI, SCOPUS: https://doi.org/10.7717/peerj-cs.671

3. **Shilpi Bose,** Chandra Das**,** Kuntal Ghosh**,** Matangini Chattopadhyay, Samiran Chattopadhyay, “A Framework for Neighborhood Configuration to Improve the KNN based Imputation Algorithms on Microarray Gene Expression Data”, International Journal of Bioinformatics Research and Applications (IJBRA), Inderscience, 2021. Major Index: SCOPUS, SCImago: https://doi.org/10.1504/IJBRA.2022.124989

4. Chandra Das, **Shilpi Bose,** Samiran Chattopadhyay, Matangini Chattopadhyay, Alamgir Hossain, “A Bicluster-based Sequential Interpolation Imputation Method for Estimation of Missing Values in Microarray Gene Expression Data", Current Bioinformatics, Bentham Science, 12(2), pp. 118-130, 2017. Major Index: SCI Expanded: http://dx.doi.org/10.2174/1574893612666170106102019

5. Chandra Das**, Shilpi Bose**, Matangini Chattopadhyay, Samiran Chattopadhyay, “A Novel Distance Based Iterative Sequential *K*NN Algorithm for Estimation of Missing Values in Microarray Gene Expression Data", International Journal of Bioinformatics Research and Applications (IJBRA), Inderscience, 12(4), pp.-312-342, 2016**. Major Index: SCOPUS, SCImago: https://doi.org/10.1504/IJBRA.2016.080719**

6. Chandra Das, **Shilpi Bose**, Matangini Chattopadhyay, Samiran Chattopadhyay., “A Novel Distance based Modified K-Means Clustering Algorithm for Estimation of Missing Values in Microarray Gene Expression Data”, International Journal of Information Technology and Management Information System, Volume 5, Issue 3, pp. 1-13, September, 2014.

   https://d1wqtxts1xzle7.cloudfront.net/63226040/50320140503001202005O7-92013-1wsh1yt-libre.pdf?1588843441=&response-content-disposition=attachment%3B+filename%3DA_NOVEL_DISTANCE_BASED_MODIFIED_K_MEANS.pdf&Expires=1664354280&Signature=CKdBgzkmN0aiVAQt2k6tQdHG9exVDgbJOkgM2J0kfyKNLlmqIoaDTYKBIuBo53Eq3cfX93feAJMHyMcnWcaEhWA6SeIWAzMv2mS32j6AhUA8IDL4zHU5mTVwlai2ZIejAJ8VThCjxYlJGG5VXvssss61h7M9vMB9MCu7vNlsaxQz9MBVCRzZDcWOLnc6nd2ApvJGAkeoJ6XWtGGrXVOzStWPce7mqw1HtutFxowrNBYz75I8C~Fs7yGVC6~sQJ4SzzTuBwIpXF0aIz870ueqR99Hf12eXRs1upvG40Zju9t

o~XLO~mVnOEKlDk32t2LHvAvc7Z82ungOI9BkIIEEuw__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

Conference Papers:

7. Chandra Das, **Shilpi Bose**, Sourav Dutta, Kuntal Ghosh, Samiran Chattopadhyay," New Hybrid Gene Selection-Sample Classification Method in Microarray Data, Handbook of Research on Engineering Innovations and Technology Management in Organizations, IGI GLOBAL, April, 2020. DOI: 10.4018/978-1-7998-2772-6

   ISBN13: 9781799827726|ISBN10: 1799827720|EISBN13: 9781799827733 https://www.igi-global.com/book/handbook-research-engineering-innovations-technology/237842
8. **Shilpi Bose**, Chandra Das, Abhik Banerjee, Matangini Chattopadhyay, Samiran Chattopadhyay," An Ensemble Filtering and Supervised Clustering based Informative Gene Selection Algorithm in Microarray Gene Expression Data", 4th International Conference on Computational Intelligence and Networks (CINE-2020), IEEE, 27-29 February, 2020. https://doi.org/10.1109/CINE48825.2020.234391

9. Chandra Das, **Shilpi Bose**, Abhik Banerjee, Sourav Dutta, Kuntal Ghosh, Matangini Chattopadhyay, "Comparative Performance Analysis of Different Measures to Select Disease Related Informative Genes from Microarray Gene Expression Data", International Conference on Innovation in Modern Science and Technology (ICIMSAT-2019), Springer, September 20th-21st, 2019. Published as a book chapter in a book named Intelligent Techniques and Applications in Science and Technology, Proceedings of the First International Conference on Innovations in Modern Science and Technology. https://doi.org/10.1007/978-3-030-42363-6_105

10. Chandra Das, **Shilpi Bose**, Debanjana Karmakar, Agniswar Roy, Natasha Ghosh, Abhik Banerjee, Matangini Chattopadhyay, "Impact of Partition Based Clustering Algorithms to Cluster Samples in Microarray Gene Expression Data", International Conference on Innovation in Modern Science and Technology (ICIMSAT-2019), Springer, September 20th-21st, 2019. Published as a book chapter in a book named Intelligent Techniques and Applications in Science and Technology, Proceedings of the First International Conference on Innovations in Modern Science and Technology. https://doi.org/10.1007/978-3-030-42363-6_77

11. **Shilpi Bose**, Chandra Das, Samiran Chattopadhyay, Matangini Chattopadhyay, "A Novel Bicluster Based Missing Value Prediction Method for Microarray Gene Expression Data", IEEE, MAMI 2015 , December 17-19, 2015. https://doi.org/10.1109/MAMI.2015.7456603

12. **Shilpi Bose**, Chandra Das, Tamaghna Gangopadhyay, Samiran Chattopadhyay, "A Modified Local Least Square based Missing Value Estimation Method in Microarray Gene Expression Data", IEEE, ADCON 2013 , December 15-17, 2013. https://doi.org/10.1109/ADCONS.2013.11

13. **Shilpi Bose**, Chandra Das, Sourav Dutta , Samiran Chattopadhyay, “A Novel Interpolation based Missing Value Estimation Method to Predict Missing Values in Microarray Gene Expression Data”, IEEE, CODIS 2012 , December 28 -29, 2012. https://doi.org/10.1109/CODIS.2012.6422202

14. **Shilpi Bose**, Chandra Das, Abirlal Chakraborty, Samiran Chattopadhyay, "Effectiveness of Different Partition based Clustering Algorithms for Estimation of Missing Values in Microarray Gene Expression Data", Springer sponsored The Second International Conference on Advances in Computing and Information Technology (ACITY 2012), July 13-15, 2012. https://doi.org/10.1007/978-3-642-31552-7_5

List of Presentations in National and International Conferences and Workshops

1. **Shilpi Bose**, Chandra Das, Samiran Chattopadhyay, Matangini Chattopadhyay, “A Novel Bicluster Based Missing Value Prediction Method for Microarray Gene Expression Data”, IEEE, MAMI 2015 , December 17-19, 2015.

2. **Shilpi Bose**, Chandra Das, Tamaghna Gangopadhyay, Samiran Chattopadhyay, “A Modified Local Least Square based Missing Value Estimation Method in Microarray Gene Expression Data”, IEEE, ADCON 2013 , December 15-17, 2013.

3. **Shilpi Bose**, Chandra Das, Sourav Dutta , Samiran Chattopadhyay, “A Novel Interpolation based Missing Value Estimation Method to Predict Missing Values in Microarray Gene Expression Data”, IEEE, CODIS 2012 , December 28 -29, 2012.

4. **Shilpi Bose**, Chandra Das, Abirlal Chakraborty, Samiran Chattopadhyay, "Effectiveness of Different Partition based Clustering Algorithms for Estimation of Missing Values in Microarray Gene Expression Data", Springer sponsored The Second International Conference on Advances in Computing and Information Technology (ACITY 2012), July 13-15, 2012.

"Statement of Originality"

I Sri Shilpi Bose registered on 16.12.2016 do hereby declare that this thesis entitled "**Mining Microarray Gene Expression Data Using Machine Learning Techniques**." contains literature survey and original research work done by the undersigned candidate as part of Doctoral studies.

All information in this thesis have been obtained and presented in accordance with existing academic rules and ethical conduct. I declare that, as required by these rules and conduct, I have fully cited and referred all materials and results that are not original to this work.

I also declare that I have checked this thesis as per the "Policy on Anti Plagiarism, Jadavpur University, 2019", and the level of similarity as checked by iThenticate software is 8%.

Shilpi Bose.
Signature of Candidate:

Date : 13.10.2022

Certified by Supervisor(s):(Signature with date, seal)

1. Matangini Chattopadhyay 13/10/2022

Professor
School of Education Technology
Jadavpur University
Kolkata - 700 032

2. Chandra Das 13/10/2022

Associate Professor
Computer Sc. & Engineering
Netaji Subhash Engineering College
Technocity, Garia, Kolkata -700152
West Bengal, India

CERTIFICATE FROM THE SUPERVISORS

This is to certify that the thesis entitled " **Mining Microarray Gene Expression Data Using Machine Learning Techniques**" *submitted by Shri Shilpi Bose, who got his name registered on 16.12.2016 for the award of Ph. D. (Engg) degree of Jadavpur University is absolutely based upon his own work under the supervision of Prof. Matangini Chattopadhyay and Dr. Chandra Das and that neither his thesis nor any part of the thesis has been submitted for any degree/ diploma or any other academic award anywhere before.*

1. Matangini Chattopadhyay

(Prof. Matangini Chattopadhyay)
Professor
School of Education Technology
Jadavpur University
Kolkata-700032, India

Professor
School of Education Technology
Jadavpur University
Kolkata - 700 032

2 Chandra Das

(Dr. Chandra Das)
Associate Professor
Department of Computer Sc. & Engg.
Netaji Subhash Engineering College
Kolkata-700152, India

Associate Professor
Computer Sc. & Engineering
Netaji Subhash Engineering College
Technocity, Garia, Kolkata -700152
West Bengal, India

Acknowledgement

I would like to express my deepest gratitude to all the people who were actively involved in making me successfully complete this research work.

First and foremost, I wish to sincerely acknowledge the help and support of my supervisor, Prof. Matangini Chattopadhyay, School of Education Technology, Jadavpur University for her immense patience to carry on my research work.

A special thanks to Prof. Samiran Chattopadhyay, Department of Information Technology, Jadavpur University, who has encouraged me and supported me a lot during my research work both physically and financially.

My sincere acknowledgement also goes to the competent authority of Netaji Subhash Engineering College for allowing me to carry on my research work.

I also like to thank Dr. Kuntal Ghosh, Associate Professor, MIU, ISI, Kolkata, for his sincere help and support in some of my research work. He has also constantly inspired me to continue my research and provide active knowledge assistance whenever required.

Finally my unbounded thanks goes to Dr. Chandra Das, Associate Professor, Department of Computer Sc. & Engg., Netaji Subhash Engineering College, without whose constant inspiration and active involvement it is impossible for me to continue my research work. I am also thankful for her consent to co-supervise my research work.

[illegible]

Shilpi Bose

Contents

List of Figures:

List of Tables:

Abstract

The inter-networked society has been experiencing an explosion of biological data. However, the explosion is paradoxically acting as an impediment to acquiring knowledge. The meaningful interpretation of these large volumes of biological data is increasingly becoming difficult. This analysis is very much crucial to elucidate several secrets of life and several aspects of medical sciences. The most efficient method to investigate these data is via laboratory experiments, but it involves lots of time, money, and manpower. So, effective and efficient computational tools are needed to store, analyze, and interpret these diverse types of biological data. Data mining bridges this gap. Data mining techniques are of two types: data management techniques and data analysis techniques. Among the different data mining techniques, machine learning based data mining techniques are widely used to mine biological data.

Among the different types of bio-molecular data, gene expression data is highly impactful. Microarray techniques such as DNA and high density oligonucleotide chips are powerful biotechnologies as they are able to record the expression levels of thousands of genes simultaneously. Microarray data analysis has great impact in a number of studies over a broad range of biological and medical disciplines, including identification of functions of novel genes, identification of pathway in gene regulatory network, cancer classification by class discovery and prediction, identification of unknown effects of a specific therapy, identification of genes relevant to a certain diagnosis or therapy, and cancer prognosis etc. So, microarray gene expression data analysis plays an important role in real life applications.

Due to several shortcomings of microarray experiments, considerable missing values (MVs) are introduced in the resultant matrix. Such incomplete matrices pose a problem in analysis algorithms as they need complete matrices. It is not feasible to repeat microarray experiments as they are overwhelmingly costly. So, designing algorithms for predicting these missing values accurately have become a mandatory preprocessing step before analysis.

The challenge in this thesis, overall, is to devise powerful machine learning methodologies by symbiotically combining different tools to mine gene expression data in more efficient ways. In this regard, this thesis presents some new supervised and unsupervised learning methodologies, which are efficient in terms of prediction accuracy. The proposed methodologies are used to solve certain problems related to DNA microarray gene expression data.

In the first and second works of this thesis, clustering techniques are used to develop new imputation methods for prediction of missing values more accurately. In the first work, we have introduced a new robust framework which is embedded in the *K*-nearest neighbor imputation method (*K*NNimpute), as well as its different versions to achieve better neighborhood formation, in order to improve the prediction accuracies. Apart from this a new version of *K*NNimpute is proposed here. From the experimental results it has been found that in each and every case, the proposed modified method with new framework significantly outperform their corresponding traditional versions and are also comparable with the existing robust numerical methods. In the second work we have given focus on developing a new imputation method via combining clustering and numerical approach to improve prediction accuracy. Existing numerical methods are very complex and hard to implement. The proposed work is simple compared to existing numerical methods and shows its superiority.

In third and fourth work new supervised and unsupervised learning techniques are developed to analyze gene expression data in more efficient manner. Here, in the third work, we have designed a novel framework using different partition-based clustering algorithms (mainly different versions of *K*-Means) to provide an intuitive model for eliminating noise and also generating functional gene clusters. The model is also capable of clustering genes without using any predefined *K* as *K* is automatically detected here. The effectiveness of the algorithm, along with a comparison with other algorithms, is demonstrated on different microarray datasets.

In the fourth work, a new ensemble machine learning classification model named Multiple Filtering and Supervised Attribute Clustering algorithm based Ensemble Classification model (MFSAC-EC) is proposed to classify cancer samples more accurately which can handle class imbalance problem and high dimensionality of microarray datasets. The superiority of the algorithm, along with a comparison with other algorithms, is demonstrated on different microarray data sets.

Chapter 1

Introduction

Due to intense research activities and wide use of high-throughput technology in the area of biological sciences, society is experiencing an explosion of biological data. As the size of the biological databases increases day by day, analyzing this enormous volume of biological data has become complicated. This analysis is very much crucial to elucidate several secrets of life and several aspects of medical sciences. The most efficient method to investigate these data is via laboratory experiments. But it involves lots of time, money, and manpower. So, effective and efficient computational tools are needed to store, analyze, and interpret these different types of biological data. In this regard, a new field called bioinformatics [1, 2, 250] has risen to overcome the above-mentioned issues.

Bioinformatics [1, 2, 250] is the conceptualizing biology in terms of molecules and applying informatics techniques to understand and organize the information with these molecules, on a large scale. It entails the development of algorithms utilizing various methods, such as machine learning, data mining, pattern recognition, applied mathematics, statistics, informatics, and biochemistry to store, analyze and interpret this vast amount of biological data [3]. Gene sequence alignment and sequence analysis, finding informative genes , applying classification and clustering techniques for gene dimension reduction or feature selection from gene expression data, genome annotation, protein structure prediction and alignment of proteins, protein- protein docking and study the modeling of evolution are the significant fields of research initiatives in the field of bioinformatics. Therefore, bioinformatics can be treated as the field where computational tools and techniques are used to foster the biological discoveries [1, 2 ,3].

Machine learning [4, 5] is one sub field of artificial intelligence which involves the study of computer algorithms that let systems to automatically learn from experience and get better at doing so. The systems which are equipped with machine learning algorithms enable them to decide on their own without outside assistance. Such choices are made through identifying important underlying patterns in large, complex data sets. Machine learning techniques are divided into three broad categories: supervised learning, unsupervised learning, and reinforcement learning. These techniques are used in several ways in different fields like pattern recognition, image processing, data mining, natural language processing [4-12] etc.

Data mining [11, 12] is a branch of computer science where several hidden information are extracted from data using several mining techniques. Data mining techniques are widely used in medical data examination field for analyzing, obtaining, modifying, deciphering, and displaying medical records that are kept in repositories. Mining medical records is very important for improvement of medical therapy and in parallel it is very challenging also, because diagnosis and prediction of diseases are directly related to a patient's life or death situation. Patients' and their families' lives could end in catastrophe if a classification or forecast is made incorrectly. Data analysis techniques consist of two types of techniques: data management techniques and data analysis techniques. Among the several data analysis techniques, machine learning techniques are widely used in medical data mining field to make decisions to easily and quickly diagnose and predict diseases.

Among the several types of bio-technology, microarray technology is one of the most popular high-throughput bio-technology which is used to measure the expression level of a huge number of genes simultaneously in particular cells or tissues [13]. Results of microarray technology are large matrices known as gene expression data matrices where a row contains information about a gene; a sample/experiment is represented by a column and a cell contains information about a gene for a specific sample/experiment. The use of Microarray data analysis has been applied in several studies covering a wide variety of biological and medical disciplines, including identification of functions of novel genes, identification of pathway in gene regulatory network, identification of unidentified side effects of a particular medicine, the discovery and prediction of cancer categorization by class, the identification of genes linked to a certain diagnostic or therapy, and cancer prognosis etc. [14-30]. So, microarray gene expression data analysis has important aspects in real life applications.

Due to several shortcomings of microarray experiments [31], considerable missing values (MVs) are introduced in the resultant matrix [31]. Sometimes, a large number of genes (up to 90%) are affected and contain missing values [31]. Such incomplete matrices pose a problem in analysis algorithms [14-30] as they need complete matrices. It is not feasible to repeat microarray experiments as they are overwhelmingly costly. So, designing algorithms for predicting these missing values accurately have become very important. Accuracy of these prediction methods can affect the results of analysis algorithms as these methods require complete gene matrices. So, missing value prediction in gene expression data is a mandatory preprocessing task before analysis.

The challenge is, therefore, to devise powerful machine learning methodology-based data mining techniques to preprocess and analyze gene expression data in more effective manner. The focus should be given to develop capable information processing systems which can deal with real life ambiguous problems and achieve tractable low-cost solutions. In the above background, the focus of the research undertaken in this thesis is presented next.

1.1 Aim of the Dissertation

The major focus of this research work is to devise machine learning methodology based new data mining techniques to preprocess and analyze gene expression data, which are efficient in terms of prediction accuracy.

In view of the several technical problems associated with microarray experiments, a considerable number of entries are found missing in a typical microarray gene expression dataset. As a consequence, due to the unavailability of complete data, the effectiveness of the downstream analysis algorithms deteriorates. Different imputation techniques are employed to address this problem. These techniques are developed based on two approaches. One approach is weighted average based methods and second one is numerical approach-based methods. Among these techniques, the numerical methods are more robust than the weighted average based methods. However, weighted average based methods are widely used in several applications as these methods generate consistent results and are algorithmically simple, but these methods also suffer from some drawbacks that are seldom elaborated upon. These deficiencies have been pointed out in our first work. To solve these problems, in the first work we have first proposed a primary framework via proposing a new hybrid distance based a new version of the *K*-nearest neighbor imputation method (*K*NNimpute) named iterative sequential *K*-nearest neighbor imputation method (IS*K*NNimpute) approach. This work has not the capability to overcome all those deficiencies. So, we have introduced a new robust framework which is embedded in the *K*-nearest neighbor imputation method (*K*NNimpute), as well as its different versions. The idea is to achieve better neighborhood formation, in order to improve the prediction accuracies. From the experimental results it has been found that in each and every case, the proposed modified methods with new framework significantly outperform their corresponding traditional versions and are also comparable with the existing robust numerical methods.

In the second work, we have given focus on developing a new imputation method via combining clustering and numerical approach to improve prediction accuracy. It has been already found that prediction accuracy of numerical methods is high but these methods sometime generate inconsistent results. Existing numerical methods are very complex and hard to implement. Due to unavailability of codes in the internet these methods are not used frequently in the applications. Considering this view, we have proposed a new imputation method which is a combination of bi-clustering and interpolation based numerical work. The proposed work is simple compared to existing numerical methods and shows its superiority.

One important analysis task of microarray gene expression data is to find hidden patterns among genes present in this data to extract relevant information which will be beneficial for functional genomics. It has been already found that genes with similar expression patterns (co-expressed genes) may have similar biological functions. The information from these hidden patterns among genes may help in analysing functional enrichment of genes, understanding gene function of uncharacterized genes, understanding cellular processes, gene co-regulation or relation in functional pathways, and finding out information related to transcriptional regulatory networks. So, it is a great challenge to identify groups of genes based on similar patterns from this large voluminous gene expression data. Clustering techniques [10, 14, 15] are widely used in gene expression data for clustering genes to partition genes among relevant functional groups. A huge number of different clustering techniques are already developed to solve this problem. However, the clustering of genes is an old problem but as gene expression data is very much noisy so proper noise deletion is an important task before clustering and is still challenging. Although tight clustering methods are developed, these methods have several computational limitations. Among the different category-based clustering methods, partition-based clustering methods are most popular but these methods are unable to eliminate noise. Here, in the third work, we have designed a novel framework using different partition-based clustering algorithms (mainly different versions of *K*-Means) to provide an intuitive model for eliminating noise and also generating functional gene clusters. The model is also capable of clustering genes without using any predefined *K* as *K* is automatically detected here.

Sample classification is one of the important downstream analysis based application of microarray gene expression data. The gene expression data matrix contains a huge number of genes compared to a limited number of samples and this is a most important problem for sample classification. Most classification algorithms suffer from such a high-dimensional input space. Also, a very small number of genes contain relevant information for sample classification. Secondly, the class imbalance problem is an overhead for sample classification also. In this regard, in the fourth work, a new ensemble machine learning classification model named Multiple Filtering and Supervised Attribute Clustering algorithm based Ensemble Classification model (MFSAC-EC) is proposed which can handle class imbalance problem and high dimensionality of microarray datasets. The MFSAC method is a supervised feature selection technique combining multiple filters with a new supervised attribute clustering algorithm. Then for every sub dataset, a base classifier is constructed separately, and finally, the predictive accuracy of these base classifiers is combined using the majority voting technique forming the MFSAC-based ensemble classifier.

1.2 Organization of the Dissertation:

Prior to dealing with the proposed works on gene expression data, a comprehensive survey of the relevant research publications is reported in Chapter 2. The survey covers following aspects:

1. A general overview of the existing machine learning techniques for mining data.
2. A brief overview of bioinformatics.
3. A brief overview of different analysis tasks related to microarray gene expression data

In the third chapter, we have given our focus on modification of the existing most popular weighted-average based imputation technique and its several versions. In this regard, we have found out the drawbacks of the most popular weighted-average based imputation technique and its several versions. To solve these problems, we first proposed a new hybrid distance based a new version of the *K*-nearest neighbor imputation method (*K*NNimpute) named iterative sequential *K*-nearest neighbor imputation method (IS*K*NNimpute). Although IS*K*NNimpute outperformed several existing imputation techniques, it failed to overcome all those deficiencies found in weighted-average based imputation approaches. So, we have introduced a new framework to apply in weighted-average based imputation procedure for neighborhood formation. The proposed framework is embedded in the *K*-nearest neighbor imputation method (*K*NNimpute), as well as its different versions. The idea is to achieve better neighborhood formation, in order to improve the prediction accuracies. It is based on a hybrid distance and gene transformation procedure which utilizes simultaneously the advantages of Euclidean distance, Mean squared residue score, and Pearson correlation coefficient to select the best possible neighbors, using pattern-based similarity. The new framework is tested on ten well-known microarray datasets. From the experimental results it has been found that in each and every case, the proposed modified methods significantly outperform their corresponding traditional versions and are also comparable with the existing robust numerical methods. Among all the versions, IS*K*NNimpute with the new framework is better than other *K*NN versions.

In the fourth chapter, we have proposed a new imputation technique which is a combination of bi-clustering and numerical work. In our method, we have used bi-clustering technique for neighborhood formation and then applied interpolation technique for predicting missing values. The proposed method is tested on different microarray datasets and from the experimental results it has been found that in each and every case, the proposed modified methods significantly outperform existing robust numerical methods.

Here, in the fifth chapter, we have designed a novel framework using different partition-based clustering algorithms (mainly different versions of *K*-Means) to provide an intuitive model for eliminating outliers and also generating functional gene clusters. The model is also capable of clustering genes without using any predefined *K* as *K* is automatically detected here. The effectiveness of the algorithm, along with a comparison with other algorithms, is demonstrated on different microarray datasets.

In the sixth chapter, a new ensemble machine learning classification model named Multiple Filtering and Supervised Attribute Clustering algorithm-based Ensemble Classification model (MFSAC-EC) is proposed which can handle class imbalance problem and high dimensionality of microarray datasets. This model first generates a number of bootstrapped datasets from the original training data where the oversampling procedure is applied to handle the class imbalance problem. The proposed MFSAC method is then applied to each of these bootstrapped datasets to generate sub-datasets, each of which contains a subset of the most relevant/informative attributes of the original dataset. The MFSAC method is a supervised feature selection technique combining multiple filters with a new supervised attribute clustering algorithm. Then for every sub dataset, a base classifier is constructed separately, and finally, the predictive accuracy of these base classifiers is combined using the majority voting technique forming the MFSAC-based ensemble classifier. Also, a number of most informative attributes are selected as important features based on their frequency of occurrence in these sub-datasets. The effectiveness of the algorithm, along with a comparison with other algorithms, is demonstrated on different microarray data sets.

The last chapter, that is, Chapter 7, draws the conclusion along with the future research.

Chapter 2

Literature Survey

This dissertation reports the application of different machine learning techniques to mine useful biological information from gene expression data. In this background, basic concepts of data mining techniques, machine learning techniques, bioinformatics and gene expression data are discussed in this chapter. The first section covers the basic overview of machine learning techniques. In the second section the fundamentals of data mining techniques are discussed. In the third section fundamentals of bioinformatics is given while in the last section gene expression data is elaborated.

2.1 A Brief Overview on Machine Learning Techniques

Machine learning [4, 5, 6] is a sub field of artificial intelligence. It is the field research of computer techniques that provide computers the capacity to automatically learn from experience. The systems which are equipped with machine learning algorithms, enable them to decide on their own without outside assistance. Such choices are made through identifying important underlying patterns in large, complex data sets. Some of the machines learning approaches are decision tree, genetic algorithm, neural network, inductive logic procedures, etc. These are currently under active investigation. With a hierarchical structure of attributes and classes, the ultimate goal is to handle any general sorts of data, including situations where the amount and type of characteristics may fluctuate and where new learning layers are superimposed In order to offer understanding of the decision-making process, it must sufficiently resemble human reasoning. Although background information may be used in machine learning development, human intervention is not expected during operation.

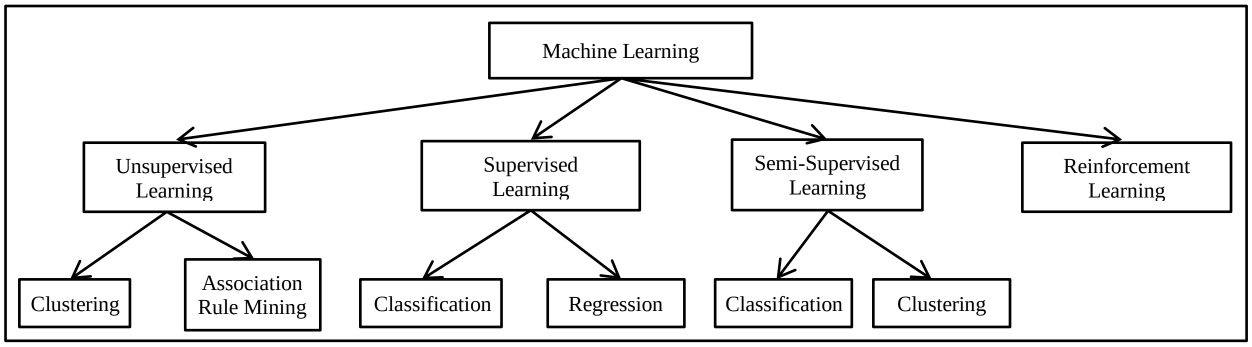


Figure 2.1 Taxonomy of Machine Learning

The different application domains of machine learning are Pattern recognition, Computer vision, prediction, data mining, natural language processing and information retrieval.

Depending on the method of learning, the kind of data they input and output, and the kind of issue they resolve, machine learning algorithms are divided into four broad categories as shown in Figure 2.1. These are supervised, unsupervised, semi-supervised and reinforcement learning. Apart from these categories, there are a few approaches.

2.1.1 Supervised Learning

Supervised learning [4, 5, 6] is one kind of machine learning technique which is to learn a function from sample input-output data pairs that transfers an input to an output. When the data takes the form of input variables and output target values, supervised learning is used. The algorithm picks up the function for mapping input to output to infer a function; it makes use of labeled training data and a variety of training samples. After inferring the function, it applies it on test data, and predicts output target value of test samples.
Supervised learning algorithms are divided into two categories: 1) Classification (2) Regression.

In classification task, the target values are discrete while in regression the target values are continuous. As for example predicting the class label or sentiment of a text message in Twitter or Facebook is an example of text classification while predicting the loan amount for a customer in a bank is an example of regression.

2.1.2 Unsupervised Learning

It can be very expensive, difficult, or even impossible to accurately label a training sample with its real category in many pattern recognition applications. The land-use classification in remote sensing can be treated as an example. Unsupervised learning [4, 5, 6] is applied when there is no class label is associated with input data that means data is available only in the form of inputs these algorithms simulate the underlying patterns in the data to gain a better understanding of its properties. It is frequently used for generative feature extraction, trend and structure identification, result grouping, and exploratory purposes.

One of the most popular unsupervised learning methods is data clustering which is a catch-all term for a number of techniques used to identify inherent natural groupings, or clusters, in multidimensional data based on similarities between the patterns and then these are used to forecast results from unknown inputs. Predicting consumer purchase behaviour would be an illustration of this strategy.

There are numerous other unsupervised learning challenges than clustering. These include identifying association rules, dimensionality reduction, feature learning, and density estimation.

2.1.3 Semi-supervised learning

This is another kind of machine learning technique. This technique is a hybridization of supervised and unsupervised machine learning techniques, as it operates on both labelled and unlabelled data. Thus, it falls between learning "without supervision" and learning "with supervision". In the real world, in different contexts, where the number of unlabelled data is greater than the number of labelled data, semi-supervised learning is very useful.

A semi-supervised [4, 6] learning model's main objective is to create predictions that are superior to those made using the model's labelled or unlabelled data alone. Semi-supervised learning is utilised in a variety of applications, including text categorization, fraud detection, machine translation, and fraud detection.

2.1.4 Reinforcement learning

Another type of machine learning technique is reinforcement learning [4,6], which enables software agents and computers to automatically assess their best behaviour in a given situation or environment for improving its performance based on achievement of penalty and reward. It is basically an environment driven approach and its ultimate goal is to choose the best action from environmental activities to increase reward or minimize the task.

It is an effective tool in a variety of AI-related domains, including robotics, autonomous driving, manufacturing, and supply chain logistics, although it is not recommended for usage in simple or fundamental situations.

Apart from these four major learning techniques there are a few other derived learning techniques. The most popular ones are discussed below.

2.1.5 Multitask Learning

The straightforward aim of multitask learning [6] is to improve the performance of other students. When multitask learning algorithms are used on a task, they recall the steps taken to solve the issue or arrive at the desired result. The algorithm then applies these processes to solve other problems or tasks that are comparable to the one at hand. This process for transferring information from one algorithm to another is known as inductive transfer. Students can learn concurrently rather than individually and considerably more quickly if they share their experiences with one another.

2.1.6 Ensemble Learning

Ensemble learning [6] is the term for the type of learning that occurs when multiple individual learners are brought together to create only one learner. The particular learner could be a neural network, decision tree, or Naive Bayes, for example. Since the 1990s, ensemble learning has been a popular issue. It has been noted that a group of learners does a certain task more effectively than an individual student almost always.

2.1.7 Active Learning

The active learning algorithm [4] chooses the subset of data samples from which it wishes to learn in advance. From a big collection of unlabeled samples, the samples are chosen, and then they are labeled. As a result, the algorithm can outperform more established techniques while using far less training data that has been labeled. These techniques are quite helpful in situations where there may be a lot of unlabeled data yet labels are hard to come by or are expensive.

2.1.8 Online Learning

Utilizing data that becomes available in a sequential manner allows for training during online learning [3]. This method differs from batch sampling-based learning, which always has access to all of the training data. It can be helpful in situations where algorithms must dynamically adjust to new data patterns from all incoming input.

2.1.9 Incremental Learning

The incremental learning [4] approach is extremely similar to (and occasionally identical to) online learning. The primary distinction is that a training sample from an incoming data stream is only used once in online learning. Samples are typically chosen from a finite dataset for incremental learning, and the same samples may be processed more than once.

2.1.10 Meta Learning

In a meta-learning [4] paradigm, the machine learning model accumulates knowledge over a number of learning episodes, which frequently include a range of related tasks, and then makes use of this knowledge to enhance any subsequent learning performance. The objective is to solve novel tasks with just a few training samples. Meta-learning, also known as learning the learning process, seeks to enhance the learning algorithm given the knowledge of numerous learning episodes, in contrast to standard machine learning approaches where a particular task is learnt from scratch using a preset learning algorithm. Few-shot learning and metric learning are two examples.

2.1.11 Deep Learning

Deep learning [4] is a method for using multiple-layer neural networks to a variety of machine learning methods. For the purpose of comprehending the incoming data, these various processing layers learn representations of the data at various levels of abstraction.

2.2 Data Mining

The field of computer science known as data mining explores several computational methods for locating and extracting implicit knowledge that is valuable from massive databases [11,12]. Since the past two decades, this field's significance has grown steadily, and today it is crucial to the process of finding hidden knowledge in databases (KDD). Numerous sectors, including e-commerce, fraud detection, instruction detection, lie detection, customer relationship management, market basket analysis, telecommunication networks, the banking industry, inventory control, and bioinformatics, among others, have used data mining techniques [11, 12].

Disciplines including machine learning, artificial intelligence, probability, and statistics serve as the foundation for data mining. Predictive and descriptive tasks are the two main categories of data mining tasks. In the task of prediction, supervised learning functions are employed to forecast unknown or potential values of other relevant variables. The unsupervised learning functions are frequently used in the descriptive task to uncover patterns that describe the data and that people can understand. Data mining involves several techniques. These techniques are divided into two categories: (1) data management techniques like data pre-processing techniques, dimension reduction techniques etc. (2) machine learning techniques for data analysis tasks like Clustering, Classification, Regression, Association rule mining, and other techniques for anomaly/outlier detection and summarization.

2.2.1 Data Preprocessing

Due to the explosion of the huge amount of raw data, originated from multiple heterogeneous sources under diverse environmental and physical conditions, these data are prone to several errors. The errors in data include the presence of noise, missing elements, and inconsistent data that will produce low-quality data which in turn produce erroneous or low-quality data mining results. To remove these errors and improve the quality of data it is essential for these data to be preprocessed (to apply data preprocessing techniques). The methods for data preprocessing [11, 12] are categorized into data cleaning, data integration, data reduction, and transformation. To eliminate noise and fix discrepancies in the data, data cleaning can be used. By aggregating, removing redundancies, applying clustering, etc., data reduction reduces the size of the data. Data fusion creates a coherent data repository by combining data from several sources. In data transformations, the data are transformed into forms appropriate for mining.

These categories of preprocessing techniques may be working together to produce cleaned high-quality data useful for mining. Among different other preprocessing methods some important techniques are discussed in the next sections.

2.2.1.1 Missing Value Estimation

Missing Value Estimation [11, 12] is an important preprocessing step falling under the category of data cleaning. Due to the imperfect data acquisition technique or multiple errors in the data preparation, many entries in the produced data remain absent or missing. These missing entries create severe problems during the analysis of these data as many analysis algorithms require complete data. Ignoring those missing entries leads to the cancelation of the related important information producing imperfect analysis results. For example, consider the produced data is in the form of a matrix where some entries are found missing. Ignoring those entries results in ignoring either the entire corresponding rows or columns in which missing entries are present. This will produce low-quality analysis results as some important features may be ignored during the cancelation of those entire rows and columns. Hence, proper methodologies are essential to predict those missing entries to achieve better analysis results. There are multiple categories of such missing value estimation techniques exist in the literature which we will discuss in detail in chapter 3.

2.2.1.2 Dimension Reduction

A minimally sized subset of qualities (features) that are pertinent to the target concept should be found using dimension reduction [11, 12]. A better knowledge of the underlying process that produced the data is one of three goals of dimension reduction [11, 12]. The other two goals are to speed up and reduce the cost of prediction. There are two main methods for dimension reduction: one is feature selection and other is feature extraction. Some of the commonly used methods for feature extraction and feature selection are discussed below.

Feature Extraction

In the original feature space of dimension d (m d), feature extraction algorithms choose a suitable subset of dimension m (either linearly or nonlinearly). For feature extraction and dimensionality reduction, linear transforms including principal component analysis (PCA), independent component analysis (ICA), linear discriminant analysis (LDA), and projection pursuit (PP) have been extensively used. Techniques for nonlinear feature extraction can be defined in a variety of ways. The kernel PCA is one such approach that has a close connection to PCA [32]. Multidimensional scaling is another nonlinear feature extraction technique [32]. Neural networks are also used for feature extraction [32].

Feature Selection

Given a set of d features, choose the subset of size m that results in the minimum classification error, according to the definition of the feature selection problem given in [32, 33].

Filter, wrapper, and embedding relationships are the three basic types that can exist between a feature selection algorithm and the inducer or classifier used to assess the utility of the feature selection process.

1. **Filter Scheme:** The feature selection process can be thought of as a filter of unhelpful features prior to induction if it occurs before the induction or c stage. Filter methods assess the quality of the suggested feature subset based only on the inherent properties of the data and based on the link between each individual feature and the class label by computing straightforward statistics derived from the empirical distribution. The most common way is to rank the features in terms of the values of an univariate scoring metric. Then, the m features with the highest score are chosen to build the classifier. There is a large variety of different measures such as probabilistic or Distance metrics, measures inspired by the information theory. In a general sense, it can be seen as a particular case of the embedded scheme in which feature selection is used as a pre-processing. The filter schemes are independent of the induction algorithm.
2. **Wrapper Scheme:** In the wrapper approach, a search is conducted in the space of features, evaluating the goodness of each found feature subset by the estimation of the accuracy percentage of the specific classifier to be used, training the classifier only with the found features. In this scheme the relationship is taken the other way around: it is the feature selection algorithm that uses the learning algorithm as a subroutine. The general argument in favor of this scheme is to equal the bias of both the feature selection algorithm and the learning algorithm that will be used later on to assess the goodness of the solution. The main disadvantage is the computational burden that comes from calling the induction algorithm to evaluate each subset of considered features.
3. **Embedded Scheme:** The inducer or classifier in this approach has a custom feature selection method (either explicit or implicit). An illustration of this embedding can be found in the procedures for causing logical conjunctions. Traditional machine learning tools like decision trees and random forest are included in this scheme.

2.2.1.3 Data Discretization

Data discretization is the procedure by which a wide range of continuous values given for an attribute can be reduced into some discrete small intervals. Discretization techniques divide the continuous range into small intervals thereby giving a label for each of the values falling under that interval. Discretization lowers and streamlines the initial data in order to create a clear, user-friendly knowledge-level representation of the mining findings.

Discretization methods can be grouped as either top-down or bottom-up depending on the direction on which discretization will proceed. On the other hand, it can also be categorized as supervised or unsupervised depending on whether the class information is used for discretization or not.

2.2.2 Data Analysis Techniques

There is several data analysis technique under data mining. These are classification, regression, clustering, association rule mining, anomaly detection, summarization, sequential pattern techniques.

2.2.2.1 Classification

Classification is among the classical data mining technique that is established as a task under supervised machine learning. It finds mutual properties amongst a set of objects in a dataset and categorizes them into diverse classes in accordance with the classification model. Its primary goal is to carefully examine the training data and create an accurate description or model for each class using data features. This method uses mathematical techniques like decision trees, Neural networks and statistics.

2.2.2.2 Regression

It is one among data mining techniques that defines the association between dependent and independent variables. It is a supervised machine learning task. Prediction is accomplished with regressions support. Regression analysis is a mathematical concept that establishes a relationship between the values of the dependent variable and those of the independent or predictor variables. The predicted variable in regression may be a continuous variable. Regression involves mapping real valued prediction variables from learning function elements. Statistical regression, Neural Network, and Support Vector Machine regression are some of the commonly used regression strategies. More complex techniques such as Logistic regression, Decision Trees or Neural Networks could also be utilized for forecasting future values, these techniques could also be combined for attainment of better result.

2.2.2.3 Clustering

It is a data mining technique which classifies tangible or intangible objects into groups of related items. Clustering divides a set of data (records, tuples, objects, and samples) into several groups (clusters) based on prior similarities. The principal aim of clustering is finding groups (clusters) of objects based on affinity so that within individual cluster there is great resemblance to each other while clusters are diverse enough from one another. In machine learning terminology, clustering is a form of unsupervised learning.

2.2.2.4 Dependency Modeling (Association Rule Mining)

It’s amongst the finest acknowledged data mining techniques and is categorized under unsupervised data mining technique, which aims at finding connections or relations between items or records belonging to a large dataset and labels significant dependencies among variables. Association rule mining is implication of the form $X \rightarrow Y$, where X and Y are distinct items or item sets manufacturing if-then statements regarding attribute values. In market basket analysis this rule has been commonly used, it tries to analyze customers purchasing certain items and provides insight into the combinations customer frequently purchases together.

2.2.2.5 Anomaly detection

Synonymous to its name, it deals with the unearthing of most substantial changes or aberrations from the standard behavior.

2.2.2.6 Summarization

Though not amongst the techniques of data mining, but is a resultant of these methods, which is also known as generalization or description, which focuses on finding a concise depiction for a subset of facts.

2.2.2.7 Discovery of Sequential Patterns

Sequence discovery is a data mining technique that is used to determine sequential patterns or associations or regular events/trends between variable data fields over a business period.

2.3 Brief Overview of different Classification and Clustering Techniques

In this thesis we mainly developed different classification and clustering based machine learning models to mine gene expression data. So, a brief overview of different well-known classification is given first in this subsection and then different category based clustering algorithms are discussed in next subsection.

2.3.1 Different Classification Techniques

2.3.1.1 Bayesian Classifier

A fundamental idea in the creation of pattern classifiers is the probabilistic approach. Bayesian classifiers [34] and Naive Bayesian classifiers [34] belong to this group. Bayesian classifiers are statistical classifiers and are based on Bayes' theorem.

2.3.1.2 Decision Tree

A special type of classifier is the decision tree [35]. The process of learning decision trees from class-labeled training tuples is known as decision tree induction. In a decision tree, each internal node (non-leaf node) symbolizes a test on an attribute, each branch shows the test's result, and each leaf node, or terminal node, stores a class label. The attributes values of a given tuple X are checked against the decision tree when the corresponding class label is unknown. From the root to a leaf node, which contains the class prediction for that tuple, a path is drawn. Because they can be built without any subject expertise or parameter setup, decision tree classifiers are ideal for exploratory knowledge discovery. This is why they are so popular. High dimensional data can be handled via decision trees.

2.3.1.3 Neural Network Based Classifiers

The field of neural networks has attached attention of researchers from diverse fields. It covers a wide range of topics, such as comprehending and mimicking the human brain, as well as replicating more general human talents like voice and language use as well as the business, scientific, and technical disciplines of pattern recognition. A broad class of techniques can come under this heading. The basic structure of a neural network is layers of connected nodes, where each node produces a non-linear function of its input. A node's input can come directly from the input data or from other nodes. Some nodes can also be recognized by the network's output. As a result, the entire network reflects an extremely complex set of interdependencies that may include any level of nonlinearity, enabling the development of models for any generic functions. In the most basic networks, messages are propagated via layers of interconnected nodes by feeding the output from one node into another. Networks that connect the final output nodes to earlier nodes can model more complex behavior by giving the system the traits of a highly nonlinear system with feedback. The most commonly used family of neural networks for pattern recognition or classification tasks are multi-layer perceptron, radial basis function network, support vector machines [36-38], and so forth.

2.3.1.4 K-Nearest Neighbor Classifier

The k-nearest-neighbor method [34, 35] was first described in the early1950s. It has been extensively applied to pattern recognition. Based on learning by analogy, nearest neighbor classifiers compare a given test tuple with training tuples that are similar to it. N attributes are used to describe the training tuples. A point in an n-dimensional space is represented by each tuple. This results in the storage of all training tuples in an n-dimensional pattern space. A k-nearest neighbor classifier looks for the k training tuples that are most similar to an unknown tuple when given an unknown tuple. These k training tuples are the unknown tuple's k nearest neighbors. A distance metric, like Euclidean distance, is used to define proximity.

2.3.2 A Brief Overview of Different Clustering Techniques

Many clustering algorithms exist in the literature. In general, the major clustering methods can be classified into the following categories.

2.3.2.1 Partitioning Methods

A partitioning method creates k partitions of the data from a database containing n objects or data tuples, where each partition represents a cluster and k< n. In other words, it divides the data into k groups that collectively meet the requirements listed below: At least one object must be present in each group, and each object must be a part of exactly one group. The typical standard for a successful partitioning is that objects belonging to the same cluster are near or connected to one another, whilst those belonging to different clusters are separated or quite dissimilar. For evaluating the quality of partitions, there are numerous more types of criteria. Examples of partitioning-based clustering algorithms are the k-means algorithm, k-modes method, k-medoids (for example, PAM, CLARA, CLARANS [12, 39]) algorithm.

2.3.2.2 Hierarchical Methods

A hierarchical approach divides the supplied group of data objects into hierarchical decompositions. Depending on how the hierarchical breakdown is created, a hierarchical technique can be categorized as either agglomerative or divisive. The bottom-up technique, also known as the agglomerative approach, begins with each object creating a separate group (for instance, AGNES [12, 39]). Until all of the groups are merged into one (the top level of the hierarchy), or until a termination condition is met, it sequentially merges the items or groups that are close to one another. The top-down method, also known as the divisive method, begins with all of the items in the same cluster (for instance, DIANA [12, 39]). A cluster is divided into smaller clusters in each succeeding iteration, until eventually each item is one cluster, or until a termination condition is satisfied. Examples of hierarchical clustering algorithms are Chameleon, ROCK,CURE, BIRCH [12, 39] etc.

2.3.2.3 Density-Based Methods

The majority of object clustering techniques use object distance to group things together. Such algorithms struggle to find clusters of any shape and can only locate spherical-shaped clusters. On the basis of the idea of density, other clustering techniques have been created. Their basic premise is to grow a given cluster as long as the density (number of objects or data points) in the vicinity exceeds a predetermined threshold; specifically, for each data point inside a given cluster, the neighborhood within a predetermined radius must contain at least a predetermined number of points. A technique like this can be used to remove noise (outliers) and find clusters of any shape. The standard density-based approaches DBSCAN [12, 39] and its extension OPTICS [12, 39] grow clusters in accordance with a density-based connectivity analysis. A technique called DENCLUE [12, 39] groups items based on an examination of the density function value distributions.

2.3.2.4 Grid-Based Methods

Grid-based approaches divide the object space into a fixed number of grid-like cells. On the grid structure, all clustering actions are carried out (i.e., on the quantized space). The key benefit of this strategy is its quick processing time, which is usually unaffected by the quantity of data items and only depends on the number of cells in each dimension of the quantized space. A prominent example of a grid-based approach is STRING [12, 39]. Grid-based and density-based Wave Cluster uses wavelet transformation for clustering analysis [12, 39].

2.3.2.5 Model-Based Methods

Model-based methods posit a model for every cluster and then determine which model best fits the data. By creating a density function that reflects the spatial distribution of the data points, a model-based method may detect clusters. It also results in a method for automatically calculating the number of clusters based on common data, accounting for noise or outliers and producing reliable clustering techniques. Based on statistical modeling, the method EM [12, 39] performs expectation-maximization analysis. A probabilistic learning method called COBWEB [12, 39] uses concepts as a model for clusters and does probability analysis. A neural network-based clustering approach called SOM (or self-organizing feature map) [40] converts high-dimensional data into a 2-D or 3-D feature map that is also helpful for data visualization.

The clustering technique that is used depends on the type of data that is available as well as the application's specific goals. It is feasible to test several algorithms on the same data when cluster analysis is used as a descriptive or exploratory tool to discover what the data may reveal. There are more forms of clustering algorithms that need specific consideration in addition to the categories of clustering methods mentioned above. One is graph based clustering (CLIQUE [12, 39] and PROCLUS [12, 39]) and the other is constraint-based clustering [12, 39, 41].

As in this thesis gene expression data is mined and it is molecular biology related data so in the next section, fundamentals of bioinformatics is discussed as this field deals with molecular biology related data.

2.4 Bioinformatics

Over the last few decades, there has been an explosion in the amount of biological information generated by the scientific community due to major advances in the field of molecular biology, coupled with advances in genomics technologies [1, 2, 3, 250]. Due to the overwhelming amount of biological data, computerized databases to store, organize, and index the data, as well as specialized tools to view and interpret the data, are now absolutely necessary. Computers are becoming crucial for conducting biological research. Such a strategy is perfect due to how easily computers can manage massive amounts of data and examine the complicated dynamics seen in nature. So, a new field has been evolved, that is, bioinformatics. So, a new field has been evolved, that is, bioinformatics.

The use of computational techniques to biological discovery might be seen as bioinformatics [3]. To assess biological sequence data, genomic content and layout, and to forecast the function and structure of macromolecules, an interdisciplinary field integrating biology, computer science, mathematics, and statistics is used. This field's ultimate objective is to facilitate the discovery of novel biological insights and to provide a global viewpoint from which unifying biological principles can be deduced.

2.4.1 Aims of Bioinformatics

Bioinformatics has three main objectives.

1. First, bioinformatics organizes data in a way that makes it possible for researchers to access already-existing material and to submit new items as they are created, such as the Protein Data Bank for 3D macromolecular structures. So, in order to understand and organize the data related to these molecules on a large scale, bioinformatics conceptualizes biology in terms of molecules (in the sense of physical chemistry). It then applies informatics techniques (derived from disciplines such as applied mathematics, computer science, and statistics). Bioinformatics, which has a wide range of real-world uses, is essentially a management information system for molecular biology.
2. The creation of materials and tools to assist in data analysis is the second goal. For instance, it is interesting to compare a protein's sequence to ones that have been previously defined. It takes more than simply a text-based search and tools like FASTA [42] and PSI-BLAST [43] to get relevant results in this case. The creation of such resources necessitates proficiency in computational theory as well as a solid grasp of biology.
3. The final objective is to make use of these technologies to conduct data analysis and provide biologically relevant interpretation of the findings.

2.4.2 Information of different data used in Bioinformatics

Sources of data used in bioinformatics, the quantity of each type of data and bioinformatics subject areas that utilise this data [1-3] shown Table 2.1.

2.4.3 Basic Concepts of Molecular Biology

In this section the basic molecular biology related concepts are given which are required to understand the various problems of bioinformatics.

Cell

The simplest form of life is a cell [44]. They are the smallest unit capable of carrying out all of life's tasks in the current era. All living things are either made up of a single cell or are multicellular entities made up of many cells cooperating. It was discovered by Robert Hooke in 1665. Its theory was first came to known to worldwide in 1839 by Matthias Jakob Schleiden and Theodor Schwann who asserts that every organism is made up of one or more cells, and that every cell has the genetic information needed to control its operations and pass on knowledge to its progeny.

Table 2.1: Different biomolecular data

Type of Data	Volume of Data (Size)	Associated fields in Bioinformatics
Raw DNA sequence	8.2 million sequences (9.5 billion bases)	• Identifying introns and exons, • Separating coding from non-coding sections, • Predicting gene product, and • Doing forensic analysis
Protein sequence	300,000 sequences (~300 amino acids each)	• Algorithms for comparing sequences • Algorithms for multiple sequence alignments • Finding preserved sequence patterns
Macromolecular structure	13,000 structures (~1,000 atomic coordinates each)	• Protein geometry measurements • 3D structural alignment techniques • Surface and volume shape estimates • Secondary, tertiary structure prediction • Intermolecular interactions • Molecular modeling (force-field calculations, molecular movements, docking predictions)
Genomes	40 complete genomes (1.6 million - 3 billion bases each)	• Structural assignments to genes • Characterisation of repeats • Genomic-scale censuses (characterisation of protein content, metabolic pathways) • Phylogenetic analysis • Linkage analysis relating specific genes to diseases
Gene expression	largest: ~20 time point measurements for ~6,000 genes	• Mapping expression data to sequence, structural and biochemical data • Correlating expression patterns
Other data		
Literature	11 million citations	• Knowledge databases of data from literature • Digital libraries for automated bibliographical searches
Metabolic pathways		• Pathway simulations

DNA

Humans and nearly all other species carry their genetic information in DNA, also known as deoxyribonucleic acid. The majority of the DNA in eukaryotic organisms (animals, plants, fungi, and protists) is kept in the cell nucleus, while some of it is kept in organelles like the mitochondria or chloroplast [44, 45, 46]. Prokaryotes, which include bacteria and archaea, only store their DNA in the cytoplasm. DNA is compacted and organized within the chromosomes by chromatin proteins like histones. These little structures direct how DNA interacts with other proteins, helping to regulate which regions of the DNA are transcribed.

All known living things, including some viruses, are built and function according to the genetic instructions found in deoxyribonucleic acid (DNA) [44, 45, 46], a biological macromolecule. Long-term information storage is the primary function of DNA molecules. Since DNA carries the instructions required to build other components of cells, such as proteins and ribonucleic acid (RNA) molecules, it is frequently compared to a collection of blueprints, a recipe, or a code. Although other DNA sequences have structural functions or are involved in controlling how this genetic information is used, genes are the DNA segments that carry this genetic information. In terms of its chemical makeup, DNA is made up of two long polymers of nucleotides, which are simple units with backbones consisting of sugars and phosphate groups connected by ester bonds. These two strands are anti-parallel because they move in the opposite directions of one another. Each sugar has one of four categories of bases attached to it. Adenine, cytosine, guanine, and thymine are the four bases that can be found in DNA (T). The sugar/phosphate is joined with these four bases to create a full nucleotide. Information is encoded by the arrangement of these four bases along the backbone. The genetic code, which determines the order of the amino acids in proteins, is used to read this information. Transcription is the process of copying segments of DNA into the corresponding nucleic acid RNA in order to read the instructions. DNA is arranged into lengthy structures inside cells called chromosomes. Prior to cell division, these chromosomes are replicated through a process known as DNA replication.

In two ways, DNA actually has a fundamental impact on the various biochemical processes that go on within living things. The first is that it includes the blueprints for the manufacture of proteins, which are fundamental molecules for all living things. The second function of DNA in life is as a carrier of genetic information, specifically the blueprints for proteins, from one generation to the next.

- **Genes:** A contiguous stretch of DNA called a gene, which is a unit of heredity and provides the instructions needed to make a protein. A certain order of bases or nucleotides found in DNA molecules serves as the genetic code for genes. Prokaryotes often have circular chromosomes, while eukaryotes typically have linear chromosomes. A cell's genome is made up of its set of chromosomes; the human genome includes 46 chromosomes and about 3 billion base pairs of DNA.
- **Gene expression:** A protein-coding gene and its protein are separated in all species by two fundamental processes. The DNA containing gene must be transcribed from DNA to mRNA first, then it must be translated from messenger RNA (mRNA) to protein. Even though RNA-coding genes do not transform into proteins, they must nevertheless go through the first stage. Gene expression is the process of creating an RNA or protein molecule that is physiologically functional, and the resulting molecule is referred to as a gene product.
- **RNA:** Single-stranded RNA molecules are nucleic acids made up of nucleotides. As it is involved in the transcription, decoding, and translation of the genetic code to generate proteins, RNA plays a significant role in protein synthesis. Ribonucleic acid, or RNA, is a type of nucleic acid that, like DNA, has three different parts: a nitrogenous base, a five-carbon sugar, and a phosphate group. **Adenine (A), guanine (G), cytosine (C), and uracil (U)** are nitrogenous bases found in RNA**.**

  RNA is not necessarily linear despite being single-stranded. It can fold into intricate three-dimensional shapes and create **hairpin loops**. When this occurs, the nitrogenous bases bind to one another. Adenine pairs with uracil (A-U) and guanine pairs with cytosine (G-C). RNA molecules are produced in the nucleus of our cells and can also be found in cytoplasm. The three primary types of RNA molecules are messenger RNA, transfer RNA and ribosomal RNA.

Proteins

Proteins are organic substances comprised of amino acids organized in a linear chain and folded into a globular form. They are sometimes referred to as polypeptides [44]. Proteins play a crucial role in nearly every cellular process and are fundamental components of organisms, just like other biological macromolecules like polysaccharides and nucleic acids. Many proteins are enzymes, which are essential to metabolism and catalyze biological events. Actin and myosin in muscle and the proteins in the cytoskeleton, which constitute a system of folding that maintains cell shape, are examples of proteins with structural or mechanical activities. Other proteins have crucial roles in the cell cycle, immunological responses, cell adhesion, and cell signaling. Since animals cannot produce all the essential amino acids they require, they must get them from food, proteins are also crucial in animal diets. Animals convert ingested protein into free amino acids during digestion, which are then utilized in metabolism.

2.4.4 Bioinformatics Tasks for Biological Problems

The field of bioinformatics includes the study of genes, proteins, nucleic acid structure prediction, and molecular design with docking, among other biological challenges. The major biological problems and associated tasks involved in bioinformatics are described next.

Alignment and Comparison of DNA, RNA, and Protein Sequences

A sequence alignment shows where two or more sequences are similar and where they are different by mutually placing the sequences. These include the prediction and alignment of DNA, RNA, and protein sequences, as well as the assembly of DNA fragments. The alignment with the greatest number of correspondences and the fewest discrepancies is considered to be ideal.. It is the alignment with the highest score, but which may or may not be biologically meaningful. Alignment techniques can be divided into two categories: global alignment and local alignment. The number of matches between the sequences along the entire length of the sequence is maximized by global alignment [47, 250]. The local match between two sequences receives the greatest scoring in local alignment [48, 250]. Global alignment is the best option for sequences that are known to be extremely similar because it includes every character in both sequences from beginning to end. A local alignment is then utilized to detect these internal regions of high similarity if the sequences being compared are not identical along their whole lengths but instead contain brief segments that do. This is because a global alignment may overlook the alignment of these crucial parts. Most other bioinformatics tools are based on pairwise comparison and alignment of protein or nucleic acid sequences. The Smith-Waterman algorithm is a dynamic programming approach that enables accurate and thorough comparison of two (or more) biological sequences [48]. It refers to a programming method or algorithm that, when properly applied, efficiently performs all pairwise comparisons between the elements (such as nucleotide or amino acid residues) in two biological sequences. For alignment, spaces may need to be introduced within the sequences. The definition of a gap is a consecutive space. The end result is a mathematically ideal alignment of the two sequences, which isn't always biologically optimal. Given the particular parameters employed, a similarity score is also produced to indicate how similar the two sequences are.

A multiple alignment [49] places homologous sequences in a common column by arranging a group of sequences in that way. Regarding multiple alignment scoring, there are various conventions. One method simply adds the scores of all the induced pairwise alignments found in the multiple alignment. This equates to scoring each column of the alignment by the total of pair scores in this column [50] for a linear gap penalty. The distinctions between global, local, and other types of alignment are rarely made in a multiple alignment, despite the fact that they would make biological sense. Generally speaking, the multiple alignment will be overdetermined by a complete collection of optimum pairwise alignments among a given set of sequences. Pairwise alignments can be put together as long as no new pairwise alignment is added to a set of sequences that are already included in the multiple alignment. Closing loops must be avoided while assembling a multiple alignment from pairwise alignments.

Gene and Functional Site Identification from DNA Sequences

Biology has entered into a new era of genomics that has far-reaching consequences in human medicine and health. This leads to speedy identification and localization of complex disease genes. As the Human Genome Project enters its large-scale sequencing phase, gene identification has become extremely important. Gene finding often entails using algorithms to find sequence segments, typically genomic DNA, that are biologically useful. This specifically entails identifying the genes that code for proteins, but it may also involve locating other functional components like various RNA genes and regulatory areas.

Finding protein-coding genes within huge sections of uncharacterized DNA is challenging since the genomic sequence of the human body only contains a small percentage of protein-coding regions. Each protein in bacterial DNA is coded by a continuous region known as an open reading frame (ORF, beginning with a start codon and ending with a stop codon). The coding area in eukaryotes, particularly in vertebrates, is divided into a number of fragments called exons, while the remaining portions are known as introns. In essence, anticipating exon-intron structures is what it means to locate eukaryotic protein-coding genes in uncharacterized DNA sequences. [51–54] discuss a variety of works on the identification of protein-coding genes.

Splice sites or junctions, start and stop codons, branch points, promoters and terminators of transcription, polyadenylation sites, topoisomerase II binding sites, topoisomerase I cleavage sites, and various transcription factor binding sites are a few other categories of functional sites in genomic DNA that researchers have sought to identify. These nearby locations are referred to as signals, and the tools used to find them are sometimes termed signal sensors. Genomic DNA signals can be compared to extended and variable length portions like exons and introns, which are identified by various techniques that are sometimes referred to as content sensors. Determine the precise structure of genes in genomic sequences by identifying splice sites, introns, exons, start and stop codons, and branch points, which is a crucial subtask in gene prediction.

Promoter prediction and transcription factor binding site (TFBS) finding are crucial for understanding gene regulation and getting a better read on microarray expression data. With the aid of a promoter, a cell mechanism recognizes the start of a gene or gene cluster, which is required for the start of transcription. Every gene has a section in its DNA called a promoter, which tells the cellular machinery that a gene is coming up. For instance, the methionine-coding codon AUG also marks the beginning of a gene.

Different methods have been used to identify variations between collections of known promoter and non-promoter sequences [55, 56]. Except for CpG island genes, current promoter predictions are substantially less trustworthy than predictions of protein-coding regions due to the lack of protein coding signatures. Finding TFBS motifs within regulatory areas, like as promoters, can then be done either by enumeration or by alignment to find the enriched motifs.

Due to the rising number of fully sequenced genomes and the widespread usage of DNA chips, the identification of regulatory sites in DNA fragments has gained a lot of attention. Under the assumption that there are at least 30,000 possible promoter regions in the human genome, one for each gene, experimental analyses have only successfully found less than 10% of these regions.

Protein Functional Site Prediction

Another significant issue in bioinformatics is the prediction of functional protein locations. It is a crucial topic in research on protein function and, consequently, in medication development. However, it has been discovered that the relationship between functional sites and consensus patterns may not always be straightforward, necessitating the creation and application of more complex and, thus, more potent pattern recognition algorithms. For instance, bio-basis function neural networks [63–66], feed-forward and recurrent neural networks [61–62], feed-forward and recurrent neural networks trained with back-propagation [57–59], Kohonen's self-organizing map [60], and support vector machines [67] have all been used to predict various functional sites in proteins, such as protease cleavage sites of HIV (human immunodeficiency virus) and Hepatitis C virus.

DNA Structure Prediction

Numerous biological activities depend heavily on DNA structure. Base stacking energy, propeller twist angle, protein deformability, bendability, and position preference are just a few of the DNA structure characteristics that have been described using different dinucleotide and trinucleotide scales [68]. Three-dimension DNA structure and its organization into chromatin fibres are essential for its functions and are used in protein binding sites, gene regulation, triplet repeat expansion diseases, and other areas.

RNA Structure Prediction

An RNA molecule is considered as a string of n characters $R = r_1 r_2 \cdots r_n$ such that $r_i \in A, C, G, U$. n could be in the thousands but is typically in the hundreds. The molecule's secondary structure is a collection S of stems, with each stem made up of a series of successive base pairs $(r_i r_j)$ (e.g., GU, GC, AU). Here, $1 \leq i \leq j \leq n$ and $(r_i \ and \ r_j)$ are connectedthrough hydrogen bonds. If $(r_i, r_j) \in S$, in principle we should require thatri be a complement to r_j and that $j - i > t$, for a certain threshold t (because it is known that an RNA molecule does not fold too sharply on itself).

There are essentially two different ways to anticipate the RNA secondary structure automatically. The first involves adding contributions from each base pair, bulged base, loop, and other features to the overall free energy minimization [69–71]. The second method [69–71] is more empirical and looks for combinations of nonexclusive helices with the most base pairings in order to satisfy the requirement of a biomolecule having a tree-like structure. Dynamic programming techniques are the most popular within the latter [69–71]. For the prediction of RNA structure and its associated function, approaches for simulating the folding mechanism of an RNA molecule [71] and identifying key intermediate states are crucial.

Classification and Prediction of Protein Structure

Similar 3-D structures are produced by proteins with similar sequences. As a result, comparable sequences may produce identical structures, which is typically the case. Contrary to popular belief, identical 3-D structures do not always denote equivalent sequences. Because of this, homology and similarity are distinguished from one another. The databases contain instances of proteins that are homologous because they have virtually identical 3-D structures but do not show any observable or significant sequence similarity. Pairwise comparisons frequently miss tiny similarities that become apparent when numerous sequences are compared at once. They also struggle to reveal places that are conserved across an entire set of sequences. So, it makes sense to compare multiple sequences at once. In structural proteomics, a protein's three-dimensional structure is predicted based on its primary amino acid sequence [72]. One of the most difficult issues in bioinformatics is determining how a protein functions given that its structure determines how it functions. The sequence of amino acids that make up a protein is its primary structure, which is the first of the five levels of protein structure. (ii) The spatial arrangement of the atoms that make up the primary protein backbone is known as the secondary structure of a protein. The local folding pattern constructed from certain secondary structures is known as the super-secondary structure or motif (iii). (iv) Tertiary structure, or the folding of the entire protein chain, is created by compressing secondary structural components connected by loops into one or more compact globular units called domains. (v) Several protein subunits may be organized in a quaternary structure within the final protein.

In the same environment, protein sequences nearly invariably fold into the same structure. The structure of the protein is also influenced by Vander Waals interactions such as electrostatic, hydrophobic, hydrogen bonding, and others. Given its fundamental sequence, a protein's structure is the subject of numerous ongoing efforts. The spatial coordinates of every atom in a protein molecule must be calculated in a typical computation of protein folding, starting with the initial configuration and progressing to the final minimum-energy folding configuration [73, 74]. Based on homology to existing proteins, sequence similarity algorithms can forecast the secondary and tertiary structures. Methods for secondary structure predictions include those suggested by Garnier, Osguthorpe, and Robson [76], Chou and Fasman [75], and others. This can also be accomplished using nearest neighbor approaches [78] and neural networks [77]. Methods for predicting tertiary structures [73, 74] are based on stochastic conformational space searches, molecular dynamics, and energy minimization.

Molecular Design and Molecular Docking

When two molecules are close to one another, it may be energetically advantageous for them to form a strong bond. The prediction of energy and physical configuration of binding between two molecules is known as the molecular docking problem. The process of docking a tiny molecule that is a described drug to an enzyme one wants to target is a common application in drug design. For instance, the HIV protease enzyme in the AIDS virus is crucial for the virus' ability to replicate. At a specific active spot on its surface, the protease's chemical reaction occurs. Small chemicals known as HIV protease inhibitors bind to the active site of the enzyme and remain there, preventing the enzyme from performing as it should. Using docking tools, we may assess a medication design by determining if it will successfully bind tightly to the enzyme's active site. Designers can modify the therapeutic molecule based on the outcome of docking and the consequent docked configuration [79].

Studying Evolutionary Relationships with Phylogenetic Trees

Evolution is the gradual process of change that all organisms on Earth experience. Construction of trees with leaves representing current species and interior nodes representing hypothetical ancestors serves to illustrate the evolutionary history of today's species as well as how species link to one another in terms of common predecessors. Phylogenetic trees are the name given to this class of labeled binary trees [80]. The evolutionary link is investigated by phylogenetic analysis. On the basis of similarities between contemporary items, phylogenies are rebuilt. The goal of the phylogenetic tree reconstruction challenge is to identify the specific permutation of the provided objects that best meets the specified criteria. To tackle this issue, several algorithms are suggested [80].

Analysis of DNA Microarray Technology based Gene Expression Data

Gene expression is the process by which a gene’s coded information is converted into the protein and other functional product. Expressed genes include those that are transcribed into mRNA and then translated into protein, and those that are transcribed into RNA but not translated into protein (e.g., transfer and ribosomal RNAs). Not all genes are expressed, and gene expression involves the study of the expression level of genes in the cells under different conditions. Conventional wisdom is that gene products which interact with each other are more likely to have similar expression profiles than if they do not [1-3, 13-15].

DNA Microarray technology is a high–throughput [13-15] biotechnology using which expression levels of several thousands of genes is measured at the same time for a particular sample/experiment or for a particular time point of a particular sample. The outcome of DNA microarray technology is a gene expression data matrix in which outcomes of several microarrays are combined containing same set of genes and different samples or different time points of a particular sample.

Analysis of microarray gene expression data is very crucial in the field of biomedical research for deadly disease early prediction, prognosis and advancement of therapy, identification of functionality of novel genes, gene regulatory network identification [13-30] etc.

Other potential bioinformatics tasks for biological issues include the following: (i) characterizing the protein content and metabolic pathways between various genomes; (ii) identifying protein functional sites; (iii) identifying and analyzing interacting proteins; (iv) characterization of repeats from genomes; (v) gene mapping on chromosomes; (vi) analysis of genomic-scale censuses; (vii) assignment and prediction of gene products; (viii) large-scale analysis of gene expression data; (ix) mapping expression data to sequence, structural, and biochemical data; (x) creation of digital libraries for automated bibliographical searches; (xi) creation of knowledge bases of biological information from the literature; (xii) creation of DNA analysis techniques for use in forensics; and so forth [81–83, 84].

In the next section, a brief description of DNA microarray and gene expression data is given.

2.5 DNA Microarray based Gene Expression Data

DNA Microarrays have made it possible to successfully see how cells work and have the ability to simultaneously investigate the expression of tens of thousands of genes. The general procedure for getting the gene expression profiles from a DNA microarray is shown in Figure 2.2. These data on gene expression can be used as inputs for extensive data analysis, such as to better comprehend and categorize the genes into healthy and diseased cells. By counting all the mRNA that has been transcribed in a genomic system, gene expression shows how genotype changes into phenotype. Differential display, Microarray hybridization, RNA sequencing, Serial Analysis of Gene Expression (SAGE), and subtractive hybridization are just a few of the many standardized methods available for identifying variations in gene expression. All of these strategies' specifics are covered in detail in [85, 86].

The method for getting microarray data is shown in Figure 2.2 and entails collecting sample tissues from both malignant and healthy human tissues. The samples are then isolated for their mRNA using either a column or a solvent like phenol-chloroform. After that, both tissues are combined on a Microarray plate to form the cDNA label, which is subsequently hybridized.

The final step is to measure the relative fluorescence intensities using a microarray scanner in order to evaluate the results and save the data as an array for further use.

2.5.1 Microarray experiment

The hybridization process known as a microarray experiment compares the relative amounts of cellular mRNA obtained from two tissue samples. When single-stranded DNA or RNA molecules combine to form double-stranded complexes, the hybridization reaction occurs. As shown in Figure 2.2, there are generally four steps involved in determining the gene expressions in a microarray experiment. The first step is sample preparation and labeling, which involves RNA extraction from a particular tissue and labeling according to the technique chosen. Hybridization is the second stage. It is the stage where the DNA or RNA probes and labels that are intended for heteroduplexes are base-paired on the glass surface using the Watson-Crick method. Hybridization can be detected electrochemically, visually, or by using mass sensitive equipment. The third stage, washing, is where extra solution from the hybridization array is gotten rid of. In order to lessen the impact of the sensitivity and background level of the entire microarray, non-specifically bound cRNA is removed from the surface. The hybridized image of the array is created at the image acquisition stage, which concludes the process.

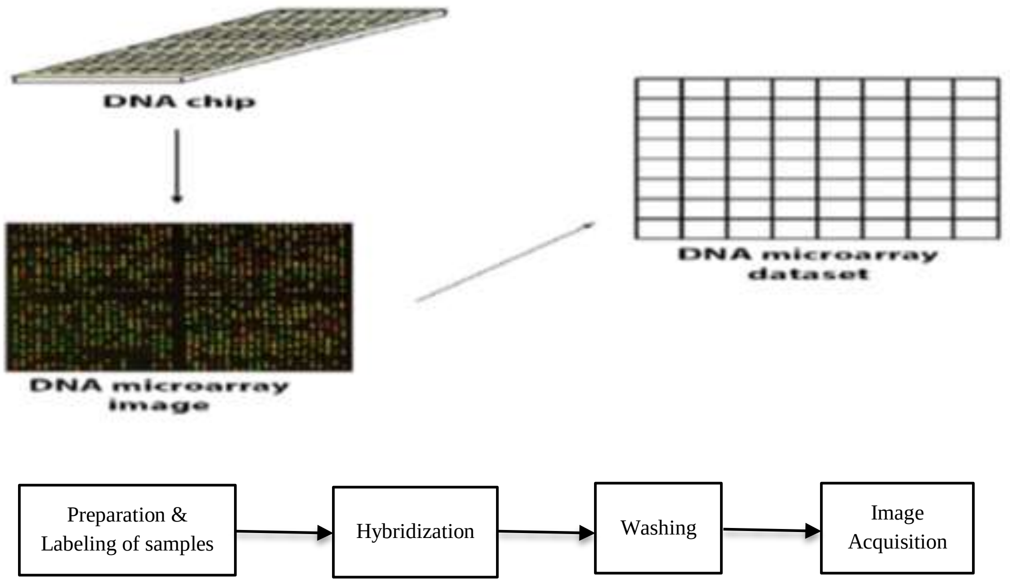


Figure. 2.2 Steps involved in Microarray Experiment

2.5.2 Microarray data

Large matrices ($M \times N$) are used to organize and store the data produced by microarray experiments. As shown in Figure 2.3, each Microarray data matrix consists of the samples presented in rows and the genes (features) in columns. Microarray data is in the form of a M by N matrix. Every single cell in a sample has a certain level of gene expression, where M is represents rows (samples) while N represents columns (genes). x_{ij} denotes the expression level of the gene j and the condition or sample i. Where i ranges between 1 and M, and j from 1 to N.

$$\mathrm{G} = \begin{pmatrix} S_{11} & S_{12} & \ldots & S_{1p} & \ldots S_{1n} \\ S_{21} & S_{22} & \ldots & S_{2p} & \ldots S_{2n} \\ S_{31} & S_{32} & \ldots & S_{3p} & \ldots S_{3n} \\ . & . & \ldots & . & \ldots . \\ . & . & \ldots & . & \ldots . \\ S_{m1} & S_{m2} & \ldots & S_{mp} & \ldots S_{mn} \end{pmatrix}_{m \times n}$$

Figure. 2.3 Microarray Data

2.6 Conclusion and Discussion

The major advancement in the field of molecular biology has led to an explosive growth in the biological information. The meaningful interpretation of these large volumes of biological data is increasingly becoming difficult. Machine learning techniques are very useful to store, analyze, and interpret these biological data. In this regard, this thesis presents some classification and clustering techniques based methodologies to solve certain problems of gene expression data in bioinformatics. To understand various problems of bioinformatics, the basic concepts of molecular biology are reviewed in this chapter. Next chapter presents a new framework-based neighborhood formation technique for existing *K*NN and its several versions based preprocessing techniques to predict missing values in gene expression data more accurately.

Chapter 3

Pre-processing on Microarray Gene Expression Data based on Clustering technique: A Framework for Neighborhood Configuration to Improve the KNN based Imputation Algorithms on Microarray Gene Expression Data

3.1 Introduction

Due to innate experimental problems [31] in DNA microarray technology, values of several cells of the data matrix are lost producing an incomplete data matrix containing missing entries. However, the requirement of a complete data matrix as input in the analysis algorithms (example: classification, clustering, and other model-based algorithms [14-30]), makes the prediction of these missing entries very essential. The simplest approach is to repeat the experiment but owing to the high financial implication, as well as frequent unavailability of the biological sample, such repetition is often not practicable. Several prediction methods have hence been suggested to overcome this problem. These methods are divided into three broad categories [31, 87]. The first approach is to delete the entire samples or genes containing the missing entries. The drawback of this method is that valuable information may be lost and as a result, the performance of the analysis methods will be degraded. In the second approach, the missing entries of a gene are substituted by some mathematical operations such as mean or mode of the expression values of that gene for all the samples. The drawback of this method is that all missing entries of a gene are filled up by the same value which distorts the original structure of the gene expression data. As a consequence, it may affect the performance of the analysis algorithms. The methods that belong to a third approach have overcome the above-mentioned problems and impute missing entries by taking correlated information from the non-missing part of the data matrix. A huge number of methods have been developed in this category. Depending on the type of information used, these methods are divided into four groups (1) global methods (2) local methods (3) hybrid methods, and (4) knowledge assisted methods [31, 87].

Global methods [88, 89, 90] impute missing values by taking information from the entire data matrix considering that a global covariance structure exists in the dataset. Global methods work well for large datasets where local dependency among the genes does not exist [31, 87]. Local methods, on the other hand, predict the missing values based on local substructure only [31, 87]. These methods predict missing values by working in two phases. First, these methods identify neighbor genes for each target gene and then estimate the missing value by forming a relationship between each target gene and its neighbors. Estimation is performed based on either weighted average [90, 91, 92, 93] or numerical approach [94-100]. The third category represents hybrid methods [31, 87, 101, 102]. These methods integrate multiple imputation methods. Some hybrid methods consider both global and local correlation-based information for imputation [31, 87, 101, 102]. This type of integration increases the complexity of the imputation process. The last category-based methods known as knowledge assisted methods [31, 87, 103-106] take the help of external domain knowledge as well as correlation information among genes from the data matrix to achieve better imputation accuracy. The limitation of these methods is that they require external domain knowledge information and hence will not work properly for newly explored cases where no such external domain knowledge information exists [31, 87].

Although many efficient missing value estimation techniques exist in the literature, performances of these methods vary greatly depending on the diverse nature and characteristics of the datasets [31, 87]. The performance also depends on several application areas [31, 87]. So, researchers have still been working on new missing value estimation techniques. Among the above mentioned four types of categories, local weighted average -based methods are simple and usually generate consistent results [105] in almost all cases. Due to this reason, the most popular weighted average-based method namely, the *K*NNimpute and its several modified versions have been widely used in different microarray data analysis tools like SAM, PAM, and MAANOVA [106-108] although their prediction accuracy is low compared to the numerical methods.

The *K*NNimpute is the simplest and most widely used missing value estimation method for microarray gene expression data. However, not much attention has of late been given to remove its drawbacks. Rather, more and more researchers have been found to focus on developing numerical approach-based imputation techniques To address the above-mentioned issues, here first a primary framework is proposed by introducing PEH distance which is a integration of Pearson correlation coefficient and Euclidean distance in a new proposed version of *K*NN named Iterative Sequential *K*-Nearest Neighbor Imputation algorithm but this framework has several limitations. In this framework, some of the shortcomings are overcome using a novel distance combining both Euclidian and Pearson correlation similarity measures [109]. However, we have observed that even though this framework is capable of producing good results, in some cases, it fails in the presence of the oppositely co-expressed pattern-based genes and the scaling pattern-based genes. This drawback has later been elaborated in section 3.4.2, but it may be mentioned here that the failure originates from its inability to handle the shifting, scaling, and inverted patterns in a proper manner in the dataset. Due to this drawback, the algorithm is incapable of involving in the weighted average computation procedure, such candidate genes which may not be in the nearest Euclidian distance neighborhood, and yet which carry considerable correlation information with respect to the target gene in terms of the underlying pattern. Overall, we feel that more attention needs to be devoted to identify, address, and overcome the shortcomings of the *K*NNimpute because of the inherent simplicity of this method.

So, in the second work, we have proposed a more robust framework for more accurate neighborhood formation in *K*NNimpute and its several versions to improve upon the weighted average procedure in these methods [110].

The framework is developed based on a hybrid distance and gene transformation procedure. The motivation behind this work is that the neighborhood for each target gene will be constructed in such a way that the weighted average procedure will run only on the maximum positively co-expressed and magnitude wise closest genes. The integrated K*N*Nimpute method with the new framework is renamed as a modified *K*NN imputation method (M*K*NNimpute). Similarly, the modified sequential *K*NN imputation method (MS*K*NNimpute), modified iterative *K*NN imputation method (MI*K*NNimpute), and modified iterative sequential *K*NN imputation method (MIS*K*NNimpute) have also been proposed. The performances of these modified methods are compared with their corresponding established versions on different categories of microarray gene expression datasets. From the experimental results, it has been found that the proposed methods significantly outperform their corresponding existing counterpart versions in all cases for different types of datasets, and among these three, the modified iterative sequential version (MIS*K*NNimpute) is found superior. From experimental results, it is also found that the MIS*K*NNimpute method is also comparable with the robust numerical imputation methods for all types of datasets and other versions are also comparable with robust numerical imputation methods for local structured based datasets.

3.2 Existing Nearest Neighbor Imputation Techniques

Notation

G is the input data matrix containing m genes and n samples where $m \gg n$. The i^{th} row/gene is represented by g_i and g_{ij}or $g_i(j)$ represents the information of the gene i in the j^{th} column/experimental condition. A missing value in the i^{th} gene at p^{th} column position is represented by α. $I_{m \times n}$is the missing indicator matrix for tracking missing positions in G. If any entry r_{ij} in matrix I is 1, it means that the expression value of the corresponding position in matrix G i.e. g_{ij} the position is present and if r_{ij} is 0 then the value of the corresponding position i.e. g_{ij} is missing.

A gene with one or more missing entries, currently considered in the imputation procedure, is treated as a target gene and the remaining genes are considered as candidate genes with respect to that target gene.

3.2.1 *K*-Nearest Neighbor Imputation (*K*NNimpute)

K-Nearest Neighbor imputation technique (*K*NNimpute) is the oldest and most popular weighted-average based imputation technique [90]. *K*NNimpute predicts the missing value of a target gene by taking information from its nearest neighbor genes. The nearest neighbor genes are selected based on Euclidean distance. The Euclidean distance d_{ik} is calculated as shown in the equation (1).

$$d_{ik} = \sqrt{\frac{\sum_{j=1}^{n} r_{ij} r_{kj} \left(g_{kj} - g_{ij}\right)^2}{\sum_{j=1}^{n} r_{ij} r_{kj}}} \qquad (3.1)$$

Here i and k are representing the target gene g_i and the candidate gene g_k. The missing value y_{ij} at column j of the target gene g_i is calculated by taking the weighted average of K nearest neighbor genes for column j as shown in the equation (3.2).

$$y_{ij} = \sum_{k=1}^{K} w_{ik}\, g_{kj} \qquad (3.2)$$

Here, w_{ik} represents the weight of the k^{th}neighbor gene of the target gene $g_i$and is calculated as shown in equation (3.3).

$$w_{ik} = \frac{1/d_{ik}}{\sum_{k=1}^{K} 1/d_{ik}} \qquad (3.3)$$

3.2.2 Sequential *K*-Nearest Neighbor Imputation (S*K*NNimpute)

S*K*NNimpute is an improved version of *K*NNimpute [91]. Similar to *K*NNimpute, for every missing position in the target gene S*K*NNimpute selects *K*-neighbor genes using Euclidean distance as mentioned in the equation (1). There are two main differences between S*K*NNimpute and *K*NNimpute. One difference is that, in S*K*NNimpute, initially the entire gene expression matrix (G) is divided into two sub-matrices: one containing genes with missing entries ($G_{missing}$) and the other containing genes with no missing entries ($G_{nomissing}$). Instead of selecting the *K*-nearest neighbor genes of the target gene from the whole matrix, it selects genes from the $G_{nomissing}$ matrix. The second difference is that S*K*NNimpute sequentially predicts the missing entries starting with the target gene containing the least number of missing positions. The already predicted values are used for subsequent imputations.

3.2.3 Iterative *K*-Nearest Neighbor Imputation (I*K*NNimpute)

I*K*NNimpute is another improved version of the *K*-nearest neighbor imputation [92]. In I*K*NNimpute, the missing positions in G are initially replaced by the corresponding row (gene) averages to form a complete gene expression matrix $G_{complete}(0)$. For each target gene, it selects *K*-nearest neighbor genes using Euclidean distance as mentioned in the equation (1) and simultaneously imputes all the missing positions in that gene using the weighted average computed according to the equation (2). After imputing all the missing entries in all the target genes, again the complete gene expression matrix $G_{complete}(1)$ is formed. This process is iterated until the difference between the estimated values of two successive iterations denoted by $\hat{y}_j{}^{(h)}$ and $\hat{y}_j{}^{(h-1)}$reaches a given threshold. In $G_{complete}(h)$, h indicates the iteration number. The sum of squared differences between the estimated values obtained in the last two iterations is calculated using the equ-ation (4) given below.

$$\psi^{(h)} = \sum_{j=1}^{z} \left(\hat{y}_j{}^{(h-1)} - \hat{y}_j{}^{(h)}\right)^2 \qquad (3.4)$$

If $\psi^{(h)}$ is less than a given threshold, then the process will stop; otherwise, iteration will be continued. Here z represents the number of missing entries and h indicates iteration.

3.3 Impact of different Distance Measures and their drawbacks in Predicting Missing Values in Gene Expression Data

In the above-mentioned weighted average-based imputation methods, the imputation accuracy can be improved if the neighborhood is constructed with those genes which are most similar based on the positively co-expressed pattern, as well as closer to the target gene in terms of magnitude [109]. However, the existing distances are not capable enough to satisfy the above-mentioned criteria in presence of different types of gene patterns. Here, to represent the drawbacks of the different distance metrics, some figures (Figures 3.1-3.3) are considered. In every figure, there is a target gene and a set of different pattern-based candidate genes to show the drawbacks of these metrics with respect to the different patterns used in the above-mentioned imputation methods for neighborhood formation.

3.3.1 Euclidean Distance

In the above-mentioned imputation techniques, Euclidean distance [34] is used for neighborhood formation. It is a well-accepted distance measure to calculate the distance between two objects. The Euclidean distance between two d-dimensional objects $P = (p_1, p_2, \dots., p_d)$ and $Q = (q_1, q_2, \dots., q_d)$ can be defined as in the equation (3.5) below:

$$E(P,Q) = E(P,Q) = \sqrt{\textstyle\sum_{i=1}^{d} (p_i - q_i)^2} \qquad (3.5)$$

If P and Q are close in terms of their expression value, their Euclidean distance will be lower; otherwise, it is high. It means Euclidean distance between two objects is small if the two objects show similarity in magnitude or expression value.

In gene expression data, it is already known [34] that the biological function based similar genes are also pattern-based similar (co-expressed), but may not still be similar expression value-wise. So, Euclidean distance is not a good measure to select co-expressed genes. This is described in Figure 3.1. In Figure 3.1, a target gene and seven candidate genes are shown. Among them, gene3, gene6 are highly positively co-expressed and gene2, gene4, and gene7 are highly negatively co-expressed with the target gene according to the Pearson correlation coefficient. In Table 1, the corresponding expression values and the Euclidean distance and Pearson correlation coefficient for every gene in Figure 3.1 are given. The gene1 is magnitude wise closest to the target gene but not so in terms of pattern-based similarity. The gene5, on the other hand, is much more pattern-based similar to the target gene but is scaled in a positive direction. If K nearest neighbor genes of the target gene are selected using Euclidean distance value and K is set at four then gene1, gene2, gene3, and gene5 will be selected. From this figure, it is clear that irrespective of the pattern, Euclidean distance selects neighbor genes which are magnitude wise closer to the target gene. So, in the neighborhood formation of the target gene, Euclidean distance cannot always pick up the pattern-based most similar genes.

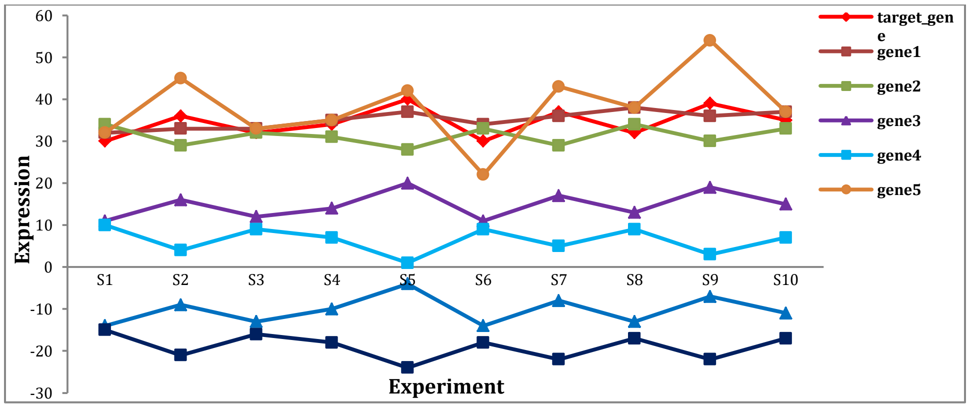


Figure3.1. Microarray Gene Expression Patterns showing drawbacks of Euclidean distance

Table 3.1: Euclidean distance and Pearson Correlation coefficient for genes in figure1 with respect to target_gene

	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**Pearson**	**Euclidean**
target_gene	30	36	32	34	40	30	37	32	39	35		
gene1	32	33	33	35	37	34	36	38	36	37	0.488901	9.486832981
gene2	34	29	32	31	28	33	29	34	30	33	-0.858934	19.49358869
gene3	11	16	12	14	20	11	17	13	19	15	0.994941	62.31372241
gene4	10	4	9	7	1	9	5	9	3	7	-0.965924	90.97801932
gene5	32	45	33	35	42	22	43	38	54	37	0.834125	21.3541565
gene6	-14	-9	-13	-10	-4	-14	-8	-13	-7	-11	0.97532	141.689802
gene7	-15	-21	-16	-18	-24	-18	-22	-17	-22	-17	0.895396	170.2615635

3.3.2 Pearson Correlation Coefficient

On the contrary, the Pearson correlation coefficient [34] is a well-known metric to measure the correlation-based similarity between two objects. Pearson correlation coefficient is also used as a similarity measure in KNNimpute, SKNNimpute, and IKNNimpute. The Pearson correlation between two objects P and Q, each of which is a d-dimensional vector, i.e. $P = (p_1, p_2, \dots, p_d)$ and $Q = (q_1, q_2, \dots, q_d)$, can be defined as in equation (3.6) below:

$$PC(P,Q) = \frac{\sum(P-\bar{P})(Q-\bar{Q})}{\sqrt{\sum(P-\bar{P})^2 \times (Q-\bar{Q})^2}} \quad (3.6)$$

It varies between -1 to +1. If two objects are perfectly positively correlated then the Pearson correlation coefficient value between them is 1 and if two objects are perfectly negatively correlated then the Pearson value between them is -1.

Using this measure, it is possible to select the most similar positively co-expressed and negatively co-expressed neighbor genes of the target gene. The drawback of this measure is that it can select any type of pattern-based similar gene (like perfect pattern-based similar, perfect inverted pattern-based similar, shifting, or scaling pattern-based similar in positive/negative direction) which adversely affects the weighted average procedure. In Figure 3.2, except the target gene, all other genes are candidate genes. According to absolute Pearson correlation value (which is $|PC(P,Q)|$) if K nearest neighbor genes are chosen and K is set at 5, then gene2, gene3, gene4, gene6, gene7 will be selected. Among them, gene2 and gene3 are positively co-expressed shifting patterns while gene6 is positively co-expressed scaling pattern with respect to the target gene. gene4 is a negatively co-expressed shifting pattern and gene7 is a negatively co-expressed scaling pattern. The absolute Pearson correlation coefficient values of these genes are equal for that target gene. So, when the Pearson correlation coefficient is used for neighborhood formation, it selects any type of pattern-based most similar genes and cannot distinguish between the shifting patterns and scaling patterns. If the weight of each of the neighbor genes is calculated using Pearson correlation coefficient value with respect to the target gene according to the equation (3.2), then their weights are the same and it negatively affects the weighted average procedure-based imputation. Naturally, this in turn affects the prediction process at the time of calculation of the weighted average.

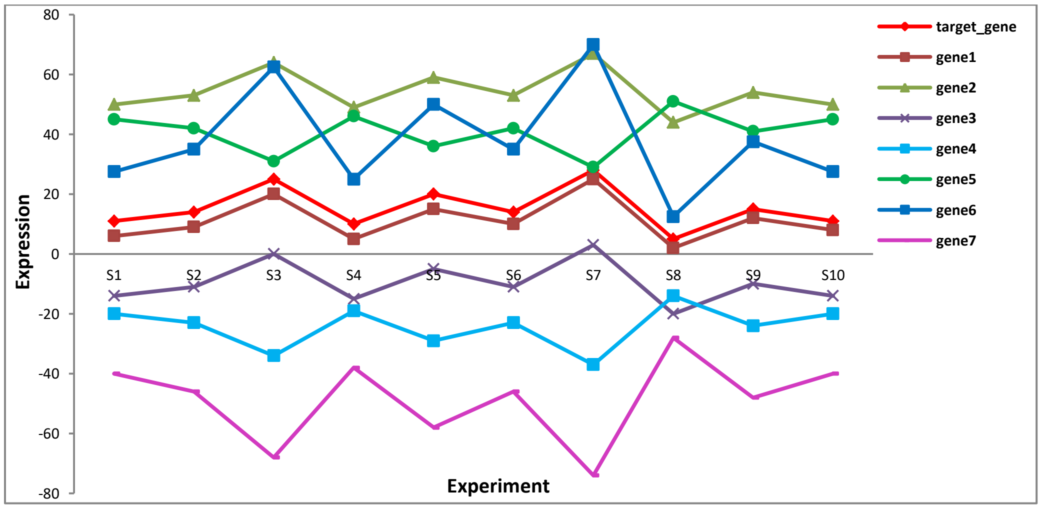


Figure 3.2. Microarray Gene Expression Patterns showing drawbacks of Pearson Correlation Coefficient

Table 3.2: Pearson correlation coefficient for genes in Figure2 with respect to target_gene

	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**Pearson**
target_gene	11	14	25	10	20	14	28	5	15	11	
gene1	6	9	20	5	15	10	25	2	12	8	0.990134
gene2	50	53	64	49	59	53	67	44	54	50	1
gene3	-14	-11	0	-15	-5	-11	3	-20	-10	-14	1
gene4	-20	-23	-34	-19	-29	-23	-37	-14	-24	-20	-1
gene5	45	42	31	46	36	42	29	51	41	45	-0.999364
gene6	27.5	35	62.5	25	50	35	70	12.5	37.5	27.5	1
gene7	-40	-46	-68	-38	-58	-46	-74	-28	-48	-40	-1

3.4 Proposed Primary Framework: Pearson and Modified Euclidean based Hybrid (PEH) Distance Measure

3.4.1 PEH Hybrid distance based First Framework

To recover the above-mentioned drawbacks of Euclidean distance and Pearson correlation coefficient a new framework is proposed via introducing a novel hybrid (PEH) distance in a new KNN version named iterative sequential *K*-nearest neighbor imputation. The hybrid distance is a combination of Euclidean distance and Pearson correlation coefficient.

The proposed Hybrid (PEH) distance is defined in the equation (3.7).

$$PEH(g_i, g_j) = \begin{cases} (1 - |PC(g_i, g_j)|) \times NE(g_i, g_j)\,, & if PC(g_i, g_j) > 0 \\ (1 - |PC(g_i, g_j)|) \times \min\{NE(g_i, g_j), NE(g_i, -g_j)\}, & if PC(g_i, g_j) < 0 \end{cases} \quad (3.7)$$

$|PC(g_i, g_j)|$ and $NE(g_i, g_j)$ represent the absolute value of the Pearson correlation co-efficient and Normalised Euclidean distance between the genes g_i and g_j respectively. Normalised Euclidean distance is defined below:
$NE(g_i, -g_j)$ represents the Euclidean distance between gene g_i and sign-flipped g_j (i.e. taking the negative values of the samples of gene g_j to get the mirror gene).

The normalised Euclidean distance is a Euclidean distance which normalizes the distance in the range of 0–1. The modified Euclidean distance between two objects p and q with n attributes/features is introduced in equation (3.8).

$$NE(p, q) = NE(q, p) = \sqrt{\frac{\sum_{i=1}^{n} \frac{(q_i - p_i)^2}{MD_i^2}}{n}} \quad (3.8)$$

Here, MD_i is the maximum difference along the ith attribute, and it is calculated as in equation (3.9).

$$MD_i = \begin{cases} 1 & if\ Max_i = Min_i \\ Max_i - Min_i & otherwise \end{cases} \quad (3.9)$$

In the first framework PEH distance is applied in the proposed IS*K*NN impute. The proposed iterative sequential *K*-nearest neighbor imputation procedure (IS*K*NNimpute) is another improved version of *K*NNimpute and combines the concepts of S*K*NNimpute and I*K*NNimpute. In every iteration, like S*K*NNimpute it also sequentially imputes the missing values starting from the gene that has least number of missing entries, and uses the imputed values for the later imputation. According to *K*NN principle, in every iteration, for every target gene i this procedure finds the nearest *K*-neighbor genes which are selected using the proposed *PEH* distance as mentioned in equation (3.5) and then estimates each missing value $\hat{y}_{ij}$ in that given target gene simultaneously using the weighted average of the selected *K*-neighbor genes as mentioned in equation (3.10). The weight $\overline{w}_{ik}$ is calculated using equation (11). A single iteration is completed when all missing entries of all target genes are imputed. Now for every missing position, the sum of squared differences $\delta^{(h)}$ between the estimated values obtained in last two iterations is calculated using equation (3.12). If $\delta^{(h)}$ is less than a given threshold τ, then the process will stop otherwise iteration will be continued.

$$\hat{y}_{ij} = \begin{cases} \sum_{k=1}^{K} \overline{w}_{ik} \left|x_{kj}\right| & if \quad NE(g_i, g_k) < NE(g_i, -g_k) \\ \sum_{k=1}^{K} \overline{w}_{ik}\, x_{kj} & Otherwise \end{cases} \tag{3.10}$$

$x_{kj}$is the element of kth gene at jth sample.

$$\overline{w}_{ik} = \frac{1/H_{ik}}{\sum_{k=1}^{K} 1/H_{ik}} \tag{3.11}$$

Where H_{ik} represents the PEH distance between gene i and gene k.

$$\delta^{(h)} = \sum_{j=1}^{z} \left(\hat{y}_j{}^{(h-1)} - \hat{y}_j{}^{(h)}\right)^2 \tag{3.12}$$

IS*K*NNimpute Algorithm

Input: A gene expression matrix X containing$m$ genes and n samples. Here the variable h indicates iteration number and h is initialized to 0. Let y represents number of target genes which contains missing values and z represents total number of missing entries in the given dataset.

Output: Missing values are imputed in matrix X.

Step 1. Sort all target genes in ascending order according to their missing rates.

Step 2.Mark all y number of target genes as unprocessed.

Step 3. Repeat step 3 for each target gene in the sorted list, to form the complete matrix $X_{complete}(0)$

a) Calculate the PEH distance from the target gene to each candidate gene.
b) Choose K nearest genes following the K-nearest neighbor principle.

c) Impute all the Missing values in that target gene simultaneously using the equation (3.10).

d) Mark the target gene as processed.

Step 4. Repeat step 4 until $\delta^{(h)} < \tau$ for h = 1, 2, 3......

a) Mark all the target genes as unprocessed.

b) Repeat step 4(b) for each target gene in the sorted list to form the complete matrix $X_{complete}(h)$

i. Calculate the PEH distance (considering all genes taking values from $X_{complete}(h-1)$) from the target gene to each candidate gene.

ii. Choose K nearest genes following the K-nearest neighbor principle.

ii. Impute all the Missing values in that target gene simultaneously using the equation (3.10).

iii. Mark the target gene as processed.

c) Compute $\delta^{(h)}$using the equation (3.12).

Step 5. End

Here the value of τ is taken as 10^{-3}.

Although primary framework is prepared via applying PEH distance in IS*K*NN algorithm but PEH distance can be applied in *K*NN and other versions also.

Computational time complexity

IS*K*NNimpute algorithm consists of 3 steps mainly. In step 1 it sorts the genes according to their missing rate which involves $O(m \log_2 m)$ time complexity. Step3 is repeated y times where in worst case y is equal to m. In step 3(a) IS*K*NNimpute computes the PEH distance from the target gene i to all other gene involving $O(mn)$ complexity. Imputation of missing values in step 3(b) for target gene i takes maximum of $O(n)$ complexity as number of missing entries in any gene is not more than $n/2$ which is very negligible compared to number of gene m. So step3 requires $O(ymn) \approx O(m^2n)$ time complexity. Step4 is repeated until $\delta^{(h)} < \tau$ which is not more than 5 iterations as seen during experiments. The second step (b) in step4 does the same operations as in step3(a) and require $O(m^2n)$ time complexity. The last step in step4 calculates sum of the squared differences $\delta^{(h)}$ for each missing entry which is not more than $mn/2$(with maximum of 50% missing entries) and requires $O(mn)$ complexity. Hence the time complexity of IS*K*NNimpute is $O(m \log_2 m)$ + $O(m^2n)$ + $O(m^2n)$ which is $\approx$ $O(m^2n)$.

3.4.2 Limitations of Primary Framework

Although the proposed distance can select the expression value-wise closer and most similar positively co-expressed genes, we observe that it cannot work properly for all types of negatively co-expressed genes. It cannot handle scaling pattern-based genes. This is elaborated in Figure 3.3 and corresponding Table 3.3.

In Figure 3.3, gene1 is the target gene and all other genes are candidate genes. Among these genes, gene2 and gene3 are negatively co-expressed and shifting pattern based similar. On the other hand, gene4 and gene7 are positively co-expressed and shifted pattern-based similar, while gene5 is positively co-expressed and scaled pattern with respect to the target gene, and gene6 is magnitude wise closer to the target gene. If *K* is set at 4, then according to PEH distance gene5, gene7, gene3, gene2 will be selected as neighbor genes. Among these genes, gene7 is positively co-expressed and magnitude wise closer to the target gene.gene2 is selected by PEH distance as -gene2 which is also positively co-expressed and magnitude wise close to the target gene. gene5 is scaled but selected. gene3 is selected according to PEH distance which is negatively co-expressed and shifted. So, using PEH distance, selected neighbor genes may be scaled and may be negatively co-expressed. This hampers the weighted average procedure.

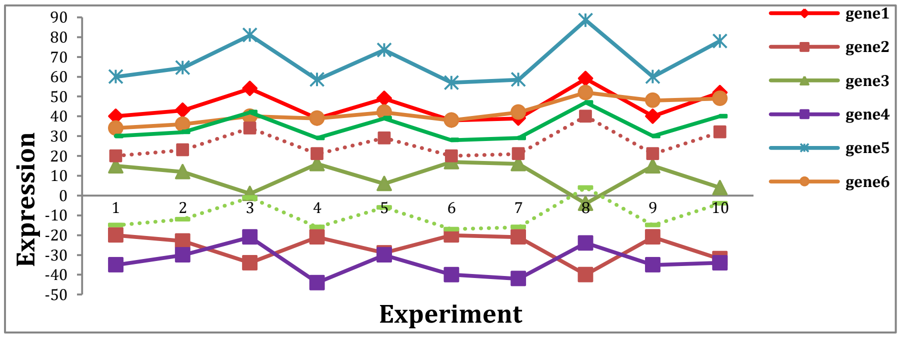


Figure3.3. Microarray Gene Expression Patterns showing drawbacks of PEH distance

Table 3.3: Pearson correlation Coefficient and PEH score for the genes in Figure3 with respect to target gene

	1	2	3	4	5	6	7	8	9	10	Pearson	PEH
gene1	40	43	54	39	49	38	39	59	40	52		
gene2	-20	-23	-34	-21	-29	-20	-21	-40	-21	-32	-0.993809	0.001174243
gene3	15	12	1	16	6	17	16	-4	15	4	-0.999192	0.000280454
gene4	-35	-30	-21	-44	-30	-40	-42	-24	-35	-34	0.833268	0.127087915
gene5	60	64.5	81	58.5	73.5	57	58.5	88.5	60	78	1	0
gene6	34	36	40	39	42	38	42	52	48	49	0.593699	0.02586895
gene7	30	32	42	29	39	28	29	47	30	40	0.99751	0.00025658
-gene2	20	23	34	21	29	20	21	40	21	32	0.993809	
-gene3	-15	-12	-1	-16	-6	-17	-16	4	-15	-4	0.999192	

3.5 A Robust Framework: A Novel Neighborhood Configuration to Improve the *K*NN based Imputation Algorithms on Microarray Gene Expression Data

To overcome the above-mentioned problems, here a robust framework has been proposed by integrating a hybrid distance and gene transformation procedure for more accurate neighborhood formation in the traditional *K*NNimpute and its different versions.

The framework consists of two parts: (1) Proposed hybrid distance-based neighbor gene selection (2) Neighbor gene transformation using the proposed transformation procedure. The proposed hybrid distance is developed based on two concepts: the Pearson correlation coefficient [34], and the Mean squared residue score [111, 112] while in the proposed transformation procedure Euclidean distance [34] is used.

The motivation behind this work is that the neighborhood for each target gene will be constructed in such a way that the weighted average procedure will run only on those genes which are simultaneously closer magnitude wise, as well as most similar with respect to the positively co-expressed pattern. To achieve this goal, first the most positively or negatively correlated/co-expressed and shifting pattern-based genes are selected using a proposed hybrid distance. Then the negatively co-expressed genes are transformed into the positively co-expressed genes. After that positively co-expressed genes are converted into magnitude wise closer genes while the scaling pattern-based positively or negatively co-expressed genes are discarded.

Already the impact of Euclidean distance and Pearson correlation coefficient in neighborhood formation has been described in section 3.3.1 and 3.3.2. In the following subsections, first, the Mean squared residue (MSR) score [111, 112] and then how it is used in this paper and its impact on neighborhood formation are described, and then the proposed framework is described in the *K*NN environment.

3.5.1 Mean Squared Residue Score and its Significance

Cheng and Church had proposed a measure known as mean squared residue score [111, 112] for measuring compact relationship among rows and also among columns in a bicluster [111, 112] in the biclustering framework. Biclustering [111] is a special kind of clustering technique in which objects/rows (genes) and features/columns (samples) are clustered at the same time. A bicluster $B_{I\times J}$ in a gene expression data matrix $G_{m\times n}$ is a sub-matrix showing similar behavior under a subset of genes ($|I|$ number of genes) and a subset of samples ($|J|$ number of samples). The residue score $R\left(b_{ij}\right)$ of ${b_{ij}}^{th}$entry of the bicluster$B$ is defined as shown in the equation (3.13):

$$R\left(b_{ij}\right) = b_{ij} - b_{iJ} - b_{Ij} + b_{IJ} \qquad (3.13)$$

where b_{ij} is the expression value of the i^{th} row (gene) and j^{th} column (sample) posi-tion in the bicluster$B$, $b_{iJ} = \frac{1}{|J|}\sum_{j\epsilon J}^{|J|} b_{ij}$, $b_{Ij} = \frac{1}{|I|}\sum_{i\epsilon I}^{|I|} b_{ij}$, $b_{IJ} = \frac{1}{|I||J|}\sum_{i\epsilon I, j\epsilon J} b_{ij}$, are i^{th} row mean, j^{th} column mean, and over all mean of the bicluster B respectively.

The mean squared residue score (M) of the bicluster B is defined as in the equation (3.14):

$$M(I,J) = \frac{1}{|I||J|}\sum_{i\in I, j\in J} R(b_{ij})^2 \qquad (3.14)$$

It has been observed in [45] that the mean squared residue score has no significant impact on positively co-expressed and shifting pattern-based genes in a bicluster but it has a great impact on positively co-expressed and scaling pattern-based genes. In the case of negatively co-expressed and shifting/scaling pattern-based genes in a bicluster, the mean squared residue score has also significant impacts.

Considering this observation, in this paper, we have used mean squared residue score $MSR(g_i, g_j)$ as a score between two genes g_i and g_j instead of a bicluster. To calculate this score, here we consider these two genes form a bicluster of size $A_{2\times n}$.The first row of this bicluster is the gene g_i and the second row is g_j. The score is defined below:

$$MSR(g_i, g_j) = \frac{1}{2\times n}\left(\sum_{k=1}^{n} R(g_{ik})^2 + \sum_{k=1}^{n} R(g_{jk})^2\right) \quad (3.15)$$

Here, g_{ik} is the k^{th} expression value of g_i and g_{jk} is the k^{th} expression value of g_j. The impact of this score on different gene expression patterns is demonstrated in Figure 3.4. In Figure 3.4, target_gene represents the target gene and all other genes are the candidate genes. If neighbor genes are selected according to MSR score and K is set at 4 then gen-e1, gene2, gene3, and gene 4 will be selected. The expressions and corresponding Pear-son correlation coefficient and MSR score are also given in Table 3.4. This is elaborated below.

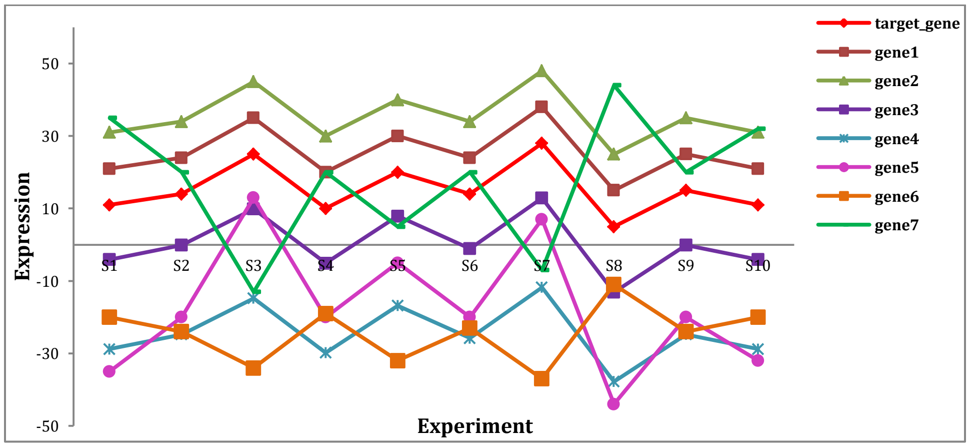


Figure 3.4: Patterns showing the effect of MSR score on scaling and shifting patterns

Table 3.4: Pearson correlation Coefficient and MSR score for the genes in figure4 with respect to target gene

	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**Pearson**	**MSR**
target_gene	11	14	25	10	20	14	28	5	15	11		
gene1	21	24	35	20	30	24	38	15	25	21	1	0
gene2	31	34	45	30	40	34	48	25	35	31	1	0
gene3	-4.1	-0.1	9.9	-5.1	7.9	-1.1	12.9	-13.1	-0.1	-4.1	0.986772	0.473
gene4	-28.8	-24.8	-14.8	-29.8	-16.8	-25.8	-11.8	-37.8	-24.8	-28.8	0.986772	0.473
gene5	-35	-20	13	-20	-5	-20	7	-44	-20	-32	0.952899	29.722
gene6	-20	-24	-34	-19	-32	-23	-37	-11	-24	-20	-0.98677	50.052
gene7	35	20	-13	20	5	20	-7	44	20	32	-0.9529	139.4

In this figure, gene1 and gene2 are perfectly positively co-expressed and display shifting patterns (with different shifting factor) with respect to the target gene. The MSR score between gene1 and target_gene is 0. The MSR score between gene2 and target_gene is also 0. So, for perfectly positively co-expressed genes, the value of the MSR score is 0 and is independent of the shifting factor. gene3 and gene4 are also positively co-expressed and show shifted (with different shifting factor) patterns with respect to the target gene. In both cases, the residue score is 0.473 and it is independent of the shifting factor. So, for positively co-expressed genes, the residue score decreases with the increase of pattern-based similarity and is independent of the shifting factor. On the other hand, gene5 is positively co-expressed and displays a highly scaled pattern with respect to the target gene. The MSR score between gene5 and the target_gene is very high. So, the MSR score between two positively co-expressed and scaling pattern-based genes is very high.

gene6 is negatively co-expressed and displays a shifting pattern. The MSR score of gene6 with respect to the target gene is very high. The case of gene6 therefore bears evidence of the fact that consideration of only MSR score may also lead to misleading decisions. Another candidate gene, gene7 is also negatively correlated and highly scaled with respect to the target gene. The MSR score between gene7 and the target gene is also very high. From these examples, it is clear that if a low MSR score is used for neighbor selection then it may favor the most similar positively co-expressed and shifting pattern genes, though there remains a problem in such cases as gene6 which is addressed in the subsequent sections.

3.5.2 Proposed Framework in *K*NN environment (M*K*NNimpute)

Here, a framework has been proposed for better neighborhood formation in traditional *K*NNimpute and it is renamed as M*K*NNimpute. The framework is developed by combining the Pearson correlation coefficient [34], mean squared residue (MSR) score [111, 112], and Euclidean distance [34]. It has two parts: (1) Proposed hybrid distance-based neighbor gene selection (2) Neighbor gene transformation using the proposed transformation procedure.

In M*K*NNimpute, the imputation procedure starts from the target gene with the least number of missing entries. Then according to our novel framework, a set (S) of K neighbor genes of the target gene is formed depending on the proposed distance. If $K \geq \alpha$, then these neighbor genes are transformed using the proposed transformation procedure. Then the missing entries in the target gene are predicted by taking the traditional weighted average of those transformed neighbor genes as in equation (2) simultaneously. If $K < \alpha$ then the α number of the nearest neighbor genes of the target gene are selected according to Euclidean distance forming a set S and then missing entries are imputed by taking their weighted average as in the equation (3.2). This process is repeated to predict missing entries in other target genes also. Here α is a user-defined threshold.

3.5.2.1 Proposed HPCMSR distance

Proposed distance for neighbor gene selection: Here a hybrid measure named Hybrid Pearson Correlation Mean Squared Residue ($HPCMSR$) distance is proposed by combining Pearson correlation (PC) coefficient and mean squared residue (MSR) score. The proposed distance is defined in equation (11):

$$HPCMSR\,(g_t, g_c) = \begin{cases} abs\left(abs(PC(g_t\,, g_c)) - NMSR(g_t, g_c)\right), if\ PC(g_t, g_c) > 0 \\ abs\left(abs(PC(g_t, g_c) - NMSR(g_t, \widetilde{g_c})\right), if\ PC(g_t, g_c) < 0 \end{cases} \quad (3.16)$$

$PC(g_t, g_c)$ represents Pearson correlation value between the gene vectors g_t (target gene) and g_c (candidate gene) and $NMSR(g_t, g_c)$ represents the normalized MSR score between them. $\widetilde{g_c}$is the inverted pattern of g_c. $NMSR(g_t, g_c)$score between two genes g_t and g_c is defined below:

$$NMSR\,(g_t, g_c) = \frac{MSR\,(g_t, g_c)}{Max_i\,(MSR\,(g_t, g_i))} \quad (3.17)$$

$Max_i\,(MSR\,(g_t, g_i))$ represents the maximum MSR score among all MSR scores of candidate genes with respect to a target gene. $HPCMSR$ value between two genes will vary between 0 and 1 which means $0 < HPCMSR \leq 1$. The higher value of $HPCMSR$ signifies that the two genes are highly pattern-based similar (positively co-expressed or negatively co-expressed and shifted). The significance of $HPCMSR$ is demonstrated in Figure 3.5 and Table 3.5.

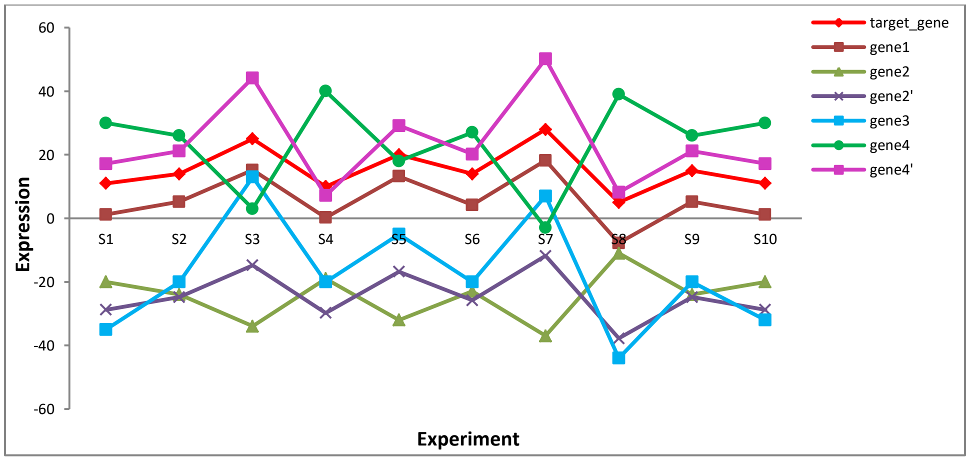


Figure 3.5.Patterns showing the effect of HPCMSR score for scaling, shifting, and inverted patterns

In Figure 3.5, target_gene is the target gene and all other genes are candidate genes. If neighbors are selected using $HPCMSR$ score and K is set at 4 then gene1, gene2, gene5, and gene6 will be selected. Among these genes, gene1, gene5, and gene6are highly positively correlated/co-expressed and shifted with respect to the target gene. So, their $HPCMSR$ value with respect to the target gene are also very high mentioned in Table 3.5. gene2 is also pattern-based similar with the target gene like gene1 but with opposite correlation. To calculate $HPCMSR$ of gene2, this gene is inverted with respect to itself by an inverting method (discussed later) and it becomes gene2'. The $HPCMSR$ value of gene2'with respect to the target gene is also very high mentioned in Table 3.5. So, from this example, it is clear that if two genes are highly positively co-expressed or negatively co-expressed and shifted, then $HPCMSR$ score between them is high.

Table 3.5. Score values of the genes in Figure3.5 with respect to target gene

in figure5	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**Pearson**	**MSR**	**NMSR**	**HPCMSR**
target_gene	11	14	25	10	20	14	28	5	15	11				
gene1	1.2	5	15.2	0	13	4	18	-7.8	5.2	1.2	0.986772	0.473	0.004763	0.982009
gene2	-20	-24	-34	-19	-32	-23	-37	-11	-24	-20	-0.986772	50.052	0.504036	0.482736
gene2'	-29	-25	-15	-30	-17	-26	-12	-38	-25	-29	0.986772	0.473	0.004763	0.982009
gene3	-35	-20	13	-20	-5	-20	7	-44	-20	-32	0.952889	29.722	0.299308	0.653591
gene4	30	26	3	40	18	27	-3	39	26	30	-0.974477	99.3025	1	0.025523
gene4'	17.2	21	44.2	7	29	20	50	8.2	21.2	17.2	0.974477	12.0225	0.121069	0.853408

On the other hand, gene3 is highly positively co-expressed and scaled with respect to the target gene with $HPCMSR$ value of 0.653591 while gene4 is highly negatively co-ex-pressed and scaled with respect to the target gene. This gene is inverted to gene4' and its$HPCMSR$ value is 0.853408. So, for the highly positively co-expressed/negatively co-expressed and scaled genes $HPCMSR$ value is small compared to the highly positively or negatively correlated shifting patterns. This is due to the reason that using the Pearson correlation coefficient it is possible to distinguish positively co-expressed and negatively co-expressed genes while it cannot identify shifting and scaling patterns. But MSR score has a significant impact on scaling and also negatively co-expressed patterns.

So, a combination of these two metrics in $HPCMSR$ score in such a manner gives this score the capability that it can select most positively/negatively co-expressed and shifted pattern-based genes and discard scaling pattern-based genes, if such genes exist in the dataset.

Now in this work, for every target gene, $HPCMSR$ distance is calculated between every candidate gene and the target gene, and then neighbor genes are selected forming a set (S) whose $HPCMSR$ value is greater than a certain threshold δ.

If δ is set at high value then only positively co-expressed/negatively co-expressed and shifted pattern-based candidate genes will be selected as neighbors according to $HPCMSR$ value and the scaled pattern-based candidate genes will be discarded.

3.5.2.2 Proposed transformation method

After the selection of neighbor genes, every neighbor gene is transformed using the following proposed transformation procedure. If any neighbor gene is negatively correlated with the target gene then this gene is inverted with respect to itself by using an inverting method (discussed later) and it becomes positively co-expressed with the target gene. Then all of these positively co-expressed genes are shifted toward the target gene by the shifting method (discussed later) so that they will come closer to the target gene. Finally, the weighted average of those trans-formed neighbor genes is taken to predict missing values in the target gene, where weights are calculated using Euclidean distance.

Table 3.6. Expression and Score value of genes in Figure6 with respect to the target gene

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson	MSR	HPCMSR
target_gene	11	14	25	10	20	14	28	5	15	11			
gene1	1.2	5.2	15.2	0.2	13.2	4.2	18.2	-7.8	5.2	1.2	0.986772	0.473	0.982009
gene1"	10.9	14.9	24.9	9.9	22.9	13.9	27.9	1.9	14.9	10.9	0.986772	0.473	0.982009
gene2	-20	-24	-34	-19	-32	-23	-37	-11	-24	-20	-0.986772	50.052	0.013228
gene2'	-28.8	-24.8	-14.8	-29.8	-16.8	-25.8	-11.8	-37.8	-24.8	-28.8	0.986772	0.473	0.977322
gene2"	10.9	14.9	24.9	9.9	22.9	13.9	27.9	1.9	14.9	10.9	0.986772	0.472	0.977342
gene3	48	39	33	46	38	41	32	54	41	52	-0.925749	45.923	0.008243
gene3'	36.8	45.8	51.8	38.8	46.8	43.8	52.8	30.8	43.8	32.8	0.925749	1.802	0.889746
gene3"	9.7	18.7	24.7	11.7	19.7	16.7	25.7	3.7	16.7	5.7	0.925749	1.803	0.889726
gene4	-35	-20	13	-20	-5	-20	7	-44	-20	-32	0.952899	29.722	0.359077
gene5	56	59	65	56	56	59	65	48	58	56	0.88275	2.912	0.824571
gene5"	13.5	16.5	22.5	13.5	13.5	16.5	22.5	5.5	15.5	13.5	0.88275	2.912	0.824571
gene6	25	27	35	26	32	26	33	21	34	29	0.836457	3.813	0.760276

This novel transformation procedure is described using Figure 3.6 and Table 3.6. In Figure 3.6(a), except for the target gene, all other genes are candidate genes. For this example, δ is set at 0.95, and K is set at 5. According to the above criteria, K number of nearest neighbor genes are selected forming the neighbor set $S = \{$gene1, gene2, gene3, gene4, gene5$\}$. Now every member of the set S is processed one after another. At first, gene1 is considered. For gene1, as it is positively co-expressed, only shifting operation is performed on it to make it (gene1”) closer to the target gene as shown in Figure 3.6(b).

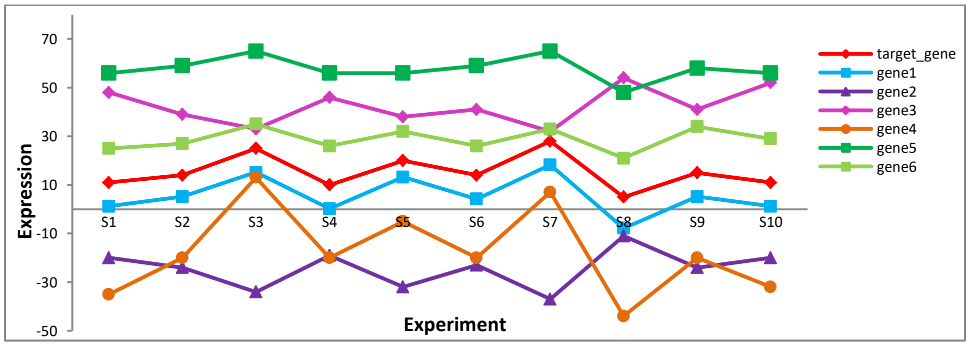


Figure 3.6(a)

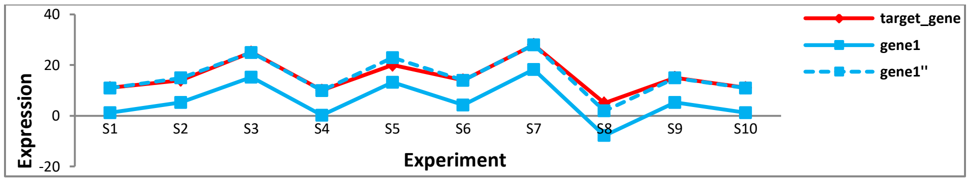


Figure 3.6(b)

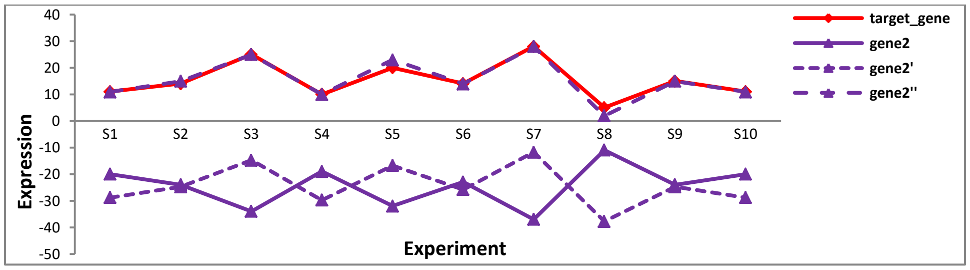


Figure 3.6(c)

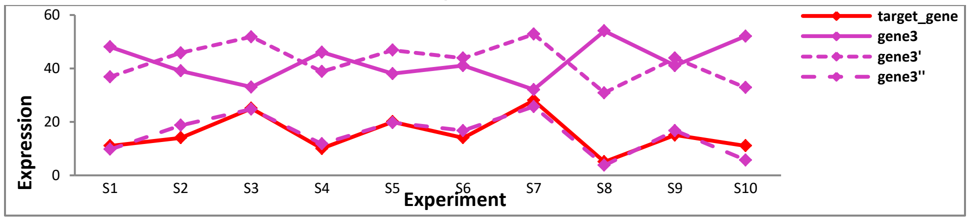


Figure 3.6(d)

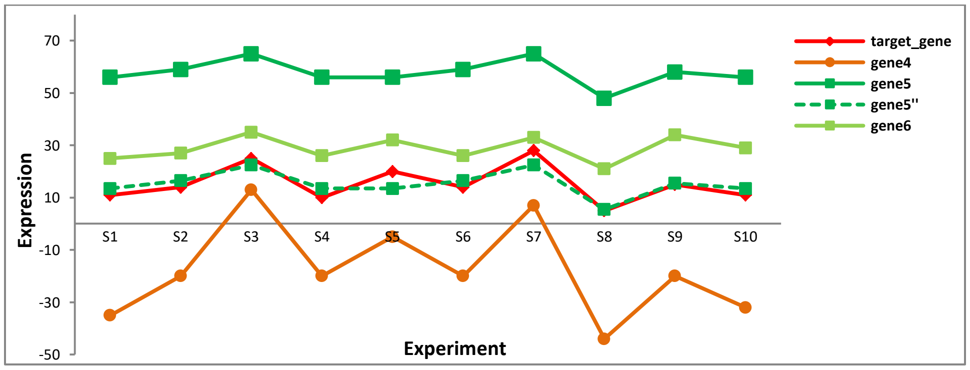


Figure 3.6(e)

Figure 3.6 (a-e). Effect of proposed transformation on different pattern-based genes

For gene2, as it is negatively co-expressed, it is first inverted with respect to itself by the inverting method (discussed below) so it will become (gene2') positively co-expressed with respect to the target gene. Then gene2' is shifted to the target gene by the shifting method (discussed below) so that it (gene2") will come closer to the target gene as shown in Figure 3.6(c). Similarly, for gene3 the same operation is carried out as it is also negatively co-expressed and shown in Figure 3.6(d). For gene3, as it is positively co-expressed only shifting operation is performed on it to make it (gene3") closer to the target gene. A similar operation is carried out for gene4 and gene5 as they are also positively co-expressed as shown in Figure3. 6(e). Finally, weight is calculated for each of these transformed neighbor genes according to the equation (3.3) and missing values in the target gene are predicted according to the equation (3.2). The flowchart for the imputation procedure using the proposed framework is shown in Figure 3.7. The algorithm is shown in Figure 3.8.

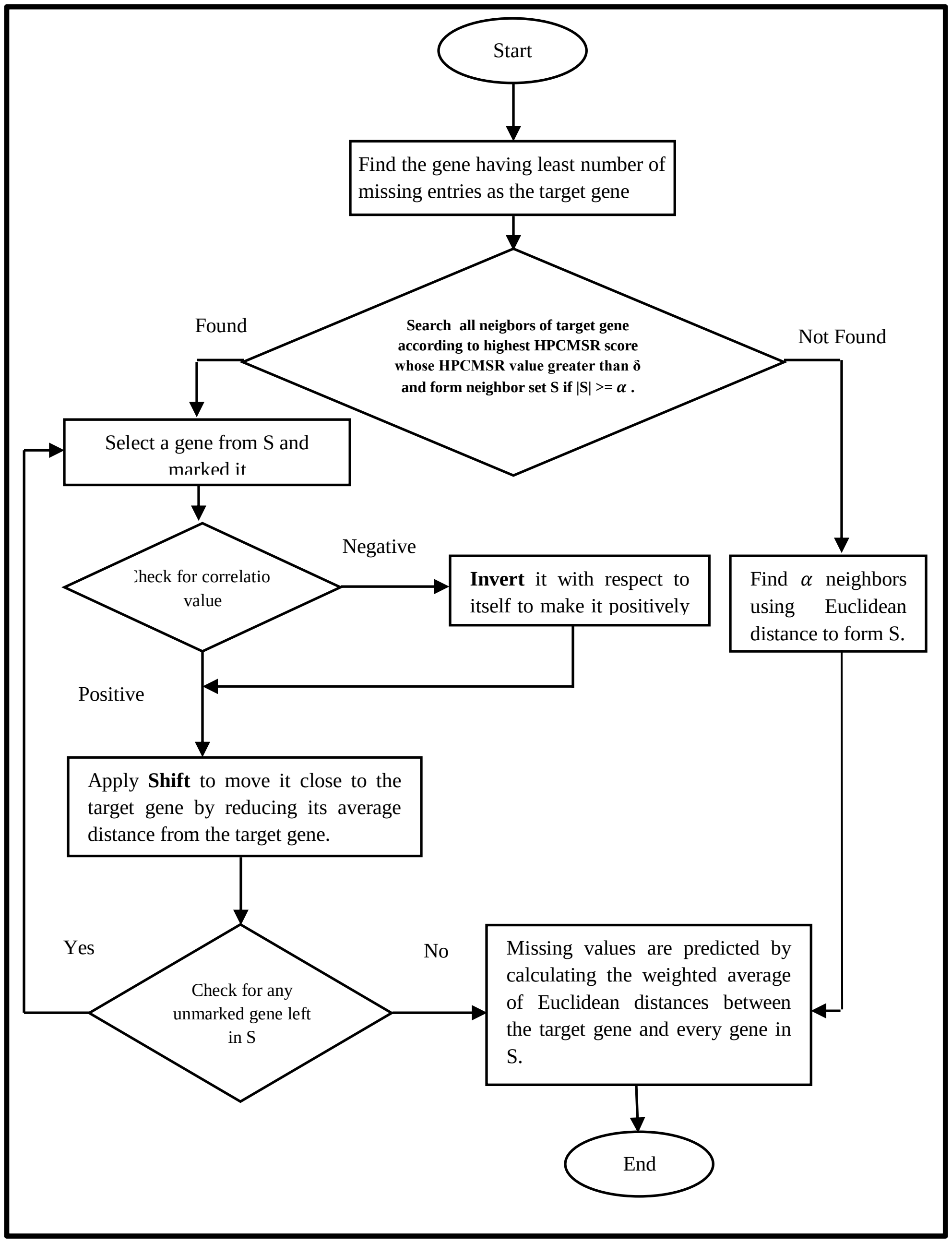

Start
Find the gene having least number of missing entries as the target gene
Search all neigbors of target gene according to highest HPCMSR score whose HPCMSR value greater than δ and form neighbor set S if |S| >= α .
Found
Not Found
Select a gene from S and marked it
Check for correlatio value
Negative
Invert it with respect to itself to make it positively
Find α neighbors using Euclidean distance to form S.
Positive
Apply Shift to move it close to the target gene by reducing its average distance from the target gene.
Yes
No
Check for any unmarked gene left in S
Missing values are predicted by calculating the weighted average of Euclidean distances between the target gene and every gene in S.
End

Figure 3.7: Flowchart of the proposed Modified *K*NNimpute

3.5.2.3 Proposed inverting and shifting methods

Method for inverting a gene with respect to a target gene:

If the Pearson correlation coefficient value of a gene (gene2, for instance, in Figure 6(a)) with respect to the target gene is less than 0 then that gene is negatively co-expressed with respect to the target gene. To invert this gene (here gene2), the average expression value of this gene with respect to all experimental conditions has been taken (here it is -24.4) and accordingly, x= -24.4 is considered as an axis. Now the expression value of this gene for each experimental condition is subtracted from -24.4 and the result is added to -24.4 to get the inverted expression value for that gene (here gene2') as shown in Figure 3.6(c). The Invert function is shown in Figure 3.8(b).

Method for shifting a gene with respect to a target gene:

To shift a gene (consider gene1 in Figure 3.6(a)) with respect to a target gene, first the average expression value for the target gene and also for gene1 with respect to all experimental conditions have been taken. Then the difference of these average expression values has been calculated. This difference value is then again subtracted from the expression value of each experimental condition of that gene (here gene1) to shift that gene (here gene1") shown in Figure 3.6(b). The Shift function is shown in Figure 3.8(c).

```
Algorithm: Modified K-Nearest Neighbor Imputation (MKNNimpute)
Input: A gene expression matrix $G_{m \times n}$ with $m$ genes (rows), $n$ samples/conditions/experiments (columns), and $m \gg n$. $p$ number of missing entries are artificially created in $G$ and these $p$ number of missing entries are present in $q$ number of target genes and $q \leq m$. $S$ is a set of $K$ number of gene vectors that means $K = |S|$. $M$ and $L$ are two $n$-dimensional arrays. $\boldsymbol{Invert}(\boldsymbol{Y})$ is a procedure which will invert a gene vector $Y$. $\boldsymbol{Shift}(\boldsymbol{W}, \boldsymbol{Z})$ is another procedure that will shift a gene vector $Z$ to its corresponding target gene vector $W$. $\alpha$ and $\delta$ are the user-defined parameters.
Output: All missing entries are imputed in $G$.
1. Arrange all $q$ number of target genes in $G$ considering in ascending order with respect to their missing rates.
2. Mark all $q$ number of target genes as unprocessed.
3. For each target gene $g_t$ in $G$ do the following:
    i. Calculate $HPCMSR$ value between $g_t$ and each other gene present in $G$ and choose all the nearest neighbor genes forming a set $S$ of $g_t$ whose $HPCMSR$ value is greater than $\delta$.
    ii. If $K \geq \alpha$ then
        A) For every member gene $g_{tneighbour_i}$ of $S$ do the following:
            a) If $PR(g_{tneighbour_i}, g_t) < 0$ then
                    $M$ = Call Procedure $\boldsymbol{Invert}(g_{tneighbour_i})$.
               Else
                    $M = g_{tneighbour_i}$
            b) $L$ = Call Procedure $\boldsymbol{Shift}$ $(g_t, M)$
            c) Calculate weight of $L$ using Euclidean distance-based weight as shown in the equation (3).
            d) Calculate all the missing entries in $g_t$ simultaneously by taking the weighted average of expression values of its modified (inverted and shifted) $K$-nearest neighbor genes according to the equation (2).
        Else
        A) Select $\alpha$-nearest neighbor genes of $g_t$ from the gene expression matrix $G$ based on Euclidean distance forming a set $S$.
        B) Calculate weight for each member gene $g_{tneighbour_i}$ as shown in the equation (3).
        C) Calculate all the missing entries in $g_t$ simultaneously by taking the weighted average of its $\alpha$-nearest neighbor genes as shown in equation (2).
    iii. Mark $g_t$ (current target gene) as processed.
4. Impute all predicted missing entries calculated in step3 in the gene expression matrix $G$.
5. End
```

Figure 3.8 a) The proposed algorithm

```
Procedure: Invert ($\boldsymbol{Y}$)
1. Calculate average expression value(μ) of $Y$ for all $n$ number of experimental samples.
2. Subtract expression value for each experimental condition of gene vector $Y$from μ and add the difference value for each experimental condition result with μ to get the expression value for each experimental condition of inverted $Y$ named $\ddot{Y}$.
3. Return $\ddot{Y}$.
```

Figure 3.8 b) The invert function

```
Procedure: Shift (W, Z)
1. Calculate average expression value ($\alpha$) of $W$ for all $n$ number of experimental samples.
2. Calculate average expression value ($\beta$) of $Z$ for all $n$ number of experimental samples.
3. Subtract $\beta$ from $\alpha$ and store this difference value in $\eta$.
4. Subtract $\eta$ from expression value of each experimental condition of $Z$ to get shifted $Z$ named as $\widehat{Z}$.
5. Return $\hat{Z}$.
```

Figure 3.8c) The shift function

Figure 3.8 (a-c): The proposed algorithm and its different versions

3.5.3 Proposed Framework for Sequential *K*-Nearest Neighbor Imputation (MS*K*NN-impute)

In modified sequential K-nearest neighbor imputation, the proposed imputation procedure starts from the target gene which has minimum entries of missing. Similar to the M*K*NNimpute, the neighbor set S is formed from the candidate genes, whose $HPCMSR$ value with respect to the target gene is greater than a certain threshold δ. Let $K = |S|$. If $K \geq \alpha$, then like M*K*NNimpute, member genes of S are transformed if needed and the weighted average of those genes is taken for imputation. If $K < \alpha$, then Euclidean distance is used to select α number of genes for the formation of the set S and the weighted average of those genes is taken for imputation. The original gene expression matrix is then replaced by the predicted value. This process continued in the subsequent estimation of other missing entries in the current target gene and other target genes. The only difference between M*K*NNimpute and MS*K*NNimpute method is that after predict-ting each missing entry, this value has been used for later imputation.

3.5.4 Proposed Framework for Iterative *K*-Nearest Neighbor Imputation (MI*K*NNimpute)

In MI*K*NNimpute, the missing positions in G are initially replaced by the corres-ponding row (gene) averages to form a complete gene expression matrix $G_{complete}(0)$. For each target gene, it applies the framework for neighborhood formation and simul-taneously imputes all the missing positions of that target gene using the weighted average computed according to the equation (2). After imputing all the missing positions in the data matrix, again the complete gene expression matrix $G_{complete}(1)$ is formed. This process is iterated until the difference between the estimated values of two successive iterations reaches a given threshold. In $G_{complete}(h)$, h indicates the iteration number. The estimated values obtained in the last two successive iterations are used to calculate their sum of squared differences according to equation (3.4).

3.5.5 Proposed Framework for Iterative Sequential *K*-Nearest Neighbor Imputation (MIS*K*NNimpute)

In modified iterative sequential K-nearest neighbor imputation, the proposed impu-tation procedure starts from the target gene which has minimum entries of missing. Similar to the M*K*NNimpute, the neighbor set S is formed from the candidate genes, whose $HPCMSR$ value with respect to the target gene is greater than a certain threshold δ. Let $K = |S|$. If $K \geq \alpha$, then like M*K*NNimpute, member genes of S are transformed if needed and the weighted average of those genes is taken for imputation. If $K < \alpha$, then Euclidean distance is used to select α number of genes for the formation of the set S and the weighted average of those genes is taken for imputation. In this way, the missing positions in G are predicted to form a complete gene expression matrix $G_{complete}(0)$. This process is applied again to form the complete gene expression matrix $G_{complete}(1)$. This process is iterated until the difference between the estimated values of two successive iterations reaches a given threshold. In $G_{complete}(h)$, h indicates the iteration number. The estimated values obtained in the last two successive iterations are used to calculate their sum of squared differences according to equation (3.4).

3.6 Experimental Results

The effectiveness of the proposed algorithm is certified by carrying out a large number of experiments over ten microarray gene expression datasets. For comparing the efficiency of the proposed methods, different versions of the proposed methods are compared with the well-known weighted average based and numerical methods based on existing missing value estimation techniques. The accuracy of the proposed methods in comparison with the above-mentioned existing techniques has been ensured using the following metrics:(a) normalized root mean squared error (NRMSE) [92] and (b) average distance between partitions error (ADBPE) [113].

3.6.1 Gene Expression Datasets

In this paper, ten different microarray gene expression data sets are taken (as listed in Table 3.7), considering different conditions of datasets like missing structure (equally and unequally distributed missing entries), missing rates (1 to 20% and uneven), species type (Saccharomyces cerevisiae, Atlantic salmon, and Homo sapiens) and type of dataset (time series, non-time series and mixed). Datasets SP.AFA [114] and SP.ELU [114] are of the type time series, GAS [115], ROS [116], GOL [117], Tymchuk [118], and HIR [119] are of non-time series while BALD [120], YOS [121] are of mixed type microarray gene expression datasets. Apart from the above datasets, a synthetic dataset generated by SynTReN [122] has also been used for our experimentation. The entropy values for the above datasets are calculated according to [123]. The lower and higher values of entropy signify the strong and weak correlation among the gene expressions respectively [123], and depending on that phenomena datasets are considered as global structured and local structured based datasets respectively.

Table 3.7: Description of microarray gene expression datasets used for experimentation

Dataset	Full Dim.	Used Dim.	Category	Organism	Expression Profile	Entropy
Spellman Alpha(SP.AFA)	7681×18	4304×18	Time-series, cyclic	s.cerevisiae	"Cell-cycle genes"	0.94
SpellmanElu (SP.ELU)	7681×14	4304×14	Time-series, cyclic	s.cerevisiae	"Cell-cycle genes"	0.909
Tymchuk	5299×34	1617×34	Steady state	Atlantic Salmon	Conservation genomics	0.922
Gasch(GAS)	6152×174	5629×25	Steady state	s.cerevisiae	Cellular response to DNA-damaging agent	0.923
Ross(ROS)	9706×60	2266×60	Steady-state	Homo Sapiens	NCI 60 cancer cell lines	0.944
Yoshimoto (YOS)	6166×24	4380×24	Mixed type	s.cerevisiae	Response to changes in calceneurine-dependent	0.826
Golub (GOL)	7129×72	7070×72	Steady-state	Homo Sapiens	Acute lymphoblastic leukemia	0.876
Baldwin (BALD)	16814×39	6838×39	Time-series, non-cyclic	Homo Sapiens	Epithelial cellular response to L monocytogenes	0.819
Synthetic	200×50	200×50	Steady state	s.cerevisiae	Regulatory interactions	0.91

3.6.2 Missing data set-up

Here, all the datasets are used in the form of a $\log_2$ transformed scale. For this reason, these datasets are first converted to $\log_2$ scale if these are already not available in $\log_2$ transformed scale. The complete gene expression dataset is formed by removing those genes which contained any missing entry.

The initial $\log_2$ transformed dataset is represented as $G_{iniitial}$ with u genes and n samples. The complete dataset $G_{complete}$ has been formed excluding any single occur-ence of missing entries having m genes and n samples. Finally, missing entries are randomly introduced in $G_{complete}$ to generate the test dataset G using procedures A and B as in [92].

A. **Missing at random (Uniform):** In this procedure, a random function is used to randomly identify the possible entries or locations to be artificially deleted to create the missing entries. The numbers of such identified locations are 1%, 5%, 10%, 15%, and 20% of the total number of locations of the dataset to create the test dataset G.
B. **Missing at random (Non-uniform):** During microarray experiments missing entries are created due to several experimental failures. These missing entries are not always uniformly distributed across the dataset. The image captured during the experiment may lead to having very low-intensity values for several columns or several rows leading to the non-uniform distribution of missing entries. The study says that 0.8% to 10.6% of locations are found missing in a dataset in the form of these non-uniform distributions. To mimic such a natural phenomenon, this procedure creates missing entries by selecting rows or columns randomly and a maximum of 50% of entries for that selected row or

column deleted continuously starting from any random location. In this way, this procedure creates non-uniform test datasets having 1.5% to 4.5% missing entries.

To get unbiased results, the procedures in [117] of creating test data matrix are repeated multiple (50, in this case) times for each of the dataset with each different percentage of missing entries. Nine different datasets are used in this study and in each of these datasets six different percentages of missing positions are created by generating 2700 number of test data sets.
A notation has been used for recognizing a particular test dataset with a particular percentage of missing entries for all 50 versions of repeated occurrences. For example, for GAS data set with 1% missing entries, the notation used here is GAS/1% and for uneven percentage, it is GAS/uneq.

3.6.3 Metrics for Assessment of Performance

The performance of the proposed methods has been evaluated using two major metrics. These are (a) Normalized Root Mean Squared Error (NRMSE) [92] and (b) Average Distance Between Partitions Error (ADBPE) [102].

1) Normalized Root Mean Square Error

For evaluating the accuracy of any method, the algorithm is first applied to every dataset to predict the artificially introduced missing values. If the actual value is (y_h) and the predicted value is $(\hat{y}_h)$ then the error is calculated as normalized root mean squared error (NRMSE) as in equation (7).

$$NRMSE = \frac{1}{\sigma_y}\sqrt{\frac{\sum_{h=1}^{n}(y_h - \hat{y}_h)^2}{n}} \qquad (3.18)$$

Where σ_h represents the standard deviation of n original values at the missing positions present in the experimental matrix.

2) Average Distance Between Partitions Error

Another measure, known as average distance between partitions error [102] is used here to compare the different imputation strategies. For every dataset, each of the prediction method is applied for prediction of artificially introduced missing values in test matrix G. For this purpose, first the $G_{complete}$ matrix is partitioned (U) into c number of clusters using c-means [62] clustering algorithm. Then test matrix G is imputed with any imputation method generating matrix G_{impute} and then divided into c number of clusters (partition V) using c-means clustering. Then using Hungarian algorithm [124] the optimal match between the clusters of two partitions is determined by minimizing the average distance between them. The distance, $D(u_i, v_j)$, of a cluster, u_i, in the partitioning of the imputed data with each cluster, v_j, in the partitioning of the actual data is calculated using the formula in [124].
ADBP error is calculated as in equation (3.19).

$$D(U,V)=\frac{1}{c}\sum_{j=1}^{c}D(u_j,v_j) \qquad (3.19)$$

If two partitions are identical then the ADBP error $D(U,V)$ becomes zero and if they are totally different then it becomes one. So ADBP error varies between 0 and 1.

3.6.4 Choice of model parameters for the proposed methods

The performance of the proposed M*K*NN, MS*K*NN, MIKNN, and MIS*K*NN depends on the proper selection of model parameter δ (the threshold for HPCMSR value) and α (user-defined cut-off for a minimum number of neighbor genes). The values of model parameters δ and α for which the proposed methods give the best performance in different types of data with different missing rates cannot be specified using a theoretical approach. Here, each proposed method is executed 50 times for each dataset and each missing rate by taking different values of δ and α to empirically judge the optimal value of δ and α. It has been found that the proposed methods generate the best results when α is set at the range of 5 to 10 for local structured datasets. For global structured based data-sets, the methods give the best results when α is set at the range of 10 to 20. It has been observed that for both local structured and global structured based datasets the proposed methods give the best results when the δ value is set at the range of 0.90 to 0.95. The sensitivity analysis of parameters α and δ, for different category datasets, are given in Figures: 3.9 (a) to 3.9 (e).

The sensitivity analysis has been performed in terms of cross validation also on Synthetic dataset Figure 3.9 f) to justify the selection of optimal values of model parameters. To perform the cross-validation testing, the Synthetic dataset has been used as training data. A testing data comprising of the same set of genes (200) with a larger number of samples (75) has also been generated. It has been found that the value of parameters α and δ are in the same range as it was in the training dataset.

3.6.5 Selection of optimal values of model parameters for other imputation algorithms

Each existing algorithm for missing value estimation, considered here, is executed 50 times by taking different values of model parameters for every missing rate in each data-set to select the optimal value of its model parameters empirically. The results are measured in terms of NRMSE. From experimental results, it has been observed that for *K*NN and its several versions, the best results are found when *K* is set between 10 and 20 and is consistent with the previous results [123]. For SVDimpute, the range of eigenvalues lies between 0.15 and 0.25 [123]. LLSimpute has its built-in parameter optimization characteristic [123].

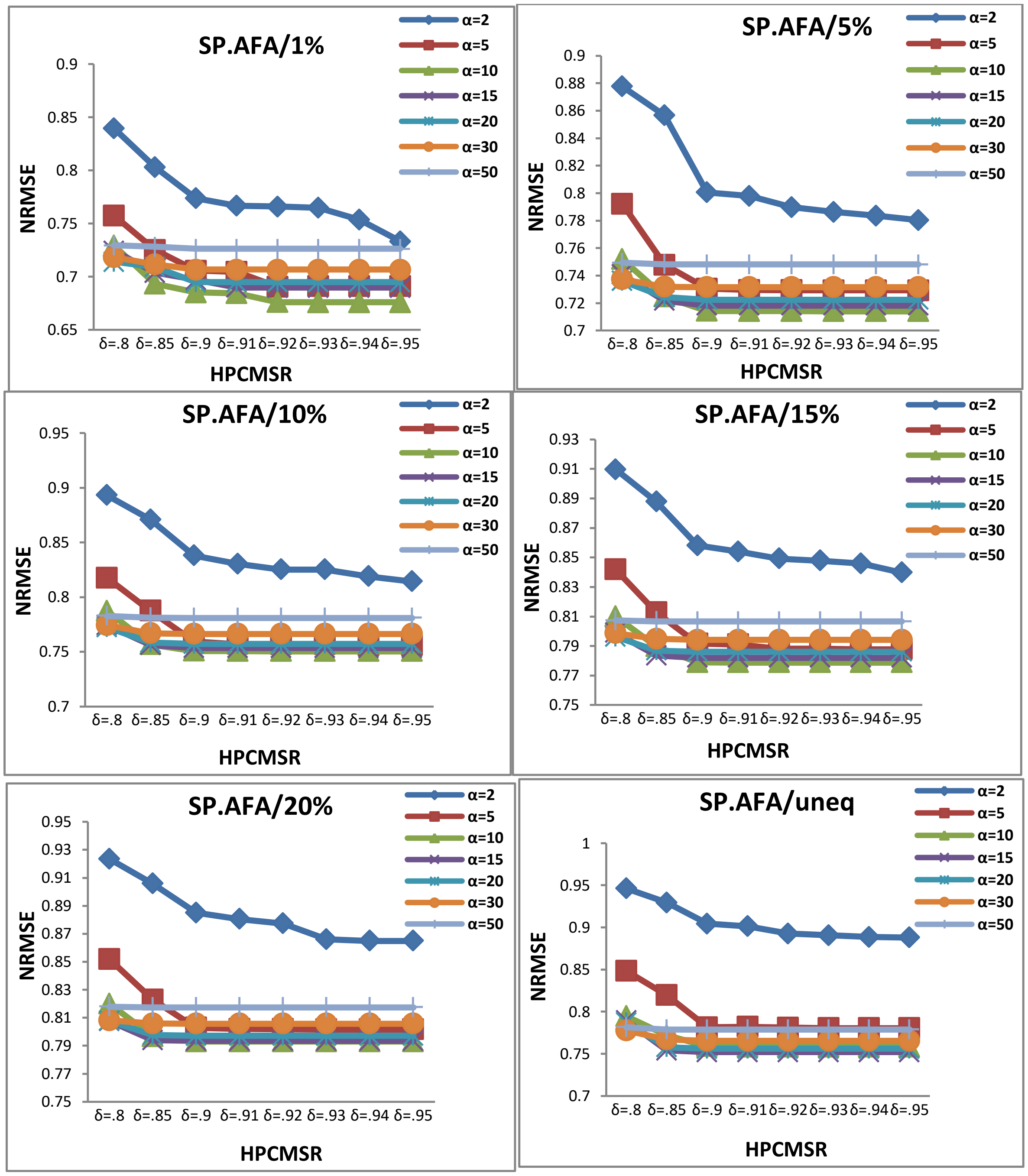


Figure 3.9a): Sensitivity graph of parameters α and δ for SP.AFA dataset

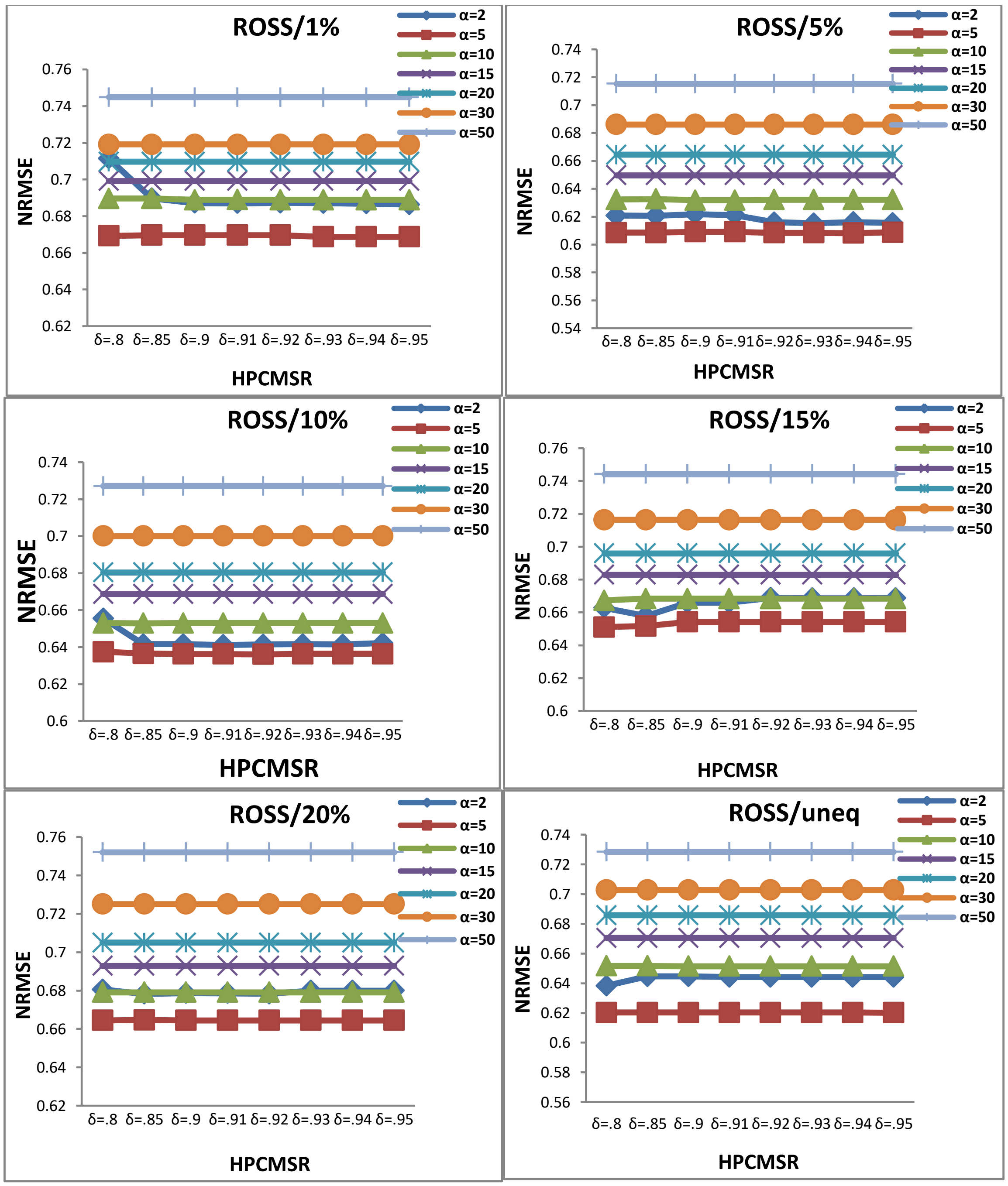


Figure 3.9 b) Sensitivity graph of parameters α and δ for ROS dataset

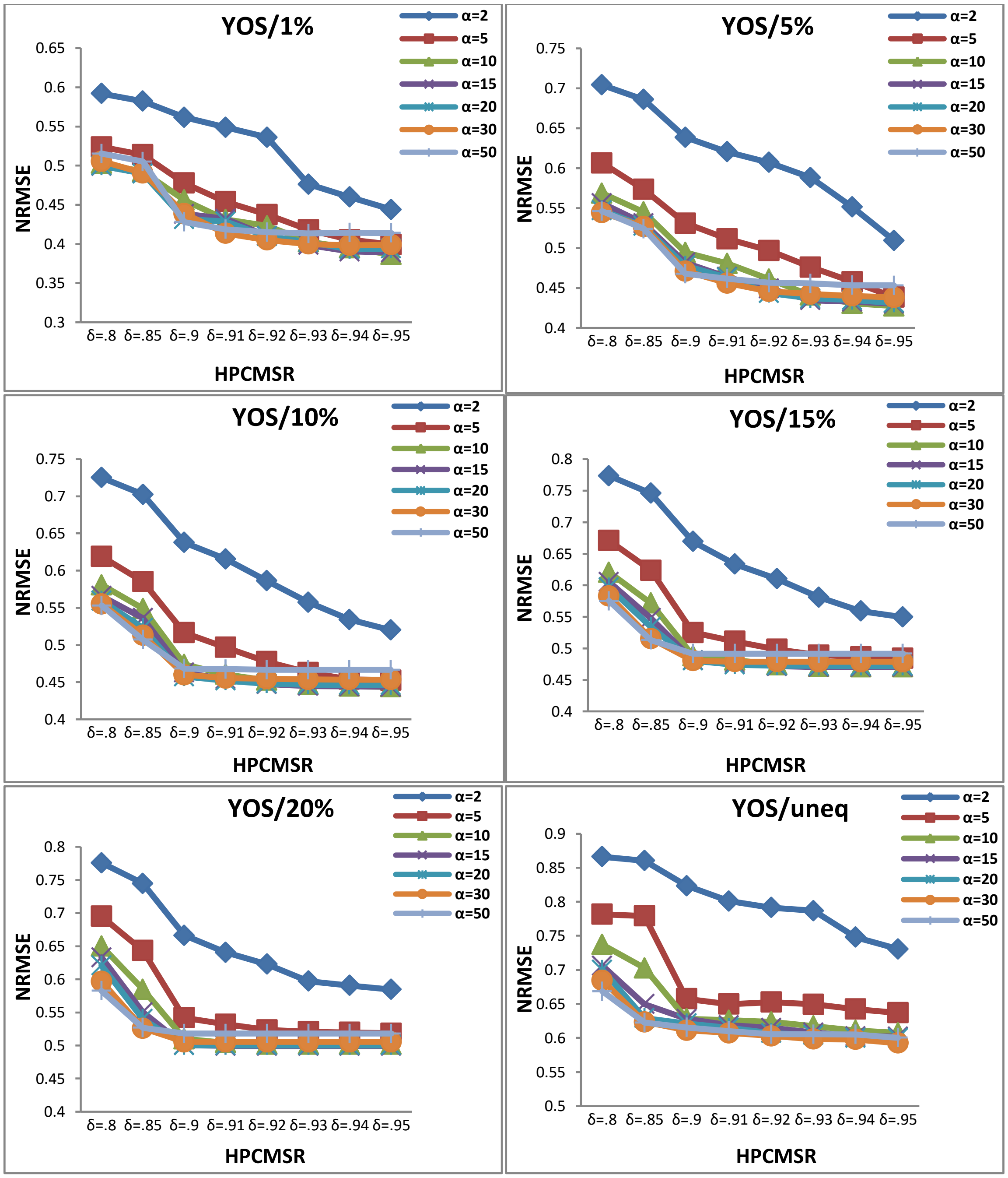


Figure 3.9c) Sensitivity graph of parameters α and δ for YOS dataset

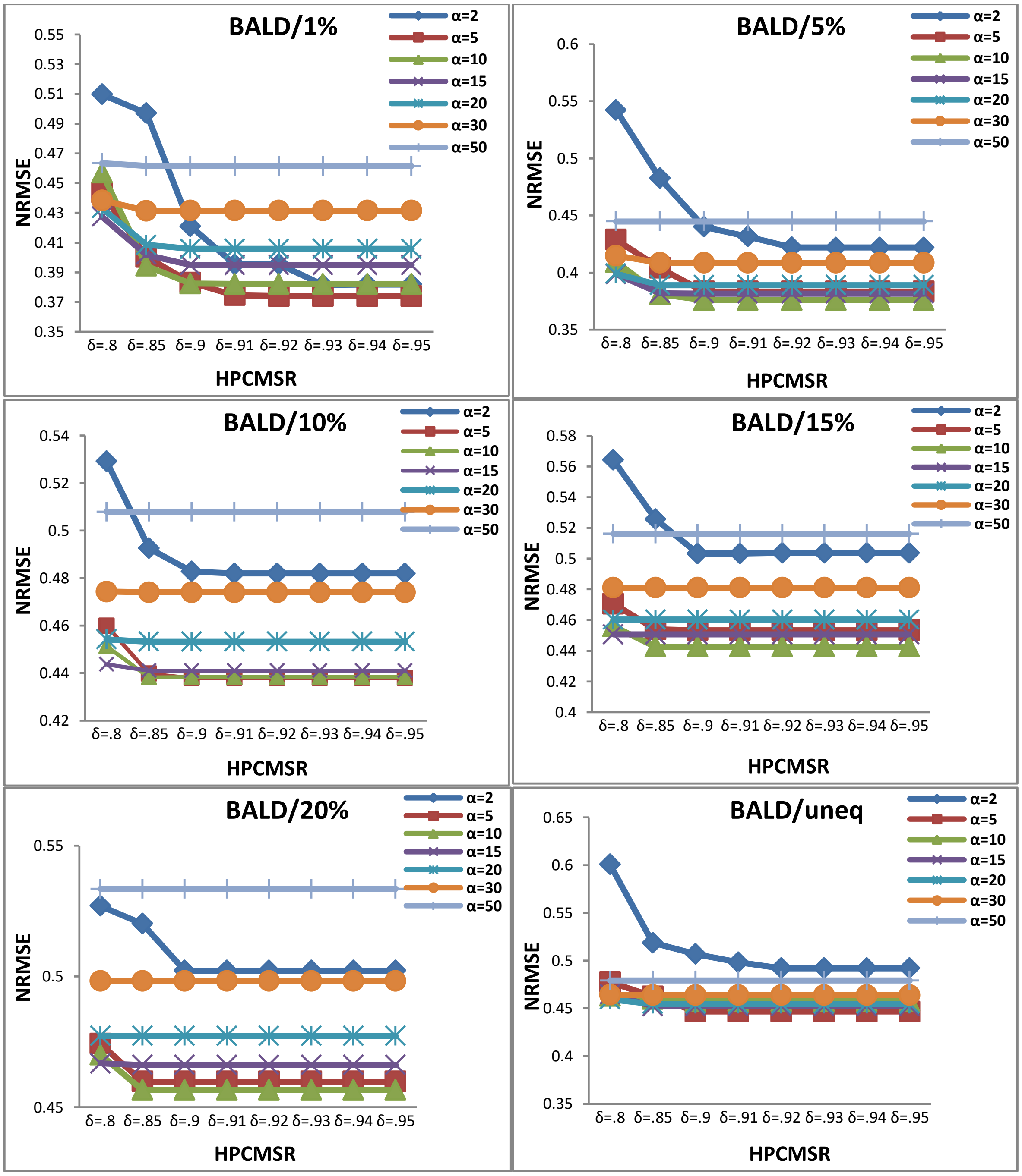


Figure 3.9d) Sensitivity graph of parameters α and δ for BALD dataset

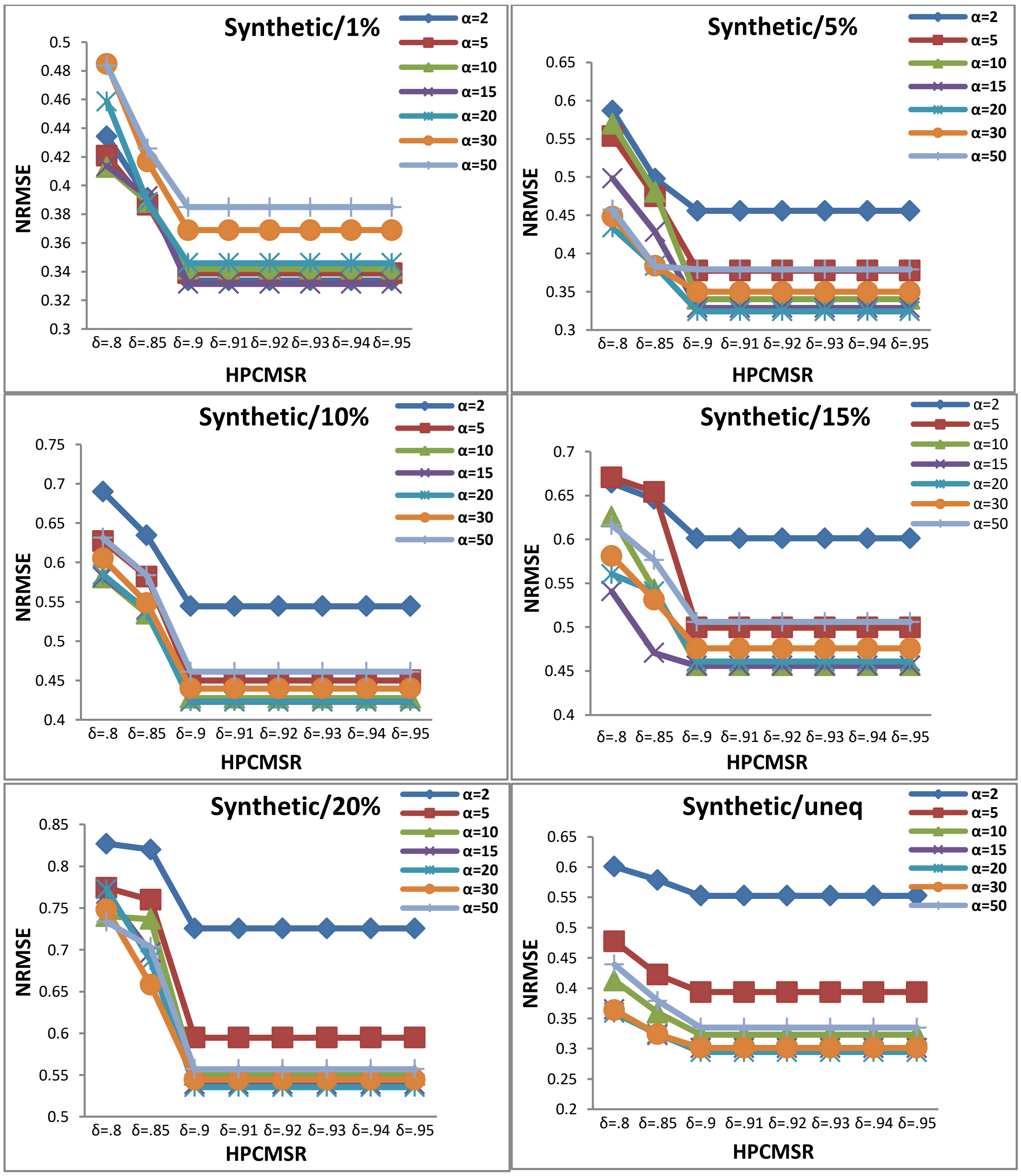


Figure 3.9 e): Sensitivity graph of parameters α and δ for Synthetic dataset

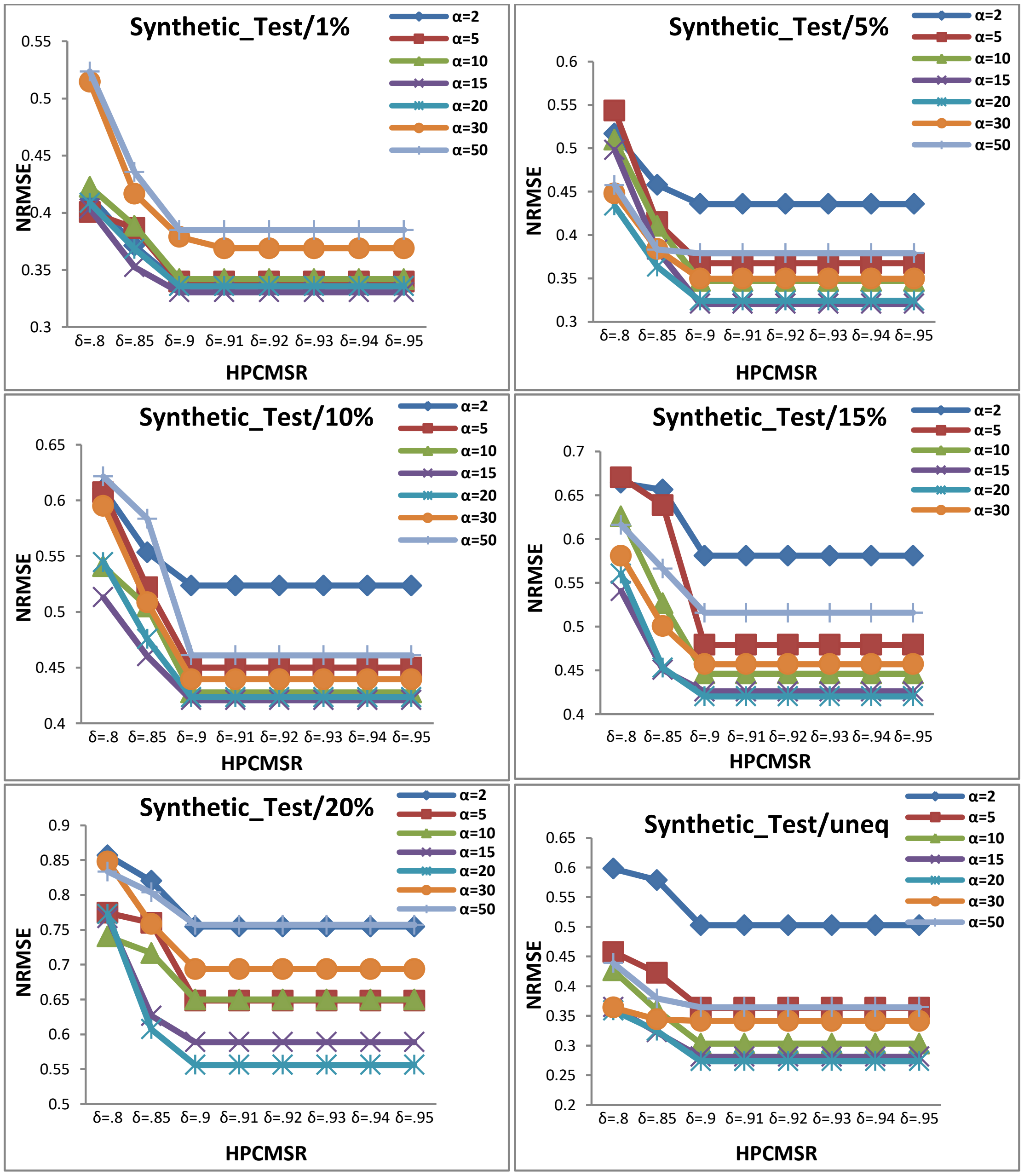


Figure 3.9f): Sensitivity graph of parameters α and δ for Synthetic dataset generated for testing.

3.6.6 Comparative Performance Analysis based on NRMS error

In Figures 3.10(a) to 3.10(j), the proposed weighted average based M*K*NNimpute, MS*K*NNimpute, MI*K*NNimpute, and MIS*K*NNimpute methods are compared with a corresponding existing weighted average based *K*NNimpute, S*K*NNimpute, I*K*NNimpute, and IS*K*NNimpute methods in terms of NRMSE to prove the effectiveness of the proposed methods. The results are shown here for all datasets.

In noisy time-series datasets SP.AFA, the different versions of the proposed algo-rithm have performed better than their corresponding existing versions for all types of missing structure and missing rates. Among them, the MIS*K*NNimpute shows the highest estimation accuracy by generating the lowest NRMS error.

In steady-state local structured based dataset Synthetic, MIS*K*NNimpute shows out-standing performance for all cases. M*K*NNimpute, MI*K*NNimpte, and MS*K*NNimpute al-so produce notably better performance than their corresponding traditional versions for all cases for this dataset.

In another local structured based dataset Tymchuk, M*K*NNimpute performs better than *K*NNimpute in a lower missing rate. Its performance degrades showing almost the same performance as *K*NNimpute at a higher missing rate. MS*K*NNimpute and MI*K*NNimpute give better performance in all cases than their corresponding existing versions. MIS*K*NNimpute shows notable performance for all missing rates.

In mixed-type global structured based dataset YOS, M*K*NNimpute, MS*K*NNimpute, and MI*K*NNimpute give notably better performance than their corresponding traditional version for all types of missing rates. MIS*K*NNimpute shows comparable performance with its other versions.

From these results, it is concluded that the use of the proposed framework in the traditional *K*NNimpute and its several versions improve their prediction accuracy significantly in most of the cases. This is true for all other datasets.

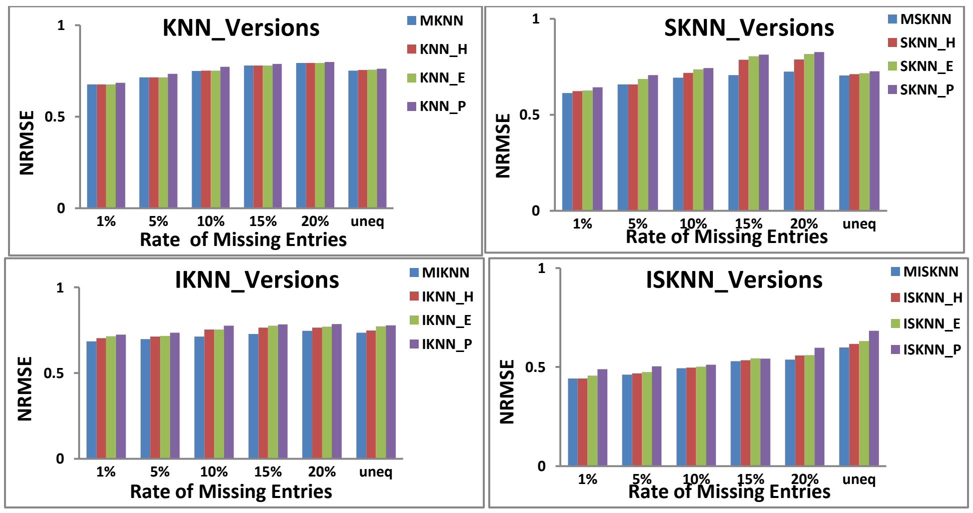


Figure 3.10(a): SP.AFA

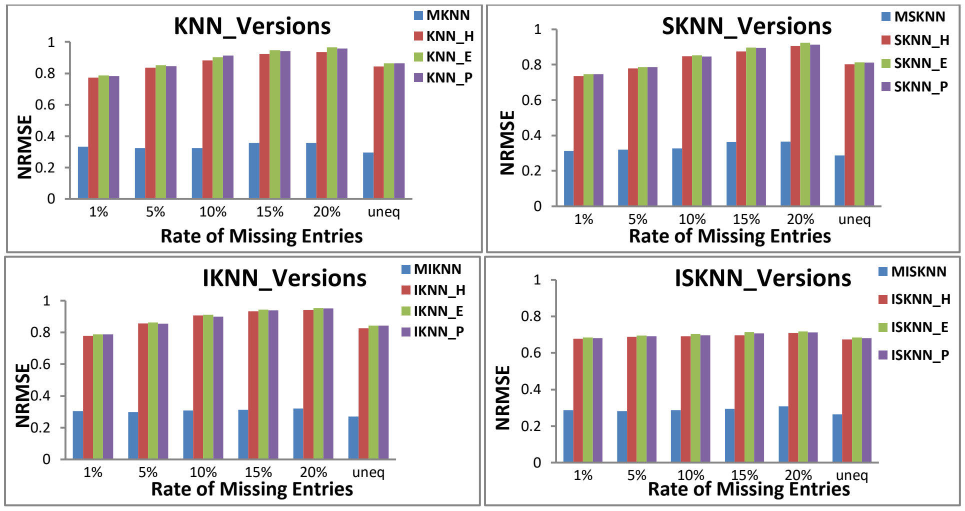


Figure 3.10(b): Synthetic

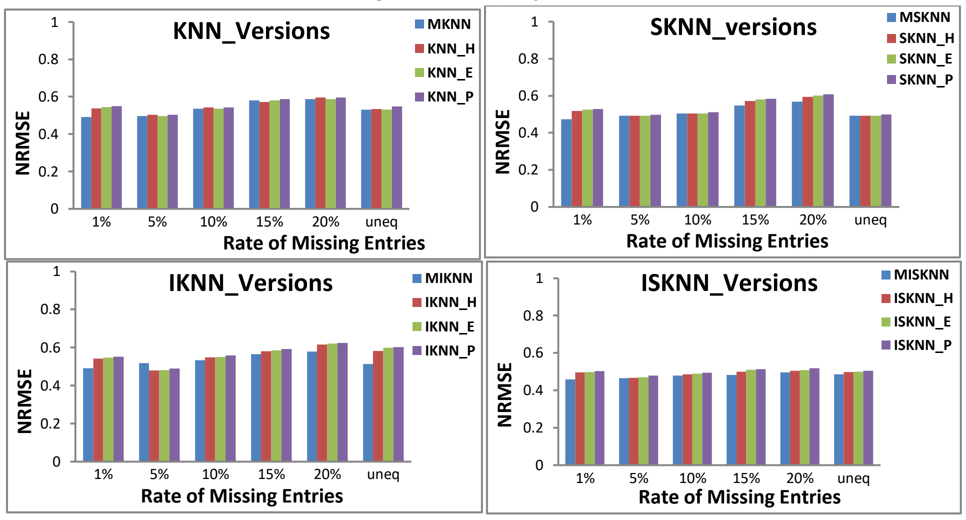


Figure. 3.10(c): Tymchuk

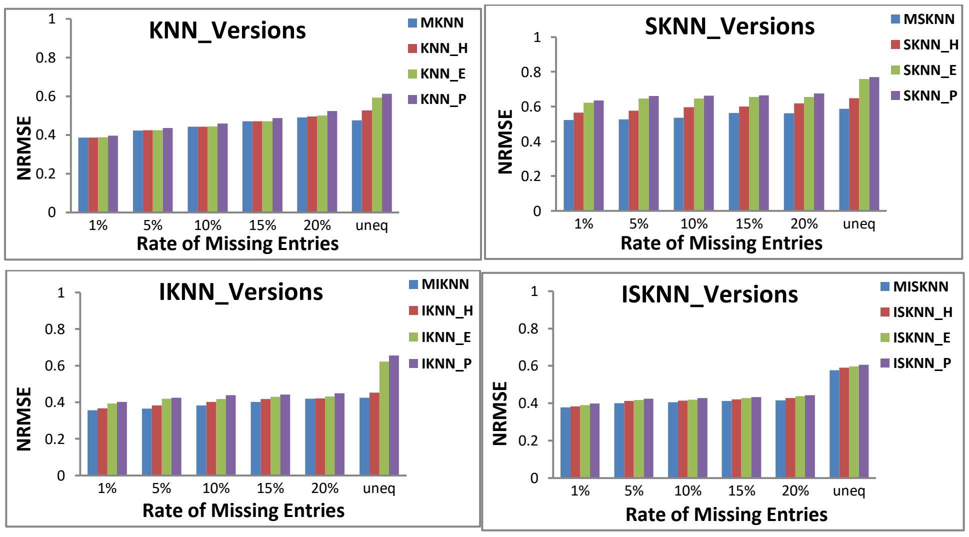


Figure 3.10(d): YOS

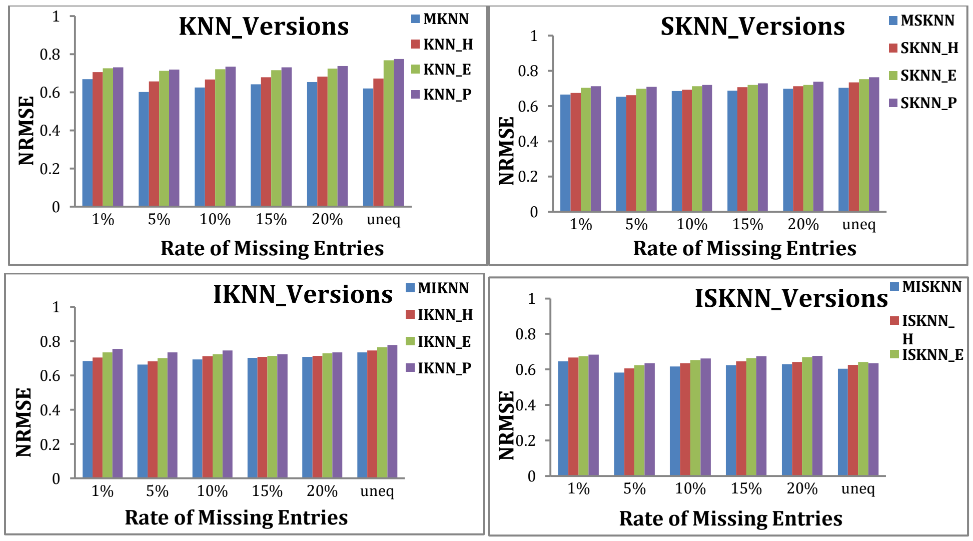


Figure 3.10(e): ROS

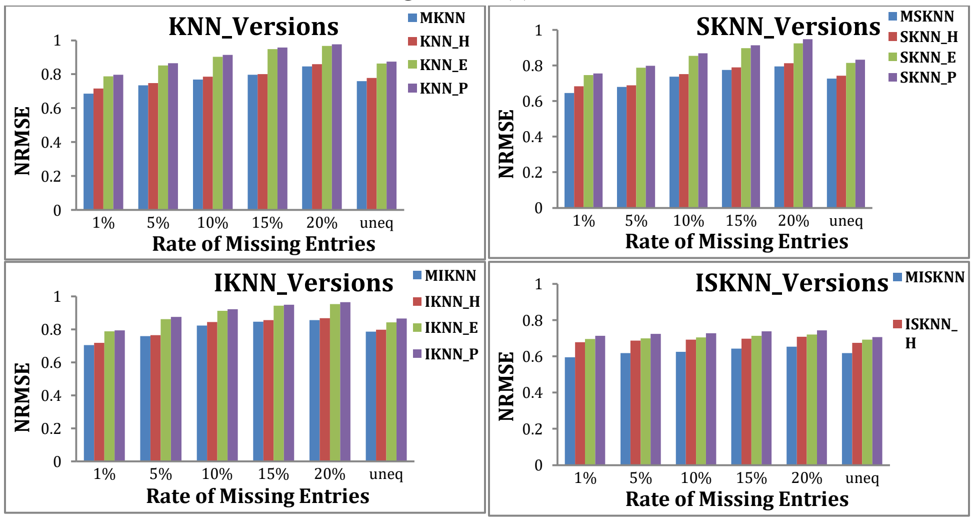


Figure 3.10(f): HIR

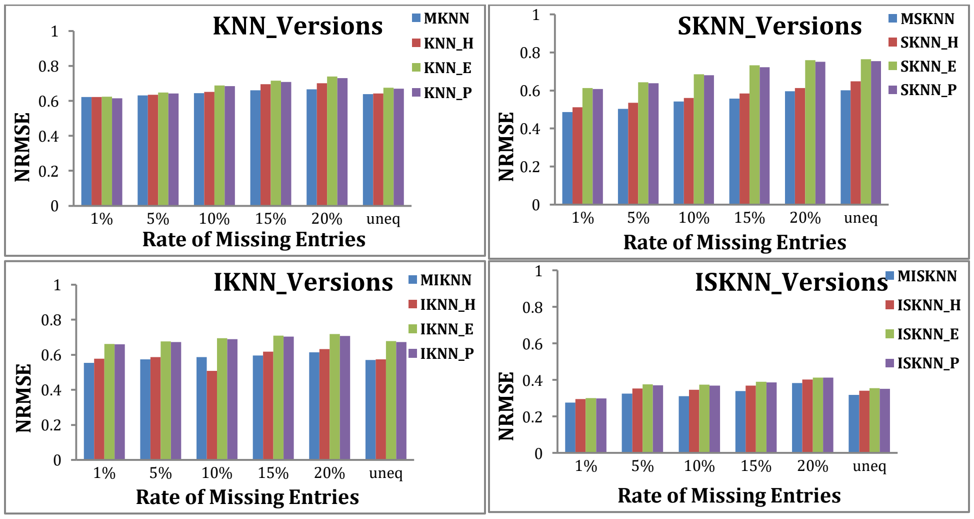


Figure 3.10(g): SP.ELU

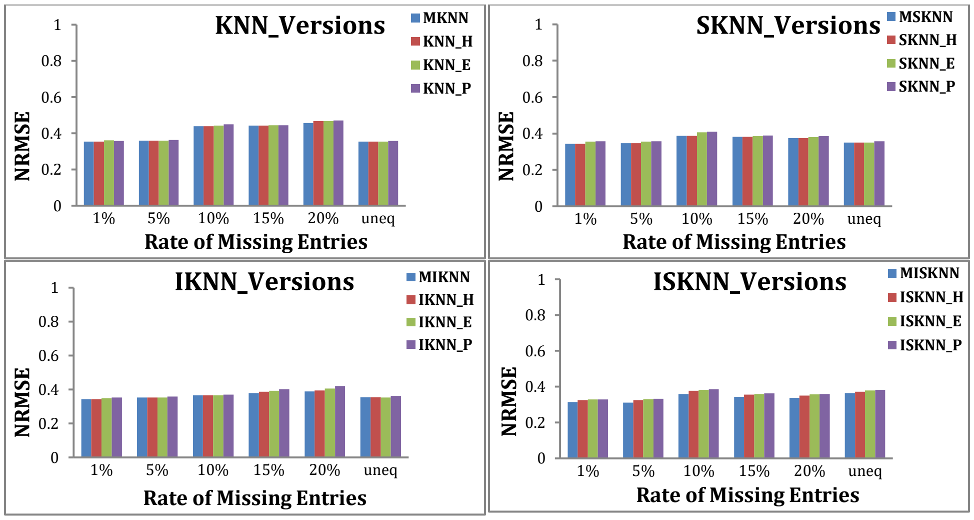


Figure 3.10(h): BALD

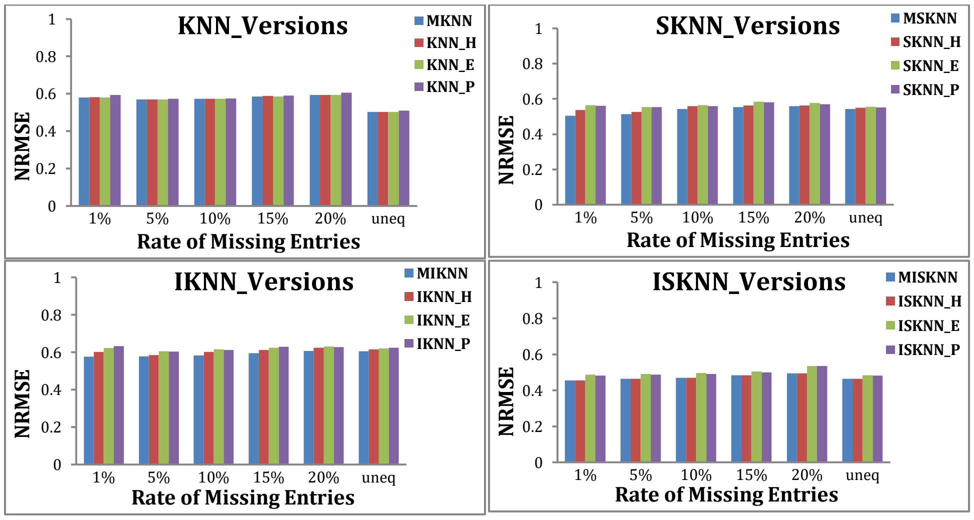


Figure 3.10(i): GAS

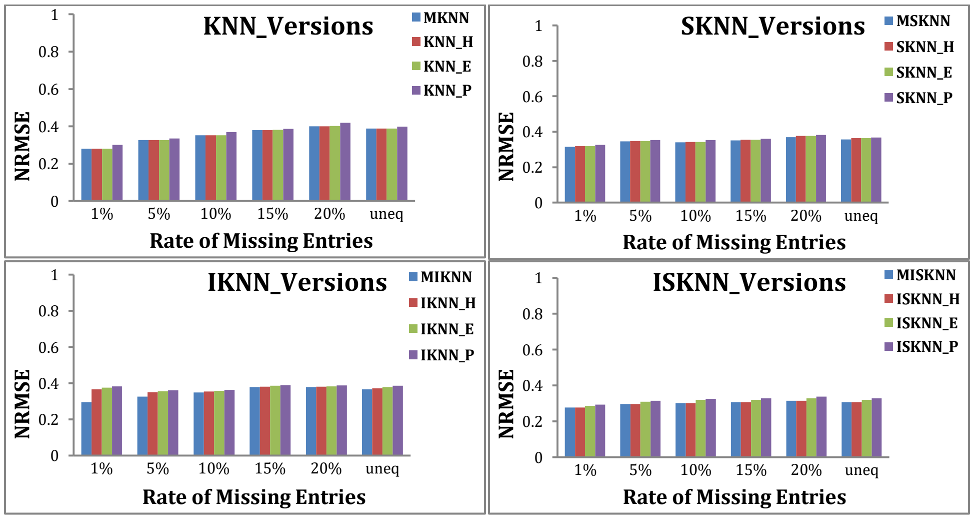


Figure 3.10(j): GOL

Figure 3.10. Comparative performance analysis of different versions of the proposed method for corresponding existing versions in terms of NRMSE for different datasets

3.6.7 Comparison of Performance for other well-known existing Imputation Techniques based on NRMS error

In Figure 3.11(a) to 3.11(j), the proposed weighted average based M*K*NNimpute, MS*K*NNimpute, MI*K*NNimpute, and MIS*K*NNimpute methods are compared with the most popular numerical methods SVDimpute[88], LLSimpute [95], BPCA[89] in terms of NRMSE by applying them on different microarray datasets to prove the effectiveness of the proposed methods. In Figure 11, results are shown for all datasets.

For SP.AFA and SP.ELU datasets, the MIS*K*NNimpute method give significantly better results than its all other versions and SVDimpute. It shows similar performance like LLSimpute and BPCA at higher missing rates. Except for MIS*K*NNimpute, all other proposed versions do not work well for these two noisy datasets.

For steady-state local structured based datasets (ROS, Tymchuk, Synthetic) MIS*K*NN-impute gives better performance compared to all its versions and other numerical methods for all types of missing rates. For these datasets, all other versions of the MIS*K*NNimpute method give better or comparable results compared to the other numerical methods. This is also true for other local structure-based datasets.

For mix type datasets (YOS, BALD) MIS*K*NNimpute is comparable with LLSimpute and BPCA at higher missing rates, but the performance of its other versions is not so good.
From these results, it can be said that for different types of datasets, among the different versions of the proposed methods, MIS*K*NNimpute is the strongest and comparable with the robust numerical methods like LLSimpute and BPCA. For steady-state local structure-based datasets, the performance of MIS*K*NNimpute and its other versions is significantly better than the numerical methods. For noisy time-series datasets and mixed datasets, MIS*K*NNimpute is comparable with other numerical methods.

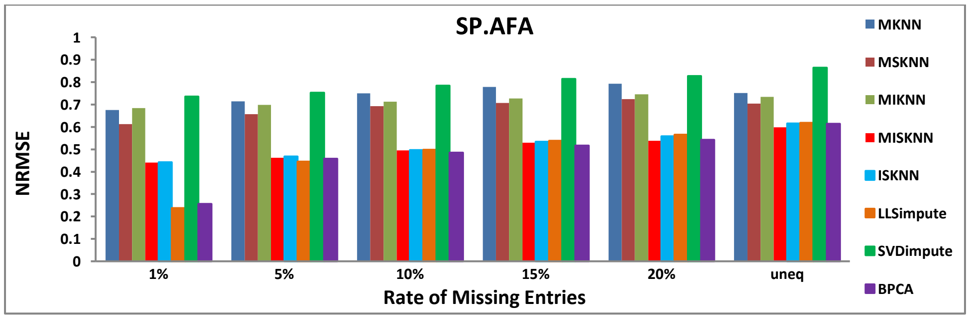


Figure 3.11(a)

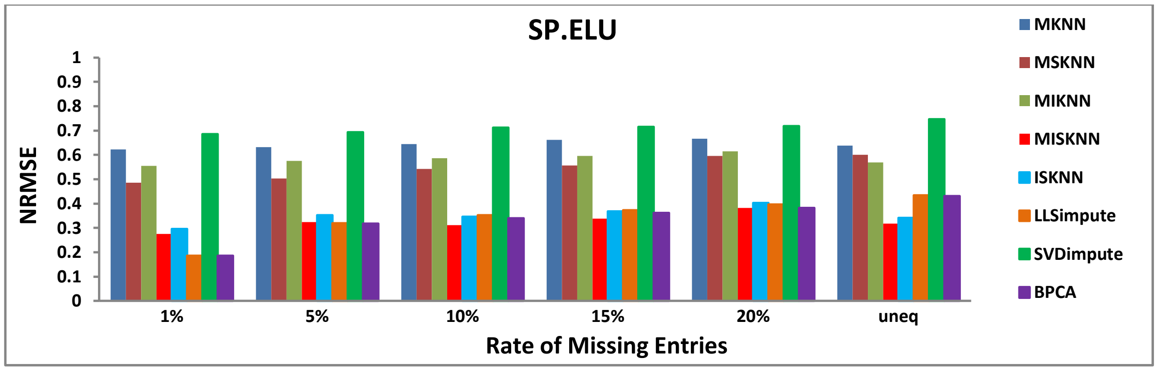


Figure 3.11(b)

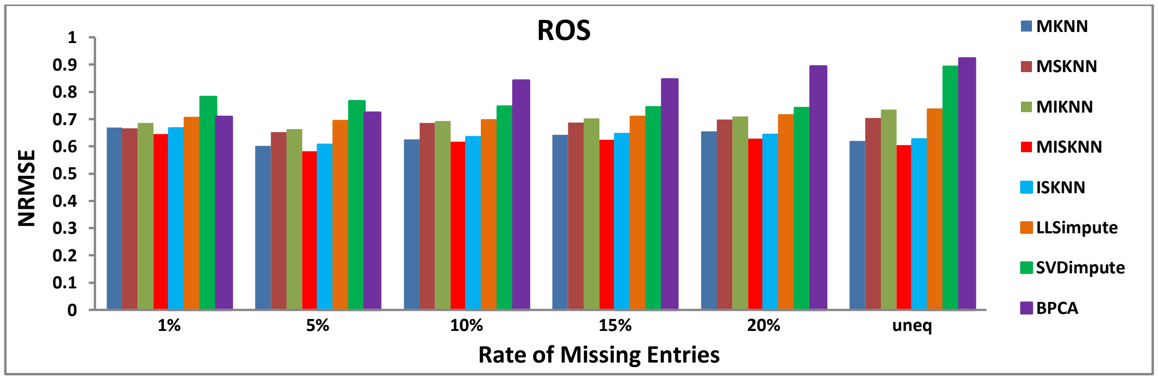


Figure 3.11(c)

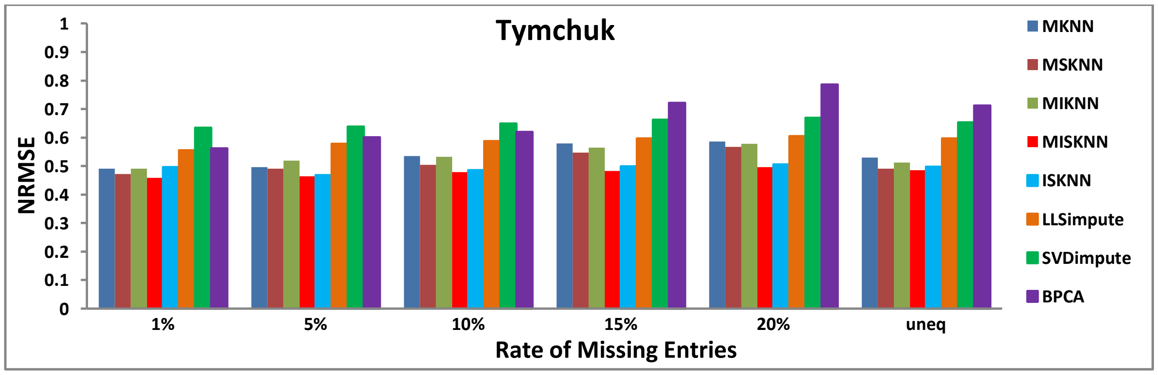


Figure 3.11(d)

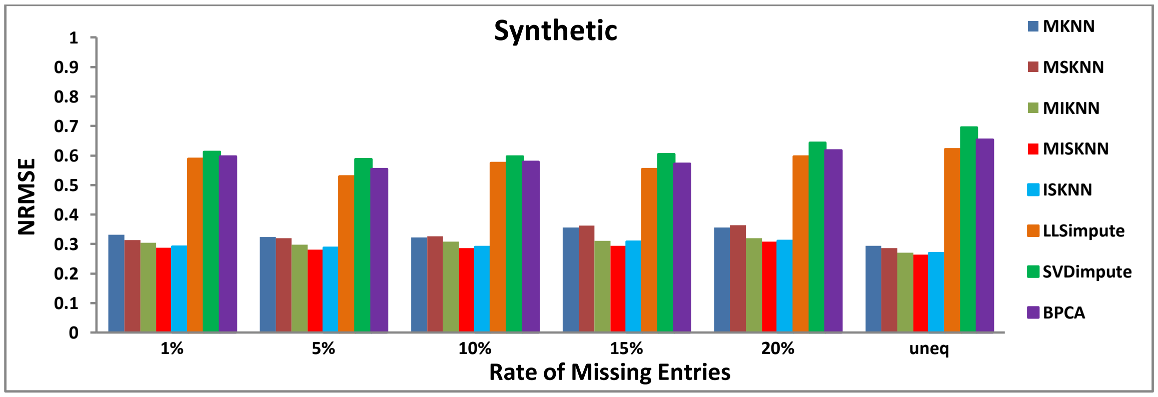


Figure 3.11(e)

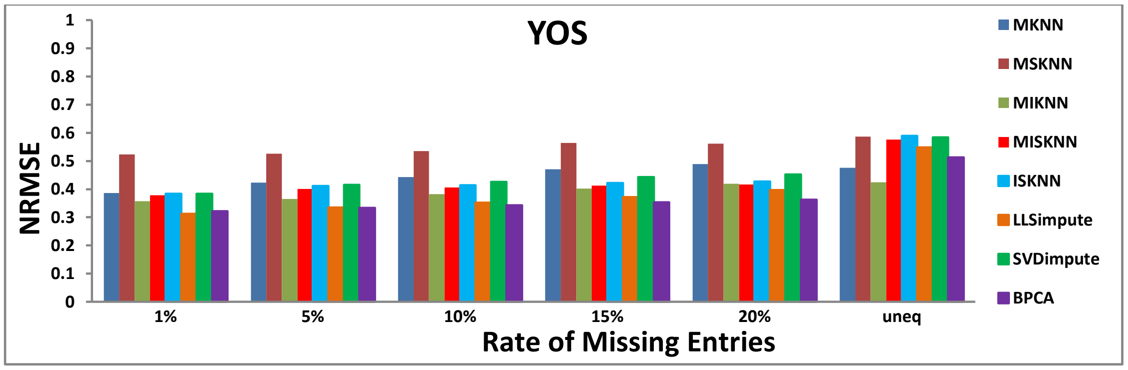


Figure 3.11(f)

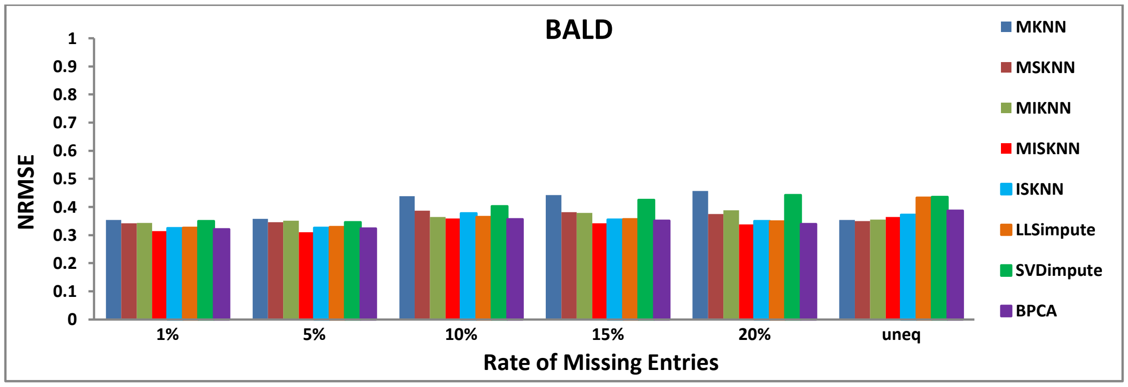


Figure 3.11(g)

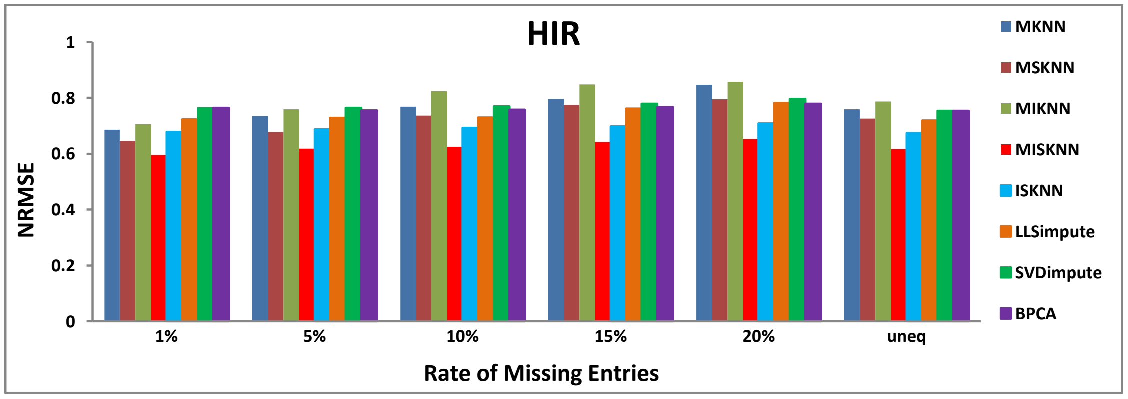


Figure 3.11(h)

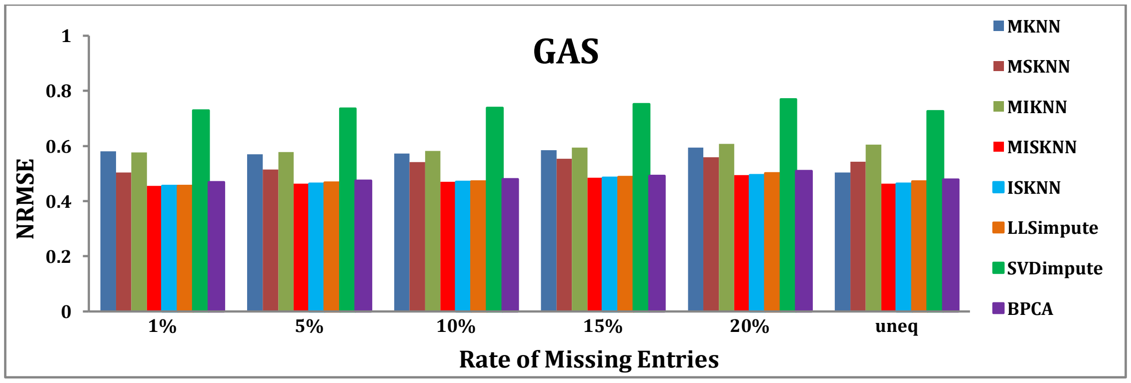


Figure 3.11(i)

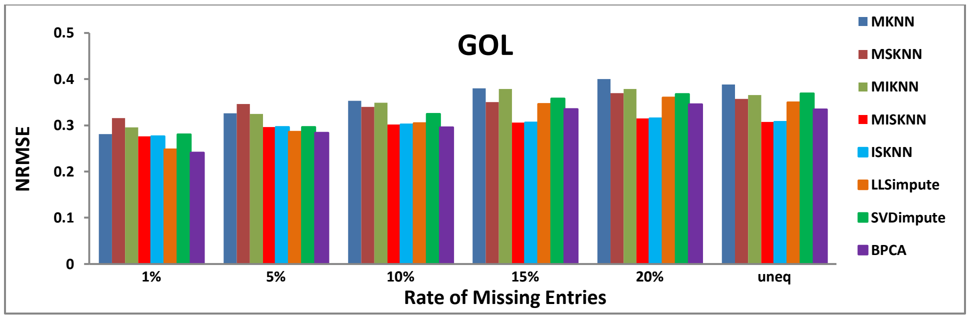


Figure 3.11(j)

Figure. 3.11. Comparative performance analysis of different versions of the proposed method for existing well-known imputation techniques in terms of NRMSE

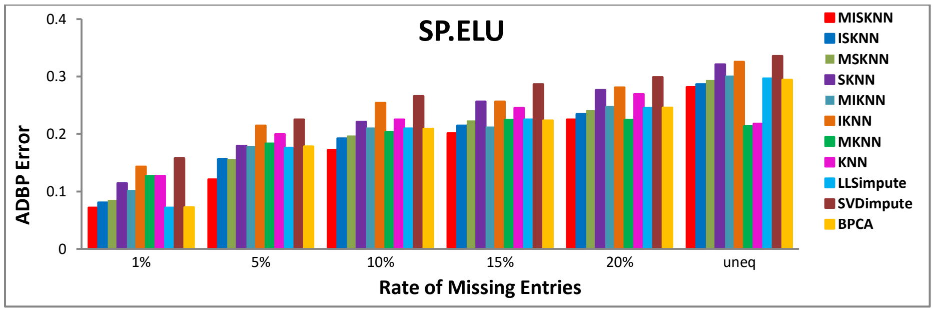


Figure 3.12(a)

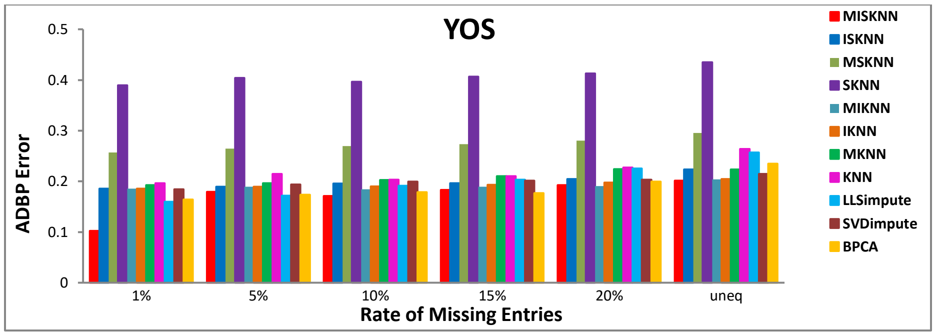


Figure 3.12(b)

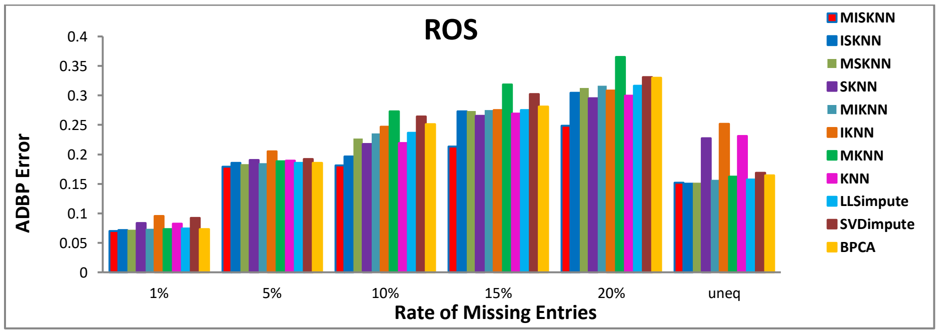


Figure 3.12(c)

Figure 3.12. ADBP error of different methods for SP.ELU, YOS, and ROS datasets

3.6.8 Comparison of Performance based on ADBP error

To measure the estimation capability of the different versions of the proposed methods, these methods are compared with the existing well-known imputation techniques with respect to the ADBP error. The results have been shown here for datasets of three different categories: (a) SP.ELU (b) YOS (c) ROS.

In Figure 3.12, the results are shown for the best parameter values of all the methods. From this figure, it has been observed that for SP.ELU dataset, MISKNNimpute performs better than all other methods at higher missing rates.

In mixed-type global structured based dataset YOS, MISKNNimpute provides lesser error than all other methods. For a 5% missing rate LLSimpute gives slightly better performance than MISKNNimpute while for 15% missing rate BPCA provides a little better performance than that of the MISKNNimpute method.

In the ROS dataset MISKNNimpute shows notably better performance than all other methods for higher missing rate while for lower missing rate it provides slightly better performance than that of other methods.

From these results it can be said that MISKNNimpute performs significantly better for steady-state local structure-based datasets and other types of datasets, it is comp-arable with or better than the robust numerical methods. For steady-state local structure-based datasets and for the noisy time series datasets the performance of its other versions is comparable with the numerical methods. For mixed datasets except for MSKNN-impute, the performance of its other versions is comparable with numerical methods.

3.7 Discussion

A large number of established imputation techniques for gene expression data exist in the literature. These methods are highly dataset dependent and application-driven also. So, there exists no one single method which is best suited for all types of datasets and all types of applications. Although more advanced imputation techniques have been deve-loped, the K-nearest neighbor principle-based imputation techniques continue to retain their popularity as these methods are very simple and usually generate consistent results in different types of datasets and also in different applications. However, their drawbacks have generally not been discussed in literature except for a few papers. In this regard, a framework is proposed in this paper for more accurate neighborhood configuration in or-der to increase the prediction accuracy of the traditional K-nearest neighbor rule-based estimation techniques by addressing the deficiencies that these suffer from.

In this context, the reasons behind the improved effectiveness of the modified KNN and its several versions are discussed here. The framework introduced here consists of two parts: one is the proposed $HPCMSR$ based neighbor gene selection and the second one is the transformation procedure applied to the relevant neighbor genes.

Advantages of the proposed HPCMSR based neighbor gene selection over other distance metric-based gene selection procedure

In traditional *K*NNimpute, S*K*NNimpute, and I*K*NNimpute methods, Euclidean distance is used to measure the similarity between any two genes. As already discussed, Euclidean distance is not a useful measure for extracting pattern-based similarity among genes. This is the one reason why the *K*NN based imputation algorithms turn out to be less effective when compared to the advanced imputation techniques.

Second, the Pearson correlation coefficient is a widely used measure to select pattern - based similarity among genes and already has been used as a similarity measure in traditional *K*NNimpute, S*K*NNimpute, and I*K*NNimpute methods. It is capable of selecting positively co-expressed as well as negatively co-expressed neighbor genes of the tar-get gene, but the problem is that it is not capable of differentiating among shifting patterns and scaling patterns using the Pearson correlation coefficient. If the two candidate neighbor genes are of the same pattern based structure but one is a scaled pattern and another is a shifted pattern with respect to the target gene, then their Pearson correlation value will be the same and it adversely affects the weighted average process. Another problem is that if the two candidate neighbor genes are of the same pattern based structure but with a different shifting factor with respect to the target gene, then also their Pearson correlation value will be the same and it too affects the weighted average process adversely.

Euclidean and Pearson correlation coefficient based hybrid PEH distance is capable of selecting positively co-expressed and magnitude wise closer genes, but still cannot handle all types of negatively co-expressed and perfectly scaling pattern-based genes. So, again weighted average procedure is affected.

To overcome the above-mentioned problems, the hybrid distance $HPCMSR$ is introduced here which is a combination of the Pearson correlation coefficient and Mean squared residue (MSR) score. A mean squared residue score was proposed to measure the compactness of biclusters in the biclustering framework. Here in this paper, we have used the MSR score as a score function between two genes. The advantage of the MSR score is that if two genes are highly positively co-expressed and shifted then the MSR value be-tween them is low but if they are highly positively co-expressed and scaled then the MSR value between them is high. On the other hand, if the genes are negatively co-ex-pressed, then the MSR score between them is high irrespective of whether they are shifted patterns or scaled patterns.

Due to the use of the Pearson correlation coefficient and Mean squared residue (MSR) score in the proposed hybrid distance, this distance function can select both the positively co-expressed and negatively co-expressed shifted genes and also discard the scaling patterns. This helps to improve the accuracy of modified *K*NN, modified S*K*NN, modified I*K*NN, and modified IS*K*NN methods.

Advantages of the proposed transformation procedure

After the selection of neighbor genes using $HPCMSR$ score, every neighbor gene is transformed using the following proposed transformation procedure. If any neighbor gene is negatively correlated with the target gene then this gene is inverted to itself by using an inversion method that makes it positively co-expressed with the target gene. Subsequently, all these positively co-expressed genes are shifted to the target gene by a shifting method so that they will come closer to the target gene. Finally, the weighted average of those transformed neighbor genes is taken to predict the missing values, where weights are calculated using Euclidean distance. By this transformation, it has become possible to take the weighted average of only the pattern-based most similar (positively co-expressed) and magnitude wise closer neighbor genes. This removes the drawbacks of the Euclidean distance and Pearson correlation coefficient based distance measure.

Advantages of MS*K*NNimpute over traditional S*K*NNimpute

In case of the conventional S*K*NN imputation method, initially, the entire gene expression matrix (G) is divided into two sub-matrices: one containing genes with missing entries ($G_{missing}$) and the other containing genes with no missing entries ($G_{nomissing}$). The estimation method starts from the gene with minimum missing positions and neighbors genes are selected from $G_{nomissing}$. As missing rates increase, $G_{nomissing}$ becomes almost blank because the maximum number of genes contains at least one missing entry. So, at a higher missing rate, its performance degrades. In the proposed modified S*K*NN method, the original data matrix is not divided into two parts and neighbor genes are selected from the entire matrix just like *K*NNimpute. It thus helps in improving the performance of modified S*K*NNimpute at higher missing rates.

Advantages of MIS*K*NNimpute over traditional IS*K*NNimpute

In the IS*K*NNimpute method a novel hybrid distance is used for neighbor gene selection. The novel distance cannot ignore scaling pattern-based genes and cannot properly consider negatively co-expressed genes. On the other hand, in the MIS*K*NNimpute method at the time of neighbor selection, all types of pattern-based genes are considered and information for the weighted average is taken from all types of genes (except scaling) after transforming them into positively co-expressed and magnitude wise closer neighbor genes. As a result, prediction accuracy increases.

Drawbacks of the Proposed Framework

Based on this framework the performance of modified *K*NNimpute and its several versions will improve if there exist scaling pattern genes and negatively or positively co-expressed and shifted pattern-based most similar genes in datasets. In absence of these types of genes in datasets, the performance of modified *K*NNimpute and its several versions will be the same as traditional *K*NNimpute and its corresponding several versions.

Another drawback is that if in a dataset only scaling pattern-based candidate genes are present then this framework will select low scaling factor based genes. In this work, how high scaling factor based genes will be handled is not considered.

3.8. Conclusion

In this work, the modified *K*NNimpute, and its different versions are proposed for improving the prediction accuracy of the traditional *K*-nearest neighbor rule-based estimation techniques. The motivation behind this work is that the neighborhood for each target gene will be constructed in such a way that the weighted average procedure will run only on the maximally positively co-expressed and magnitude wise closest genes.

To validate the effectiveness of the proposed methods, these methods are assessed and compared with existing *K*NNimpute, S*K*NNimpute, I*K*NNimpute, IS*K*NNimpute, SVDimpute, LLSimpute, and BPCA methods, using two metrics (NRMSE, and ADBPE) over ten microarray gene expression datasets considering different conditions of datasets like missing structure (equally and unequally distributed missing entries), missing rates (1 to 20% and unequal), species type (Saccharomyces cerevisiae, Atlantic salmon, and Homo sapiens), and data type (steady-state, time-series, mixed) for a different choice of model parameters.

In all cases, the proposed methods give better performance than their corresponding traditional versions, and among them, the prediction accuracy of the modified IS*K*NN-impute is the best. For local structure-based datasets, this method significantly outper-forms other numerical methods. For this type of datasets, other versions of it also give better results than numerical methods. For other types of datasets modified IS*K*NNimpute is also comparable with robust numerical methods LLSimpute and BPCA but its other versions give moderate results. The benefit of the usage of the modified IS*K*NNimpute method is that from the implementation point of view this method is simple compared to numerical methods and its prediction accuracy is comparable with robust numerical methods.

In the next chapter we have proposed another missing value prediction technique for microarray gene expression data via integrating clustering and numerical approach.

Chapter 4

Biclustering based Sequential Interpolation Imputation Technique for Missing Value Prediction

4.1 Introduction

In the last chapter we have tried to improve prediction accuracy of clustering based imputation techniques. From experimental results, it has been revealed that prediction accuracy of numerical methods based on local approaches is much better than other approaches. In these approaches, the imputation has been performed following the selection of a set of neighbor genes with respect to the gene having missing entry, based on some similarity measure. The similarity is measured considering the values of the entire set of samples. In case of gene expression data it has been found that genes (or samples) may exhibit similarity only for a subset of samples (or genes). Biclustering methods [111, 125, 126, 127] are used to compute a subset of genes (or samples) that are similar for a subset of samples (or genes). Considering this characteristic of gene expression data, recent development of biclustering based missing value imputation methods [93, 100, 128] have been becoming popular to improve the prediction accuracy. Some bicluster-based well known imputation techniques are bicluster-based impute (BIC) [93], iterative bicluster-based least square imputation (bi-iLS) [100] etc.

Among the biclustering based methods, BIC is developed in biclustering framework only. It imputes missing value by minimizing the residue score of the bicluster related to that missing position. bi-iLS generates a bicluster for every missing entry and then applies iterative local least square imputation. This method has taken advantages from both numerical and biclustering approach to estimate missing value.

Although numerical methods in the biclustering framework have been growing, to get better prediction accuracy it is still challenging. In this regard, a bicluster-based sequential interpolation imputation method named BiSIimpute [129] is proposed to predict missing values more accurately in microarray gene expression data. BiSIimpute can predict missing values in different gene expression datasets (time series and non-time series). The uniqueness of this method is to apply interpolation in biclustering framework. For every missing position in a gene, this method first generates a bicluster and then applies interpolation. The whole process of imputation starts from the gene having minimum number of missing entries. This estimated value is then placed in the original matrix and this procedure is repeated for other missing values of that gene. Sequentially all missing values of all genes in the matrix are then imputed.

The proposed method is compared with seven well known techniques, namely, *K*NNimpute [90], S*K*NNimpute [91], LLSimpute [95], SVDimpute [88], BPCA [89], NL [118], and bi-Ils[100]. Normalized root mean squared error (NRMSE) [92] and average distance between partition error (ADBPE) [113] are used to quantitatively evaluate the estimation accuracy of these techniques. The improved accuracy and robustness of the proposed method are demonstrated with the results taken using a wide range of datasets with various fault injection models.

The chapter is divided into four major sections. Section 4.2 discusses the details of the method proposed in this chapter. The results and their validations are described in the next section. Qualitative discussions and justifications of the method are presented in section 4.4. Finally, the paper is concluded in Section 4.5.

4.2 Proposed Method

In this section, a new bicluster-based sequential interpolation imputation method (BiSIimpute) is introduced for prediction of missing values. To apply interpolation we have used Lagrange's interpolation [130] technique. Throughout the paper we have adopted the following conventions to describe our imputation algorithm.

The input data is represented by a 2D matrix $G \in R^{m\times n}$ having m number of genes and n number of samples where $m \gg n$. The expression values of ith gene/row, where $1 \leq i \leq m$, for n number of samples/experiments in the gene expression matrix G , is denoted as $g_i^T \in R^{1\times n}$. g_{ij} or $g_i(j)$ denotes the expression value of gene i in jth column/ sample/experiment in G. A missing value in the tth gene at pth sample position for which the imputation is to be carried out is denoted as $g_{tp} = g_t(p) = \alpha$. $I_{m\times n}$ is the missing indicator matrix which will keep track of each missing location in G. Each entry in the indicator matrix I is denoted by a_{ij}. a_{ij} is equal to 1 if expression value of g_{ij} is available, otherwise a_{ij} is equal to 0. A gene with one or more missing values is treated as a target gene and a sample with one or more missing values is considered as a target sample.

In the next subsection, the detailed description of BiSIimpute method is given.

4.2.1 Detail Description of the Proposed BiSIimpute Method

Our algorithm sorts the genes according to their missing rate as the first step. Then imputation begins from the gene (called target gene) which has lowest missing rate. For this target gene, interpolation based estimation technique is applied to every missing position to predict the missing value for that position. This value is imputed in the gene expression matrix given as input and then used for latter imputation in other missing positions in that gene as well as in the other genes with missing positions in sequential manner.

For each missing position in the target gene, the proposed method first forms a bicluster by selecting u number of most similar correlated genes with respect to the target gene and v number of most similar correlated samples of the target sample by Pearson correlation coefficient. Then it estimates that missing value by applying Lagrange's interpolation based estimation technique on that bicluster. This estimated value is then stored in the given matrix and this same continues for other missing positions in that particular gene as well as for other genes.

4.2.1.1 Formation of Bicluster

It is assumed that genes which show similar pattern of changing tendencies in different experiments or with progress of time are under the control of the same transcription factor and are related to a similar function [111] in the cell. They are termed as correlated/co-expressed genes. The magnitude of expression levels of them may be or may not be closed. It has also been found that correlation among genes exists only for a subset of samples [111]. So, for every missing entry, while forming the bicluster, correlated genes and correlated samples are selected in respect to corresponding target gene and target sample.

At first, all missing positions in the given gene expression matrix are replaced by row averages. These are not considered for selection of correlated genes and correlated samples but are considered in interpolation based estimation technique.

$$G = \begin{pmatrix} g_1^T \\ g_2^T \\ \vdots \\ g_t^T \\ \vdots \\ g_m^T \end{pmatrix} = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1p} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2p} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_{t1} & g_{t2} & \cdots & g_{tp} & \cdots & g_{tn} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_{m1} & g_{m2} & \cdots & g_{mp} & \cdots & g_{mn} \end{pmatrix} \tag{4.1}$$

In equation (4.1), G is the data matrix, gene g_t^T is the target gene in which at pth column/sample position, the expression value (α) is missing.

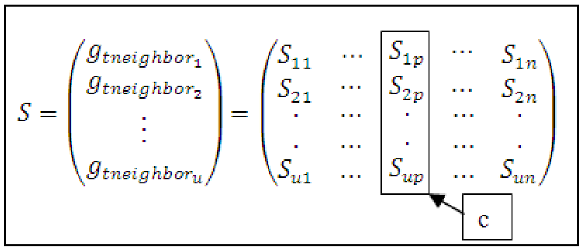
For the missing value α, at the pth column/sample position of the gene g_t^T (i.e. $g_t(p)$), u nearest correlated genes of g_t^T, represented as $g_{tneighbour_i}^T$, $1 \leq i \leq u$ are first chosen using highest magnitude/absolute Pearson correlation coefficient using equation (4.2) forming the matrix $S \in R^{u \times n}$ of G as shown in equation (4.3).

$$PR = \frac{\sum_{j=1}^{n}(g_t(j) - \overline{g_t})(g_l(j) - \overline{g_l})a_{tj}a_{lj}}{\sqrt{\sum_{j=1}^{n}(g_t(j) - \overline{g_t})^2(g_l(j) - \overline{g_l})^2}} \tag{4.2}$$

Where $\overline{g_k}$is the average values of gene vectorg_k^T. PR varies between -1 to +1.

$$S = \begin{pmatrix} g_{tneighbor_1} \\ g_{tneighbor_2} \\ \vdots \\ g_{tneighbor_u} \end{pmatrix} = \begin{pmatrix} S_{11} & \cdots & S_{1p} & \cdots & S_{1n} \\ S_{21} & \cdots & S_{2p} & \cdots & S_{2n} \\ . & \cdots & . & \cdots & . \\ . & \cdots & . & \cdots & . \\ S_{u1} & \cdots & S_{up} & \cdots & S_{un} \end{pmatrix} \quad (4.3)$$

As missing value α, is present at the pth column/sample position of the gene g_t^T (i.e. $g_t(p)$) in G, pth column of the matrix S is considered as the subsection of the target/missing column in G with respect to $g_t(p)$), and is denoted by a vector $c \in R^{u \times 1}$ shown in figure (4.1).


$S = \begin{pmatrix} g_{tneighbor_1} \\ g_{tneighbor_2} \\ \vdots \\ g_{tneighbor_u} \end{pmatrix} = \begin{pmatrix} S_{11} & \cdots & S_{1p} & \cdots & S_{1n} \\ S_{21} & \cdots & S_{2p} & \cdots & S_{2n} \\ . & \cdots & . & \cdots & . \\ . & \cdots & . & \cdots & . \\ S_{u1} & \cdots & S_{up} & \cdots & S_{un} \end{pmatrix}$
c
Figure 4.1. Formation of vector c

Now, Pearson correlation coefficient value is calculated for each column with respect to the pth column (vector c) of S. Then v nearest columns/samples to pth column/sample of S are selected according to highest magnitude/absolute Pearson correlation coefficient value. That means after initial selection of u number of neighbor genes from matrix G, subsection of these genes are reselected from S by selecting most correlated v number of columns/samples with respect to pth coumn/sample forming bicluster $\hat{S}$ as shown in equation (4.4).

$$\hat{S} = \begin{pmatrix} \hat{S}_{11} & \hat{S}_{12} & \hat{S}_{13} & \cdots & \hat{S}_{1v} \\ \hat{S}_{21} & \hat{S}_{22} & \hat{S}_{23} & \cdots & \hat{S}_{2v} \\ . & . & . & \cdots & . \\ . & . & . & \cdots & . \\ \hat{S}_{u1} & \hat{S}_{u2} & \hat{S}_{u3} & \cdots & \hat{S}_{uv} \end{pmatrix} \quad (4.4)$$

After formation of the bicluster$\hat{S}$, without no loss of generality, the subsection of target gene g_t^T named $\hat{g}_t^T \in R^{1\times(v+1)}$ is formed by placing missing value α at the first position and then placing corresponding expression values of v number of selected samples in $\hat{S}$ as shown in equation (4.5).

$$\hat{g}_t^T = (\alpha \quad r^T) \qquad (4.5)$$

where, $r^T \in R^{1\times v}$ contain expression values of g_t^T, for corresponding v number of selected samples in $\hat{S}$.

Next, the final imputation matrix $A \in R^{u+1\times v+1}$is created by concatenating $\hat{g}_t^T$ as the first row and column vector c in the first column with bicluster $\hat{S}$ as shown in equation (4.6).

$$A = \begin{pmatrix} \alpha & r^T \\ c & \hat{S} \end{pmatrix} = \begin{pmatrix} \alpha & r_1 & r_2 & \cdots & r_v \\ c_1 & \hat{S}_{11} & \hat{S}_{12} & \cdots & \hat{S}_{1v} \\ c_2 & \hat{S}_{21} & \hat{S}_{22} & \cdots & \hat{S}_{2v} \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ . & . & . & \cdots & . \\ c_u & \hat{S}_{u1} & \hat{S}_{u2} & \cdots & \hat{S}_{uv} \end{pmatrix} \qquad (4.6)$$

4.2.1.2 Interpolation Based Approximation

After bicluster formation, interpolation is applied for imputation. First, average expression profile of each column for all the rows of the bicluster $\hat{S}$ is taken forming a vector $W^T \in R^{1\times v}$. Although, neighbor genes of target gene g_t^T, are selected using absolute Pearson correlation coefficient, they are either positively correlated or negatively correlated with the target gene g_t^T. The contribution of a gene g to average expression profile is computed by using its opposite average expression profile (i.e. $-g$), for oppositely or negatively correlated genes. The meaning of this is to treat both under expression and over expression of genes equally; otherwise, expression of two genes with opposite polarity would cancel each other during their average profile calculation. Due to this reason, average expression profile of all rows (genes) of the bicluster $\hat{S}$ is calculated in the following way. At first, Pearson correlation coefficient is calculated between each row $\hat{S}_g$of the bicluster $\hat{S}$ with the first row (except the first column value that means vector r^T) of bicluster A. If Pearson correlation coefficient value is greater than 0 then it means that the particular row $\hat{S}_g$ of the bicluster $\hat{S}$ has same polarity (positively correlated) with vector $r^T$and it remains unchanged in calculating average expression profile. If Pearson correlation coefficient value is less than 0 then it means that the particular row $\hat{S}_g$ of the bicluster $\hat{S}$ has opposite polarity (negatively correlated) with vector $r^T$and its 'sign-flipped' expression value is changed to $-\hat{S}_g$ in calculating average expression profile. Changing the 'sign-flipped' value of $\hat{S}_g$ indicates that if $\hat{S}_g$is oppositely correlated with vector r^Tthen its correlation is changed in the same direction. In this way every row of bicluster $\hat{S}$ is considered and its sign is changed according to the correlation. Then, average expression profile of these rows is taken for generating vector W^T as shown in figure 4.2.

The vector W^T is taken as X-axis; the first row, r^T of the imputation matrix A is taken as Y-axis. Then, Lagrange's interpolation technique is applied to find $y_1 = f_1(x)$ for a particular value of x where x is the average value, c_{avg}, of the vector c of A.

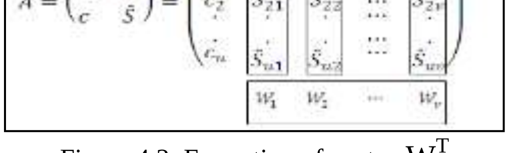

$A = \begin{pmatrix} \alpha & r^T \\ c & \hat{S} \end{pmatrix}$
α r_1 r_2 $\cdots$ r_v
c_1 $\hat{S}_{11}$ $\hat{S}_{12}$ $\hat{S}_{1v}$
c_2 $\hat{S}_{21}$ $\hat{S}_{22}$ $\hat{S}_{2v}$
c_u $\hat{S}_{u1}$ $\hat{S}_{u2}$ $\hat{S}_{uv}$
W_1 W_2 $\cdots$ W_v

Figure 4.2: Formation of vector W^T

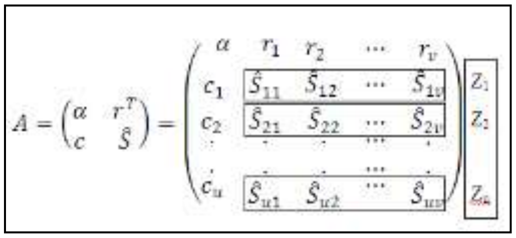


Figure 4.3: Formation of vector Z

Next, average expression profile for each row of all the columns of the bicluster $\hat{S}$ is calculated forming a vector $Z \in R^{u\times 1}$ in the following manner as shown in figure 4.3. This vector Z is taken as X-axis and the vector c in the imputation matrix A is taken as Y-axis. Lagrange's interpolation is then used to get $y_2 = f_2(x)$ for a particular x where x is the average value, r_{avg}, of the vector r^T of A. Finally α is calculated as the average of $y_1$and y_2.

Algorithm: BiSIimpute

Input: The matrix $G_{m\times n}$ with m genes and n samples where $m >> n$.
Output: The estimated values of all missing entries.

```
1. Initialization: Substitute all missing entries in $G$ by their corresponding row averages.
2. Sort the rows in $G$ in ascending order with respect to number of missing entries.
3. For each of the missing location $p$ in every target gene $g_t^T$ in $G$ perform the following.
    i. Compute Pearson correlation coefficient value between target gene $g_t^T$ and other genes. Select $u$ nearest correlated genes according to highest magnitude/absolute Pearson correlation coefficient to form $S$.
    ii. Compute Pearson correlation coefficient value for the missing column $p$ with other columns in $S$ and select $v$ nearest samples according to highest magnitude/absolute Pearson correlation coefficient to form $\hat{S}$ (bicluster).
    iii. Generate the imputation matrix $A$ by concatenating $\hat{g}_t^T$ as the first row and column vector $c$ in the first column with bicluster $\hat{S}$.
    iv. Calculate $r_{avg}$ as the mean of values in the vector $r^T$and calculate $c_{avg}$ as the average of values of the vector $c$.
    v. For every row $\hat{S}_g$ of bicluster $\hat{S}$ do:
        a. Calculate Pearson correlation coefficient of vector $r^T$ of imputation matrix $A$ and row $\hat{S}_g$ of bicluster $\hat{S}$.
        b. If Pearson correlation coefficient > 0 then row $\hat{S}_g$ remains same otherwise it is treated as $-\hat{S}_g$ by changing its sign.
    vi. Calculate the average expression profile of all rows of bicluster $\hat{S}$ forming a vector named $W^T$.
    vii. Apply Lagrange's interpolation method, considering $W^T$as $X$-axis and $r^T$ vector as $Y$-axis, for a definite $x$ value as $c_{avg}$ and calculate $y_1 = f_1(c_{avg})$.
```

viii. For every column $\hat{S}_c$ of bicluster $\hat{S}$ do:
   a. Calculate Pearson correlation coefficient of vector c of imputation matrix A and column $\hat{S}_c$ of bicluster $\hat{S}$.
   b. If Pearson correlation coefficient > 0 then column $\hat{S}_c$ remains same otherwise it is treated as $-\hat{S}_c$ by changing its sign.

ix. Calculate the average expression profile of all the signed columns of bicluster $\hat{S}$, forming a vector Z.

x. Apply Lagrange's interpolation method, considering Z vector as X-axis and c vector as Y-axis for $x = r_{avg}$ and calculate $y_2 = f(r_{avg})$.

xi. Finally calculate α as the average of y_1 and y_2 and that value is imputed at the proper position in the matrix G.

4. End.

4.2.2 Computational complexity analysis:

For every missing position BiSIimpute first creates a bicluster (sub-matrix) of order $u + 1 \times v + 1$ and then applies interpolation based approximation. To create a bicluster, it selects u number of nearest correlated genes of target gene. It takes time of order $(O(mn) + O(u))$ as each estimation of distance/similarity calculation between two genes takes linear time of order $O(n)$ and to select u number of genes $O(u)$ time is needed. Then it picks up v number of coexpressed columns of the target column from this selected u number of genes, which $(O(un) + O(v))$time. As $u \ll m$and $v \ll n$, time complexity of creation of the bicluster ($\hat{S}$) is $O(mn)$. After forming the bicluster, it first takes the average expression profile of $u$neighbor genes taking $O(v)$ comparison and then applies interpolation. Similarly, it also takes the average expression profile of v neighbor samples taking $O(u)$ comparison and then applies interpolation. In the bicluster, interpolation is applied two times, one for genes and another for samples taking $O(v)$ and $O(u)$ time complexity. So for q number of missing positions, it takes $O(qmn) + O(qu + qv)$ time. As u and $v$are very less compared to q, total time complexity of this method is $\approx O(mnq)$.

4.3 Results

In this paper, efficiency of the proposed BiSIimpute is evaluated by comparing it with a number of existing eminent imputation techniques, namely "*K*NNimpute" [90], "S*K*NNimpute" [91], "LLSimpute"[95], "SVDimpute"[88], "Bayseian principal component analysis (BPCA)"[89], "NL"[118], and "bi-iLS"[100]. Experiments have been carried out over nine different datasets. Accuracy of our algorithm is compared with other algorithms using two well known metrics: i) normalized root mean squared error (NRMSE) and ii) average distance between partitions error (ADBPE) as discussed in the previous chapter.

All the above mentioned methods are implemented in C using Linux environment in a machine with a 4 GB RAM, and 3.2 GHz core i3 processor.
Experiments have been carried out over nine different datasets as mentioned

4.3.1 Gene Expression datasets

For our experimentation, nine different microarray gene expression datasets have been used. These datasets are classified into three categories: (1) time-series dataset (SP.AFA[114], SP.ELU[114], BAL[120]) (2) mixed data set. Mixed dataset comprise time-series data as well as non time-series data or multiple time-series data measured in different experimental conditions, YOS[121]) (3) non-time series dataset (GAS[115], ROS[116], GOL[117], and Tymchuk[118]). A synthetic dataset generated by SynTReN [122] is also considered here. These datasets are described in Section 3.5.6.1.

4.3.2 Evaluation of Performance

The accuracy of the proposed algorithm is rigorously tested with already mentioned popular techniques, namely, *K*NNimpute, S*K*NNimpute, LLSimpute, SVDimpute, BPCA, NL and bi-iLS method.

4.3.2.1 Selection of values for model parameters of BiSIimpute method

The prediction capability of BiSIimpute algorithm depends on the dimension of the selected bicluster. These parameters ($u$and v) depend on the type of dataset and also on the rate of missing entries present in that dataset. For this study, BiSIimpute is executed in 50 simulations for each dataset with different values of u and v and by taking different percentages of MVs to empirically evaluate optimal values for u and v. In Table 4.1 the optimal values of u and v for each dataset with different missing rates have been shown.

From Table 4.1, it is seen that in SP.AFA for 1%, 5%, 10%, 15%, 20%, and unequal missing rate, BiSIimpute gives lower NRMS error values when v is set at 5 and u is selected between 5 and 30. For unequal case, this method gives lower NRMS error values when $v = 5$ and u is selected between 20 and 30. For SP.ELU and GAS datasets this method generates best results when u is is in the range of 10 to 20 and v is set as 5 and 8 respectively.

For BAL and GOL datasets BiSIimpute shows consistently lower NRMS error values when u is selected between 30 and 50 and v is set at 12 and 20 respectively.

The proposed algorithm gives best result when u is set at 30 and v is set at 7 for 1% missing entries of YOS dataset. But in cases of 5%, 10%, 15%, and 20% missing values, it gives best result when u varies between 20 and 30 and v varies between 7 and 10.

For Tymchuk dataset in all cases, BiSIimpute gives lower NRMS error when u is selected between 10 and 15 and v is in the range of 5 to 10.

For Synthetic dataset this method generates best result when u is set at 30 and v is set at 5.

Table 4.1. Values of model parameters u and v with respect to lowest NRMSE for different datasets with different missing rates

Dataset	Rate of Missing Entries	No. of Genes(u)	No. of Samples(v)	NRMSE	Dataset	Rate of Missing Entries	No. of Genes(u)	No. of Samples(v)	NRMSE
SP.AFA	1%	15	5	0.402357	Tymchuk	1%	10	5	0.495795
	5%	10	5	0.42782		5%	10	10	0.467629
	10%	15	5	0.46654		10%	10	10	0.485912
	15%	30	5	0.50456		15%	15	10	0.49886
	20%	20	5	0.537864		20%	10	10	0.505428
	uneq	20	5	0.58353		uneq	10	10	0.49804
SP.ELU	1%	10	5	0.296244	GOL	1%	50	20	0.256414
	5%	20	5	0.292877		5%	50	20	0.299549
	10%	20	5	0.316235		10%	50	20	0.31596
	15%	20	5	0.34834		15%	50	20	0.346384
	20%	20	5	0.362286		20%	50	20	0.371475
	uneq	20	5	0.34101		uneq	50	20	0.340752
BAL	1%	50	12	0.316962	YOS	1%	30	7	0.335538
	5%	50	12	0.327577		5%	20	7	0.366803
	10%	50	12	0.352528		10%	20	7	0.363884
	15%	50	12	0.37017		15%	10	7	0.37031
	20%	50	12	0.380543		20%	20	7	0.378129
	uneq	30	12	0.32835		uneq	20	7	0.400382
GAS	1%	10	8	0.350636	Synthetic	1%	30	5	0.334418
	5%	20	8	0.359482		5%	30	5	0.30731
	10%	20	8	0.363014		10%	30	5	0.27102
	15%	20	8	0.374638		15%	20	5	0.357481
	20%	20	8	0.382701		20%	20	5	0.278159
	uneq	10	5	0.356349		uneq	20	5	0.31663

From all these results, it can be concluded that for datasets with high entropy values (locally correlated datasets) this method gives minimal NRMS error when u is chosen in the interval of 5 to 30. For global structured based datasets this method generates optimal results with a higher range of u values, here that is from 30 to 50. For all types of datasets BiSIimpute method shows lower NRMS error values when v is set at approximately 30% of total number of samples.

4.3.2.2 Selection of optimal model parameter values for other imputation algorithms

To select optimal model parameter values for different missing value imputation algorithms other than the BiSIimpute method, 50 simulation runs of each of these methods are tested empirically for each dataset with different percentage of MVs under different values of model parameters. Then, the model parameters are chosen for these methods which generates the lowest NRMS error. After investigation, it is concluded that for the neighbor-based imputation methods (KNNimpute, SKNNimpute, NL method), the optimal number of nearest neighbor genes (K) is found within the range between 10 and 20 and is consistent with the previous studies [123]. SVDimpute generally uses a percentage of eigenvalues between 0.15 and 0.25 [123]. The LLSimpute has built in parameter optimization characteristics within the algorithm itself [123]. The value of model parameter for an imputation algorithm is selected as the one which gives the best result for that algorithm in most of the cases.

Next, 50 simulations is performed again using the optimized values of model parameters of different methods for different datasets and for different missing rates. This is described in the next section.

4.3.2.3 Comparative Performance Analysis based on NRMS error

The efficiency of the newly proposed BiSIimpute method is analyzed with *K*NNimpute, S*K*NNimpute, SVDimpute, LLSimpute, BPCA, NL, and Bi-iLS methods, after applying them on nine microarray datasets of multiple category over different rates of missing positions.

Figure 4.4 plots the performance of different methods as a function of 1%, 5%, 10%, 15%, 20%, and unequal percentages of missing entries on different datasets. The performance is judged by the NRMS error value. NRMS error tends to degrade as the percentage of MVs increases for all methods.

For noisy time-series datasets (i.e. for SP.AFA and SP.ELU), BiSIimpute method generates the best results in most of missing rates. When missing rate is below 5% in these datasets, BPCA, Bi-iLS, and LLSimpute methods provide better results than the proposed method. For other missing rates, the proposed method generates the best results, Bi-iLS generates second best results while BPCA and LLSimpute provide better results than other methods for these two datasets. For SP.AFA and SP.ELU datasets, the NRMS error values generated by the proposed method vary between 0.3 and 0.6 and 0.17 and 0.4 respectively. In these two datasets, *K*NNimpute, S*K*NNimpute, and NL methods show similar results for different missing rates. SVDimpute provide better results than *K*NNimpute, S*K*NNimpute, and NL method in these two datasets.

BAL is a time series global structured based dataset of human cancer. In this dataset, BPCA generates the best results in all missing rates. bi-iLS and LLSimpute provide similar results in all missing rates. BiSIimpute method generates worse results than BPCA, bi-iLS, LLSimpute and performs better than other imputation methods. The NRMS error values generated by this method are in the interval from 0.33 to 0.35.

According to Tryoyanskaya [90], GAS dataset is the most challenging dataset for prediction. Figure 4.4 shows that BiSIimpute outperforms LLSimpute, BPCA, *K*NNimpute, S*K*NNimpute, SVDimpute, NL method and gives a little better result than Bi-iLS for all missing rates. The NRMS error values generated by BiSIimpute method are in the interval from 0.32 to 0.36 for GAS dataset.

ROS dataset is a steady state human cancer dataset where missing value estimation is very difficult. High entropy value of this dataset indicates that this is a local structure based dataset. From Figure 4.4, it can be concluded that in this dataset, BiSIimpute notably outperforms all methods for all missing rates. BiSIimpute generates NRMS error values within range from 0.52 to 0.55 for this dataset.

Tymchuk is another steady state local structure based dataset of Atlantic Solomon and the proposed method also has generated minimal NRMS error for all types of missing entries. For this dataset bi-iLS and LLSimpute generate similar results in all cases. The NRMS error values generated by the propose method are in the range of 0.45 to 0.50.

GOL dataset is a steady state global structure based dataset of human cancer. In this dataset BPCA generates the best results for all types of missing rates. BiSIimpute gives similar results as LLSimpute and bi-iLS. It also produces better results than other imputation methods for all types of missing rates. For this dataset NRMS error values generated by the proposed method varies between 0.27 and 0.34.

In YOS dataset, BPCA generates the best result in all cases. BiSIimpute generates worse result than BPCA. Above 15% missing rate, it generates better results than bi-iLS and LLSimpute. *K*NNimpute, SVDimpute, and NL methods have shown similar pattern of performance for different missing rates. For this dataset, NRMS error values generated by the proposed method varies between 0.32 and 0.38.

For Synthetic dataset, BiSIimpute notably outperforms all methods for all missing rates. BiSIimpute generates NRMS error values within range from 0.25 to 0.35 for this dataset.
From these results it can be concluded that for local structure based datasets, the proposed method gives notably better results than all other well known existing methods. But for global structure based datasets, this method always generates a little worse results than BPCA and similar results with LLSimpute and bi-iLS method.

In case of noisy time-series datasets (SP.AFA and SP.ELU) with lower missing rate, the proposed method shows poor performance due to presence of noise as happened in [91]. At higher missing rate, the performance of BiSIimpute increases due to reuse of imputed information for its sequential nature.

In all cases the proposed method outperforms *K*NNimpute, S*K*NNimpute, SVDimpute, and NL methods.

4.3.2.4 Comparative Performance Analysis based on ADBP error

The prediction quality of BiSIimpute method is evaluated by comparing its ADBP error with that of the existing methods. The result is shown in figure 5 only for three datasets (1) SP.ELU (2) ROS (3) YOS.

For this purpose, first the $G_{complete}$ matrix is partitioned (U) into c number of clusters using c-means clustering algorithm, for $c = 2, 3, \dots \dots ..50$. Then every time, test matrix G is imputed with each imputation method generating matrix G_{impute} and then divided into c number of clusters (partition V) using c-means clustering for $c = 2, 3, \dots, .50$.

Figure 4.5 plots the performance of different methods as a function of 1%, 5%, 10%, 15%, 20%, and unequal percentages of missing entries on SP.ELU, ROS, and YOS datasets. Performance of our algorithm is judged by the ADBP error value. The results for best c values are shown here using the average normalized ADBP error. ADBP error tends to increase as the increasing of percentage of missing values for all methods.

From Figure 4.5, it can be observed that BiSIimpute method is superior to the existing methods in case of SP.ELU and ROS datasets and generates a little worse result than LLSimpute and BPCA in case of YOS dataset. For all other datasets which are not shown here, BiSIimpute generates similar or better results than all other existing methods mentioned here. In Figure 4.5, Noimp represents clustering ignoring missing values on test matrix G and then ADBP error is calculated.

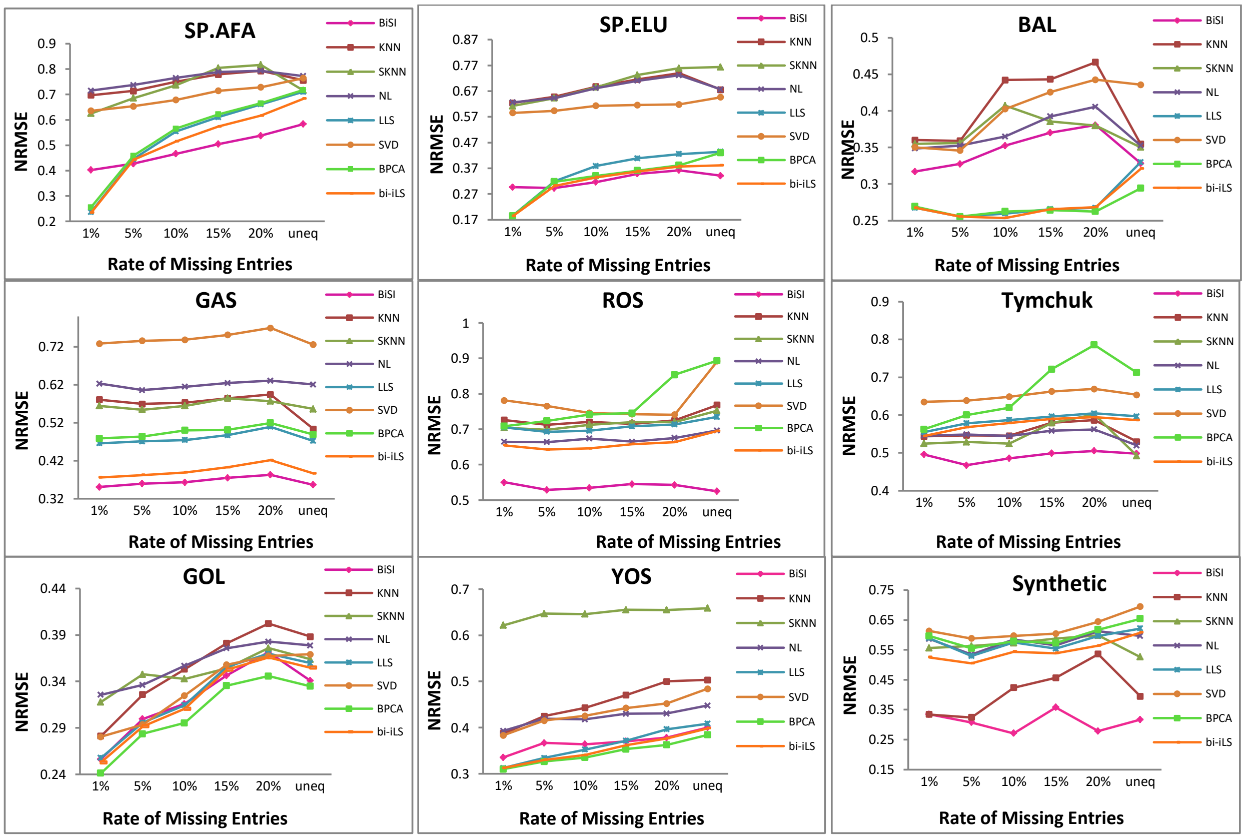


Figure 4.4. Comparative performance analysis of different methods based on NRMS error for different datasets with different percentage of missing entries

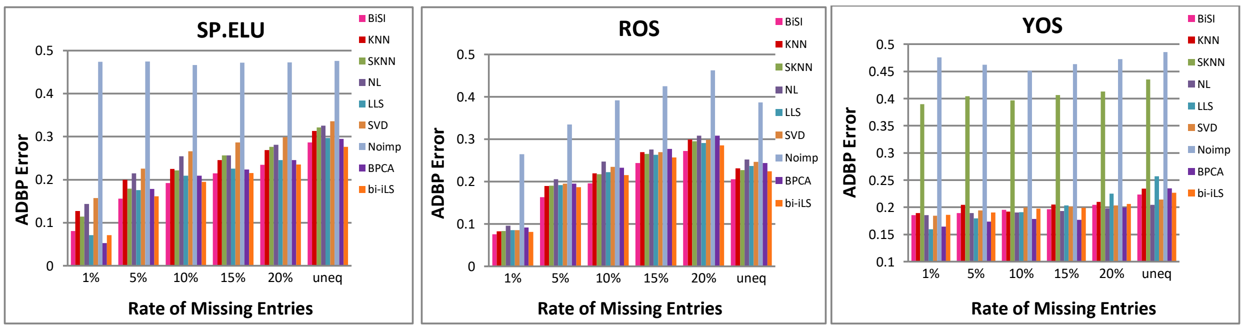


Figure 4.5. ADBP error for SP.ELU, ROS, and YOS due to different methods

4.3.2.5 Comparative Performance Analysis of BiSIimpute with respect to BCCA and our sequential interpolation based imputation technique

We have also used an existing popular biclustering algorithm, named Bi-Correlation clustering algorithm (BCCA) [108] for bicluster formation and then applied our sequential interpolation based imputation method for imputation considering it as BCCA+SIimpute. We have tested our results for datasets SP.AFA and Synthetic as shown in Figure 4.6. In all cases our method BiSIipute outperforms BCCA+SIimpute method.

BiSIimpute generates a bicluster corresponds to a missing position and then imputation is performed. The limitation of BCCA+SIimpute method is that, biclusters are generated without considering missing entries. Some genes have not been included in any bicluster formed by BCCA and missing entries in those genes have not yet been imputed by this method.

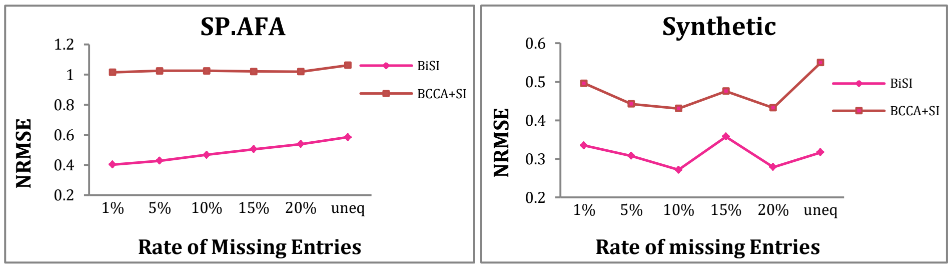


Figure 4.6. Comparative performance analysis with respect to NRMSE of BiSIimpute and BCCA+SIimpute

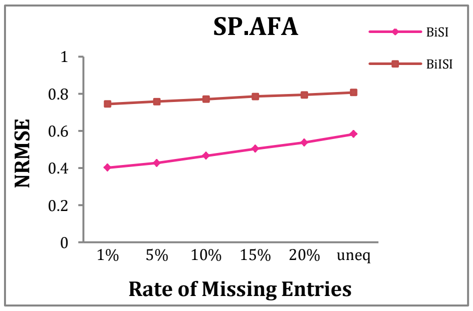


Figure 4.7. Comparative performance analysis with respect to NRMSE of BiSIimpute and BiISIimpute

4.3.2.6 Comparative Performance Analysis of BiSIimpute with respect to iterative version of BiSIimpute

We have also generated iterative version of our proposed BiSIimpute named BiISIimpute and compared it with our proposed method for SP.AFA dataset with respect to NRMS error for different missing rates as shown in figure 4.7. From these results, it has been said that BiSIimpute outperforms for all missing rates in SP.AFA dataset.

Standard Deviation Measurement

In this paper, several data for each missing rate in every dataset are generated. For each such generated data, results are taken to evaluate prediction accuracy of all methods that are considered. Then, the standard deviations of NRMS error values of every method are calculated for each generated data. It has been observed that standard deviation of BiSIimpute is in the same range as the standard deviation in other methods.

4.4. Discussion

There are a large number of imputation strategies for estimation of missing values in microarray gene expression data. Among them *K*NNimpute, S*K*NNimpute are clustering based classical approaches and LLSimpute, SVDimpute, BPCA, NL, and bi-iLS etc. are well known statistical and numerical analysis based methods. All these methods except SVD, and BPCA utilize local information. On the other hand, SVD and BPCA use global correlation structure of entire gene expression matrix. These methods except NL consider gene-gene linear dependency. Tianwei's NL method considers nonlinear relationship between genes. bi-iLS is another local imputation method in which least square is applied in biclustering framework for imputation. In this paper, a bicluster-based sequential interpolation technique (BiSIimpute) is proposed. The innovativeness of this method lies in applying interpolation based imputation technique in biclustering framework. In this section, the rationale behind the efficacy of the proposed BiSIimpute method is qualitatively justified.

BiSIimpute not only considers correlated genes of the target gene but also considers correlated samples of the target sample and outperforms all methods when genes have dominant local correlation.

It is assumed that genes which show similar pattern of changing tendencies under a number of experiments/samples are under the control of the same transcription factor and are related to similar function [100] in the cell and they are termed as correlated genes. It has also been found that correlation among genes exists only for a subset of samples [100]. On the basis of this concept, BiSIimpute forms one bicluster for each missing entry and then applies interpolation based imputation. Although, in bi-iLS also for every missing entry a single bicluster has been formed but biclustering formation framework is different in BiSIimpute. During bicluster formation genes are selected by Bi-ils using weighted Euclidean distance measure while in our method genes are selected using absolute Pearson correlation coefficient value. In case of sample selection Bi-iLS uses a different approach than our proposed method. Use of Euclidean distance selects magnitude-wise similar genes/samples in bi-iLS method which ignore pattern based similarity among genes/samples but in BiSIimpute, importance is given on selection of similar pattern based genes and samples using absolute Pearson correlation based similarity. It increases the prediction accuracy of our method than bi-iLS.

On the other hand, *K*NNimpute, S*K*NNimpute, LLSimpute, and NL select neighbor genes by considering their expression profile across all conditions/samples. The output of SVDimpute and BPCA cannot be globally optimal if the genes have a dominant local similarity. This is because these techniques consider all genes and the correlation between many of these genes and the target gene is very little.

The BiSIimpute algorithm considers both positive and negative correlation values similar to [98]

*K*NNimpute, and S*K*NNimpute methods linearly combine similar genes by using their weighted average. These methods use Euclidean distance, that is not an optimal metric to compute the gene similarity. This degrades the accuracy of *K*NNimpute, and S*K*NNimpute methods. In BiSIimpute to predict missing value in a target gene, correlated genes are considered which are similar according to their patterns. Pearson correlation coefficient is a useful similarity measure to find pattern based similarity between two objects. So, BiSIimpute algorithm uses Pearson correlation coefficient to select both correlated genes and correlated samples with respect to target gene and target sample respectively. Based on absolute correlation coefficient value, it selects genes and samples which are either positively correlated or negatively correlated with the target gene and target sample respectively. Absolute Pearson correlation coefficient is used so that highly correlated but inverted genes/samples may be taken care of.

Sequential nature of BiSIimpute generates lower estimation error when missing genes are highly correlated with each other

BiSIimpute is sequential like S*K*NNimpute. That means, it will consider the already estimated values for prediction of upcoming missing entries. So, while missing rate increases, sequentially reusing the imputed data prevents the propagation of imputation errors. This is unlike in conventional methods such as *K*NNimpute, LLSimpute, SVDimpute, BPCA, NL and bi-iLS. Though S*K*NNimpute considers reuse of already estimated missing values, it is not applicable in case of data set with higher missing rate, as almost every gene contains missing values. So, BiSIimpute provides the most accurate missing value prediction approach for datasets having high as well as low missing rate.

BiSIimpute considers gene-gene linear dependency

BiSIimpute method like *K*NNimpute, S*K*NNimpute, LLSimpute, bi-iLS uses linear dependency between two genes.

4.5. Conclusion

In this chapter, a bicluster-based sequential interpolation imputation method called BiSIimpute is proposed for estimation of missing values in DNA microarray data. The novelty of this method is that first time interpolation based imputation technique is applied in biclustering framework. The estimation accuracy of BiSIimpute is evaluated and compared with that of similar popular methods over various types of datasets (time-series, non time-series and hybrid datasets) with different proportions of missing rates (1, 5, 10, 15, 20, and unequal percentage) using different values of $u$and v.

Using NRMSE, and ADBPE metric, it is found that the proposed method outperforms all other methods mentioned here for different local structured based datasets. So, it is a new robust approach to estimate missing values in different local structured based microarray gene expression datasets.

In third and fourth chapters of this thesis, we have proposed two types of missing value prediction techniques. Among them one is developed based on clustering approach and second one is developed based on numerical approach. Actually missing value prediction in microarray data is most dominant part of data preprocessing. After data pre-processing, several machine learning techniques are applied to mine gene expression data. One such mining task is to find functionally similar groups of genes via applying clustering techniques on genes of gene expression data and in the next chapter we have proposed a new partition based clustering framework for clustering genes in gene expression data.

Chapter 5.

HCFPC: A New Hybrid Clustering Framework using Partition-Based Clustering Algorithms to Group Functionally similar Genes from Microarray Gene Expression Data

5.1 Introduction

It has been already discussed that DNA Microarray Technology [15, 16], which is one of the most famed biotechnologies, has become blessings in the field of bio-medical research since last two-decade, because downstream analysis of microarray gene expression data have several applications in different real life crucial fields such as cancer type identification, prognosis, therapeutic targets prediction for cancer etc. [15-30].

Among several analysis tasks one important analysis task of microarray gene expression data is to find hidden patterns among genes present in this data to extract relevant information which will be beneficial for functional genomics. This task is very challenging due to presence of huge number of genes. One of the solutions of this task is to apply clustering techniques for clustering genes in gene expression data to find natural grouping present among genes for identification of interesting hidden patterns among them [17-19].

Clustering technique [17-19] which is one kind of unsupervised machine learning technique is a very important tool for mining this data. In gene clustering, genes are clustered based on their similar expression patterns that means the co-expressed genes (genes with similar expression values) are grouped [17-19]. This is very helpful for identifying genes with similar cellular functions as it is already known that co-expressed genes may have similar behaviour. It is also beneficial for predicting the functionality of many genes for which functional information has not been previously available in the databases [17-19]. Furthermore, from these clusters of co-expressed genes, valuable information can be extracted to understand functionality of different transcriptional regulatory networks or scrutinizing specific pathways [20, 131].

From different empirical studies it has been found that microarray data has several inherent characteristics [17-19, 132]. One important characteristics of microarray data is that gene clusters generated from this data are highly overlapping with each other or even embedded in one another [132, 133]. Another important characteristic of gene expression data is that it is inherently noisy and a small number of genes out of a huge pool of genes has significant importance from a biological point of view [134]. Although, a huge number of different notable clustering methods like Hierarchical clustering [17-19, 135], hard partition based clustering [17-19, 137, 138], fuzzy clustering algorithms [139-142, 143-146], graph theoretical approaches-based clustering [17-19,147, 148], model-based clustering [17-19] and finally tight clustering methods [134] are applied in this field to find clusters of co-expressed genes but maximum of them have not given their emphasis on elimination of noise except a few. So, now a days the main challenge in gene clustering algorithms is to find overlapping clusters of biologically relevant small collections of genes via eliminating noisy genes from the datasets.

Although a huge number of different category based clustering algorithms are applied on gene expression data but among these partition based clustering algorithms specially different versions of k-means are most popular because of their simplicity. But there are still now different drawbacks present in different partition based clustering algorithms. As for example, partition based clustering algorithms like fuzzy k-means [143-146], possibilistic k-means [149] etc. work well on generation of overlapping clusters but main drawback of these clustering methods is that these methods work on whole dataset and are not capable to eliminate the noise. Apart from this, fuzzy k-means and possibilistic k-means clustering algorithms have also several other drawbacks due to their membership constraints [149] when run independently. As the main drawbacks of fuzzy-k-means is that different equidistant members may have different memberships values for the same cluster and it creates inconsistency in overall partitioning of the data while possibilistic k-means generates coincident clusters. On the other hand, k-means clustering algorithm always gives consistent results but cannot generate overlapping clusters.

In this regard, here, a new hybrid clustering framework using different k-means versions (HCFPC) is proposed to find clusters of relevant co-expressed genes via eliminating noise from highly noisy microarray data. This is the first partition-based clustering framework where we have tried to eliminate all those drawbacks of different k-means versions in a novel manner. Experimental results show the efficiency of the proposed hybrid framework.

5.2 Related Work

5.2.1 Hard k-means

Let $X = \{x_1, x_2, \dots \dots ., x_i, \dots \dots x_n\}$ be the set of n objects. Let k is the total number of clusters and $V = \{v_1, v_2, \dots\dots\dots, v_j, \dots\dots v_k\}$ is the set of k centres/means. $\beta = \{\beta_1, \beta_2, \dots\dots., \beta_j, \dots\dots \beta_k\}$ is the set of k clusters. This algorithm partitions X into k-clusters by minimizing the following objective function:

$$J = \sum_{i=1}^{n} \sum_{j=1}^{k} ||x_i - v_j||^2 \qquad (5.1)$$

Where v_j is the jth centres of corresponding cluster β_j and every centre is updated using the following equation:

$$v_j = \frac{1}{|\beta_j|} \sum_{i=1 \&\& x_i \in \beta_j}^{n} x_i \qquad (5.2)$$

Algorithm: Hard_*k*-means_Clustering

Input: A dataset X with n objects

Output: A set of clusters β

Step 1. Read k.

Step 2. Set iteration counter $c = 1$.

Step 3. Randomly choose k objects from X as initial cluster centres.

Step 4. Assign each object to the cluster to which the object is the most similar, based on the distance of that object from the corresponding cluster centre.

Step 5. Update all cluster centres using equation 5.2 i.e., calculate the mean value of the objects for each cluster.

Step 6. If $V^{c+1} = V^{c}$, then increment c and goto Step 4 otherwise goto Step 7.

Step 7. End

5.2.2 Fuzzy *k*-means

Let $X = \{x_1, x_2, \dots\dots., x_i, \dots\dots x_n\}$ be the set of n objects. Let k is the total number of clusters and $V = \{v_1, v_2, \dots\dots\dots, v_j, \dots\dots v_k\}$ is the set of k centres/means. $\beta = \{\beta_1, \beta_2, \dots\dots., \beta_j, \dots\dots \beta_k\}$ is the set of k clusters. This algorithm partitions X into k-clusters by minimizing the following objective function:

$$J = \sum_{i=1}^{n} \sum_{j=1}^{k} (\mu_{ij})^{m_1} ||x_i - v_j||^2 \qquad (5.3)$$

Where $1 < m_1 < \alpha,$ is the fuzzifier, v_j is the jth centres of corresponding cluster β_j. Here $W_{n \times k} = \{\mu_{ij}\}$ is a membership matrix, in which every row represents membership values of a x_i object for all k clusters. $\mu_{ij} \in [0, 1]$ is the probabilistic membership value of object x_i for the corresponding cluster β_j and it is calculated using the following equation:

$$\mu_{ij} = (\sum_{l=1}^{k}(\frac{d_{ij}}{d_{il}})^{\frac{2}{m_1-1}}),\ here\ \ d_{ij}^2 = ||x_i - v_j||^2 \quad (5.4)$$

The elements μ_{ij} of $W_{n\times k}$ satisfies the following conditions:

$$0 < \sum_{i=1}^{n}\mu_{ij} < n\ \ for\ all\ j\ \ and\ \sum_{j=1}^{k}\mu_{ij} \leq 1\ \ for\ all\ i \qquad (5.5)$$

And every centre is updated using the following equation:

$$v_j = \frac{1}{\gamma_j}\sum_{i=1}^{n}(\mu_{ij})^{m_1}\,x_i, \qquad here, \gamma_j = \sum_{i=1}^{n}(\mu_{ij})^{m_1} \qquad (5.6)$$

Algorithm : Fuzzy_k-means_Clustering

Input: A dataset X with n objects

Output: A set of clusters β

```
Step 1.  Read $k$ and $m_1$.
Step 2. Set iteration counter $c = 1$.
Step 3. Randomly choose $k$ objects from $X$ as initial cluster centres.
Step 4. Calculate membership matrix $W_{n\times k}^{c}$using equation 5.4.
Step 5. Update all cluster centres using equation 5.6.
Step 6. Calculate membership matrix $W_{n\times k}^{c+1}$ using equation 5.4.
Step 7. If $|\ W_{n\times k}^{c+1} - W_{n\times k}^{c}| >\in$, then increment c and goto Step 5 otherwise goto step 8.
Step 8. End
```

5.2.3 Possibilistic *k*-means

Let $X = \{x_1, x_2, \dots\dots., x_i, \dots\dots x_n\}$ be the set of n objects. Let k is the total number of clusters and $V = \{v_1, v_2, \dots\dots\dots, v_j, \dots\dots v_k\}$ is the set of k centres/means. $\beta = \{\beta_1, \beta_2, \dots\dots., \beta_j, \dots\dots \beta_k\}$ is the set of k clusters. This algorithm partitions X into k-clusters by minimizing the following objective function:

$$J = \sum_{i=1}^{n}\sum_{j=1}^{k}(\rho_{ij})^{m_2}||x_i - v_j||^2 + \eta_j\sum_{i=1}^{n}(1 - \rho_{ij})^{m_2} \qquad (5.7)$$

Where $1 < m_2 < \alpha,$ is the fuzzifier, v_j is the jth centres of corresponding cluster β_j and η_j are positive numbers and are defined as:

$$\eta_j = K.\frac{\sum_{i=1}^{n}(\rho_{ij})^{m_2} d_{ij}^2}{\sum_{i=1}^{n}(\rho_{ij})^{m_2}} \qquad (5.8)$$

Here $Z_{n\times k} = \{\rho_{ij}\}$ is a membership matrix, in which every row represents membership values of a x_i object for all k clusters. $\rho_{ij} \in [0,1]$ is the possibilistic membership value of object x_i for the corresponding cluster β_j and it is calculated using the following equation:

$$\rho_{ij} = \frac{1}{1+(\frac{d_{ij}^2}{\eta_j})^{\frac{1}{m_1-1}}} here \quad d_{ij}^2 = ||x_i - v_j||^2 \qquad (5.9)$$

The elements μ_{ij} of $W_{n\times k}$ satisfies the following conditions:

$$0 < \sum_{\mathrm{i=1}}^{\mathrm{n}} \rho_{ij} \leq \mathrm{n},\ for\ all\ j\ \ and \max\ \rho_{ij} > 0\ for\ all\ i \qquad (5.10)$$

And every centre is updated using the following equation:

$$v_j = \frac{1}{\gamma_j}\sum_{i=1}^{n}(\rho_{ij})^{m_2} x_i,\ here, \gamma_j = \sum_{i=1}^{n}(\rho_{ij})^{m_2} \qquad (5.11)$$

Algorithm: Possibilistic_*k*_means_Clustering

Input: A dataset X with n objects

Output: A set of clusters β

```
Step 1.  Read $k$ and $m_1$.
Step 2. Set iteration counter $c = 1$.
Step 3. Randomly choose $k$ objects from $X$ as initial cluster centres.
Step 4. Calculate membership matrix $Z_{n\times k}^{c}$using equation 5.9.
Step 5. Calculate $\eta_j\ for\ j = 1\ to\ k$ using equation 5.8
Step 6. Update all cluster centres using equation 5.11.
Step 6. Calculate membership matrix $Z_{n\times k}^{c+1}$ using equation 5.9.
Step 7. If $|\ Z_{n\times k}^{c+1} - Z_{n\times k}^{c}| > \in$, then increment c and goto Step 5 otherwise goto step 8.
Step 8. End
```

5.3. Proposed Method

Here, the proposed framework, is developed using different existing partition-based clustering algorithms. These are fuzzy k-means, possibilistic k-means, hard k-means. The proposed framework is described below.

Preliminaries

Let $G_{m\times n}$ is a data matrix (here, gene expression matrix) consisting of n number of genes and m number of samples. $X = \{X_1, X_2, \dots\dots., X_j, \dots\dots., X_n\}$ is the set of n row/genes/objects, $C = \{C_1, C_2, \dots\dots, C_i, \dots\dots, C_k\}$ is the set of k clusters and $V = \{V_1, V_2, \dots\dots., V_i, \dots..V_k\}$ is the set of k centres. $C^F = \{C_1^F, C_2^F, \dots., C_i^F, \dots, C_k^F\}$is the set of k clusters generated by fuzzy k-means clustering algorithm and $V = \{ V_1^F, V_2^F, \dots\dots., V_i^F, \dots.., V_k^F\}$ is the set of k centres generated by fuzzy k-means clustering algorithm. $C^P = \{C_1^P, C_2^P, \dots., C_q^P, \dots, C_k^P\}$is the set of k clusters generated by possibilistic$k$-means clustering algorithm and $V = \{ V_1^P, V_2^P, \dots\dots., V_q^P, \dots.., V_k^P\}$ is the set of k centres generated by possibilistic$k$-means clustering algorithm. $C^P = \{C_1^P, C_2^P, \dots., C_q^P, \dots, C_k^P\}$is the set of k clusters generated by possibilistic k-means clustering algorithm and $V = \{ V_1^P, V_2^P, \dots\dots., V_q^P, \dots.., V_k^P\}$ is the set of k centres generated by possibilistic k-means clustering algorithm. $M = \{M_1, M_2, \dots\dots, M_i, \dots\dots., M_k\}$is the set of merged clusters.

HCFPC : A New Hybrid Clustering Framework Using Partition Based Clustering Algorithms

Here, in the proposed framework, at first, the number of clusters k and corresponding cluster centres are selected automatically using a procedure named Initial_Centre_Selection procedure. The proposed HCFPC procedure is shown in Figure 5.1. After execution of $Initial\_Centre\_Selection$ procedure, some objects are marked whose maximum similarity with respect to all objects is less than α. Then possibilistic clustering algorithm is applied to cluster objects considering already selected k number of objects as initial centres. After clustering, the objects which have membership value less than β with respect to all clusters, where β is a user defined threshold, and if they are already marked then these objects are considered as noise points and deleted from the object set. Then again from the remaining objects, using procedure $Initial\_Centre\_Selection$, k number of objects are selected as initial centres. Using these k centres, fuzzy k-means, hard k-means, and possibilistic k-means run parallelly on the reduced object set and three set of clusters C^F, C^H, C^P are generated. Now for every cluster C_i^F , $1 \le i \le k,$ present in C^F, its similarity is checked with every cluster C_j^H,$1 \le j \le k$, present in C^H using a cluster-cluster similarity measure (described later) and the cluster (let C_l^H) for which similarity of $C_i^F$is maximum is chosen from C^Hset. Similar way similarity of C_i^F is also checked with every cluster $C_q^P, 1 \le q \le k,$ present in C^P and the cluster (let C_s^P) for which similarity of $C_i^F$is maximum is chosen from C^Pset. The cluster-cluster similarity measure is described below:

Let C_u and C_v are two clusters: The similarity between two clusters is :

$$CCS(C_u, C_v) = \frac{C_u \cap C_v}{C_u \cup C_v} \qquad (5.12)$$

If C_u and C_v, have no match then their similarity value will be 0 and if they are perfectly matched with one another then their similarity value will be 1. So, $CCS(C_u, C_v)$ varies between 0 and 1.

Now from these three clusters, C_i^F , C_l^H, C_s^P the common objects are taken and a merged cluster M_i is formed. The objects which are not common in these three clusters are placed in W set. This process is repeated and in this way $M = \{M_1, M_2, \dots \dots, M_i, \dots \dots., M_k\}$ merged clusters are formed and their centre are calculated. Now from the non-common object set W, one is taken and is placed in the corresponding merged cluster to which it is closest. This procedure is repeated for all object present in W set. In this way merged clusters are finally modified.

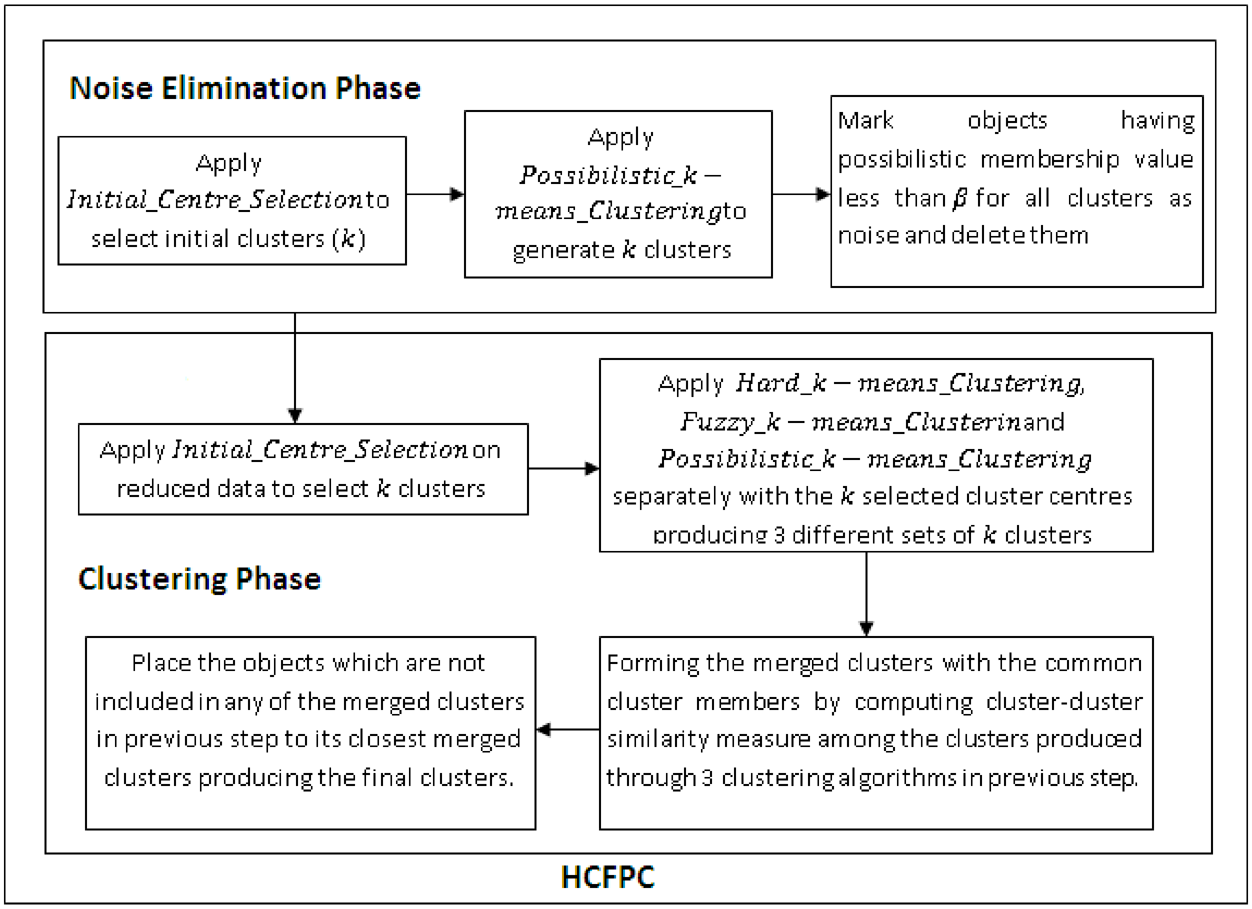

Noise Elimination Phase
Apply $Initial\_Centre\_Selection$ to select initial clusters (k)
Apply $Possibilistic\_k-means\_Clustering$ to generate k clusters
Mark objects having possibilistic membership value less than β for all clusters as noise and delete them
Apply $Initial\_Centre\_Selection$ on reduced data to select k clusters
Apply $Hard\_k-means\_Clustering$, $Fuzzy\_k-means\_Clusterin$ and $Possibilistic\_k-means\_Clustering$ separately with the k selected cluster centres producing 3 different sets of k clusters
Clustering Phase
Place the objects which are not included in any of the merged clusters in previous step to its closest merged clusters producing the final clusters.
Forming the merged clusters with the common cluster members by computing cluster-cluster similarity measure among the clusters produced through 3 clustering algorithms in previous step.
HCFPC

Figure 5.1 Block diagram of HCFPC Framework

Algorithm: HCFPC

Input:$G_{m \times n}$ is a gene expression matrix consisting of n number of genes and m number of samples. $X = \{X_1, X_2, \dots \dots., X_j, \dots \dots., X_n\}$is the set of n rows/genes/objects of G.

Output: A set M of Clusters.

Step 1. Call $Initial\_Centre\_Selection$ procedure on X set and generate k-number of cluster centres and mark the objects whose maximum similarity with respect to other objects is less than α.

Step 2. Call $Possibilistic\_k - means\_Clustering$ procedure (with skipping step of random initial centre selection) and generate k-number of clusters.

Step 3. Delete those objects from the set of objects X, which are already marked and whose possibilistic membership value with respect to all clusters less than β and form a reduced set of objects.

Step 4.Call $Initial\_Centre\_Selection$ procedure on reduced object set and generate k-number of cluster centres.

Step 5. Call $Fuzzy_k - means_Clusterin$g procedure (with skipping step of random initial centre selection), $Possibilistic\_k - means\_Clustering$ procedure (with skipping step of random initial centre selection), $Hard\_k - means\_Clustering$ procedure (with skipping step of random initial centre selection) on the reduced set of objects independently and generate three set of clusters C^F, C^P, C^H.

Step 6. Repeat step 7 to step 11 for every cluster C_i^F , $1 \leq i \leq k,$ present in C^Fdo:

Step 7. Calculate cluster-cluster similarity using CCS measure between C_i^F and every cluster $C_j^H \in C^H$.

Step 8. Select the cluster (let C_l^H) from C^H which has maximum similarity with C_i^F.

Step 9. Calculate cluster-cluster similarity using CCS measure between C_i^F and every cluster $C_q^P \in C^P$.

Step 10. Select the cluster (let C_S^P) from C^P which has maximum similarity with C_i^F.

Step 11. Take the common objects from these three clusters, C_i^F , C_l^H, C_s^P and form a merged cluster M_i and place the objects which are not common in these three clusters in W set.

Step 12. Repeat step 13 for every object present in the set W:

Step 13. Place the object in the corresponding merged cluster to which it is closest.

Step 14. End

Initial_Centre_Selection: Proposed Procedure for Initial Cluster centres Selection

The proposed procedure is developed based on a neighborhood formation criterion. The neighbor formation criterion is described first and then the proposed procedure is elaborated.

New Neighbor selection criteria: An object X_q, is in neighborhood of another object X_p or X_p is in neighborhood of X_q, if and only if

$$Neighbour(X_p, X_q) = Neighbour(X_q, X_p) = \begin{cases} 1, if\ Sim(X_p, X_q) \geq \delta \\ 0, otherwise \end{cases} \quad - (5.13)$$

Where, δ is a user defined threshold and $Sim(X_p, X_q)$ is defined below:

$$Sim(X_p, X_q) = 1 - \frac{EU\_dis(X_p, X_q)}{MAX\_EU\_dis} - - - - - - - - (5.14)$$

If two objects are similar then similarity between them is maximum, that is, $Sim(X_p, X_q) = 1$, Hence, $0 \leq Sim(X_p, X_q) \leq 1$. Also, $Sim(X_p, X_q) = Sim(X_q, X_p)$.

Algorithm :Initial_Centre_Selection

To create initial cluster centres, the proposed procedure is described below:

Step 1. For every object $X_j \in X$, using the proposed neighbor selection criterion the neighbors are selected and number of neighbors of every object X_j i.e. $CN(X_j)$is calculated, where,

$$CN(X_j) = \sum_{i=1\&\&\, i \neq j}^{n} Neighbour(X_j, X_i) - - - - - - (5.15)$$

Step 2. Sort n objects in descending order according to their CN values.

Step 3. The objects whose maximum similarity value from other objects are less than α, are marked. Here α is a user defined threshold.

Step 4. Set i to 0

Step 5. Repeat step 6 to Step 7 for all non-marked object present in X

Step 6. Select the object (let X_p) with highest CN value as an initial cluster centre of C_i and delete all objects from set X which are neighbors of X_p.

Step 7. Set $i \leftarrow i + 1$

Step 8. End

After selection of c number of initial cluster centres, using the above-mentioned procedure, Possibilistic clustering algorithm is applied considering those centres as initial cluster centres.

5.4. Results

In this research work, the performance of HCFPC is compared with that of hard k-means (HKM) [17-19], fuzzy k-means (FKM) [143-145], possibilistic k-means (PKM) [148], cluster identification via connectivity kernels (CLICK) [146], and self-organizing map (SOM) [149] on different microarray gene expression data sets (with noise and without noise version). The Silhouette index (SILH) [150], Davies-Bouldin index (DB) [151], and Dunn index (DUNN) [151] are the three main metrics for assessing the performance of various algorithms. Additionally, the Gene Ontology Term Finder is used to examine the biological significance of the HCFPC algorithm's created gene clusters [152].

In this paper, four publicly available microarray times series gene expression datasets are taken for making comparative study. The description of the datasets is given in Table 1, it can be found at http://www.ncbi.nlm.nih.gov/geo/, the Gene Expression Omnibus.

Dataset Preparation:

Four datasets have been used here for study. Among these, GDS2002 and GDS2003 datasets required no pre-processing while GDS1116 and GDS2910 required pre-processing. Initially all rows containing missing values greater than 10% are deleted. The missing values are then filled with row-average values to get the complete data. After that artificially noise has been introduced. Among those deleted rows, rows with missing values above 75% are selected and those missing values are filled with random values outside the range of possible values (i.e. greater than the maximum possible value and less than the minimum possible value) in previously created complete data. These artificially created rows are then appended with the complete data to create the noisy datasets with 15 to 20% of noise elements.

Table 5.1. Dataset Description

Dataset Name	**Species S**	**Number of Rows/Genes**	**Number of Columns/Time-Points**
GDS2910(Noisy)	Yeast	2746(1900+846)	191
GDS1116(Noisy)	Yeast	1081(798+281)	131
GDS2002	Yeast	5617	30
GDS2003	Yeast	5617	30

The parameters settings for the proposed clustering framework HCFPC are given below.

Table 5.2. Parameters Settings

Noise Parameter MAX CUT(α)	0.5
Neighbor Parameter(*ä*)	0.8
Membership Threshold for Possibilistic Noise Filter(*β)*	0.01
MAX Iteration No.	100
MAX Cluster No.	30
Convergence Parameter	0.01

Table 5.3. Performance Comparison of the Proposed Framework with HKM, FKM, and PKM

DATASET NAME	CLUSTERING METHODS	CLUSTER NO.	DB INDEX	SILHOUTTE INDEX	DUNN INDEX
GDS2910	HCFPC framework	30	1.49	0.08	0.06
	HKM		2.47	0.05	0.049
	FKM		6.13	-0.06	0.045
	PKM		6.29	-0.17	0.037
GDS1116	HCFPC framework	5	1.91	0.12	0.042
	HKM		1.98	0.09	0.047
	FKM		3.97	-0.11	0.032
	PKM		4.65	-0.13	0.036

To show the superiority of the proposed HCFPC framework, it is compared with hard k-means, fuzzy k-means and possibilistic k-means for GDS2910 and GDS1116 data set (normal datasets no additional noise is inserted here). For every dataset the number of gene clusters k is decided by using the CLICK [146] algorithm. For HCFPC framework after generating all cluster representatives first k number of representatives are chosen which are generated by Initial_Centre_Selection() method used in HCFPC framework. For other algorithms k-number of cluster centres are generated randomly. The results are shown in Table 5.3. In all cases, the proposed framework shows its superiority.

The performance of the proposed HCFPC framework is compared with HKM, FKM, PKM in presence of inherent noise and additional noise for GDS2910 and GDS1116 datasets with respect to Davis-Buldin index (DB), Silhoute index(SILH), Dunn index(DUNN) and all the results are shown in Table 4. In every case, the proposed framework shows its superiority. It has been also found that in presence of additional noise HCPFC framework gives significant better results than other partition-based clustering algorithms.

Table 5.4 Performance Comparison of the Proposed Framework with HKM, FKM, and PKM in presence of Noise

Dataset	Evaluation Metric	HCFPC framework		HKM		FKM		PKM	
		Normal dataset with inherent noise	With noise	Normal dataset with inherent noise	With noise	Normal dataset with inherent noise	With noise	Normal dataset with inherent noise	With noise
GDS2910	DVB	1.49	1.55	2.47	2.76	6.13	8.43	6.29	12.54
	SILH	0.08	0.04	0.05	0.01	-0.06	-0.21	-0.17	-0.16
	DUNN	0.06	0.076	0.049	0.52	0.045	0.46	0.037	0.32
GDS1116	DVB	1.49	1.54	1.98	2.03	3.97	3.1	4.65	14.86
	SILH	0.12	0.11	0.09	0.13	-0.11	0.71	-0.13	-0.25
	DUNN	0.042	0.043	0.047	0.37	0.032	0.43	0.036	0.47

Table 5. 5 Performance Comparison of the Proposed Framework with other Clustering Methods

Indices	Clustering Algorithms	GDS2002	GDS2003
		Normal dataset	Normal dataset
DB Index	CLICK	26.7	17.61
	SOM	13.41	15.22
	HCFPC framework	0.17	0.19
Silhouette Index	CLICK	-0.12	-0.09
	SOM	-0.05	-0.06
	HCFPC framework	0.89	0.93
Dunn Index	CLICK	0.03	0.05
	SOM	0	0.01
	HCFPC framework	4.1	2.43

In order to establish the superiority of the proposed framework over two other category based existing gene clustering algorithms, namely, CLICK [146] and SOM [149], results are shown for GDS2002 and GDS2003 yeast microarray data sets. These results are obtained from literature survey[131]. Table 5 presents the comparative assessment of these three clustering algorithms, in terms of Silhouette index, DB index, Dunn index, From the results reported in this table, it can be seen that the proposed framework performs significantly better than both CLICK and SOM.

Table 5.6. Significant GO Terms Obtained Using Proposed Algorithm for GDS2003

Ontology Aspects	**Cluster Number**	**Gene Ontology term**	**Cluster frequency**	**Genome frequency**	**Corrected P-value**	**FDR**	**FALSE Positives**
Biological Process	12	cytoplasmic translation	0.357	0.029	3.25E-117	0.00%	0
	3	mitochondrial translation	0.284	0.024	5.39E-73	0.00%	0
	18	ribosome biogenesis	0.538	0.067	3.87E-72	0.00%	0
	18	ribonucleoprotein complex biogenesis	0.543	0.08	1.15E-64	0.00%	0
	18	rRNA metabolic process	0.431	0.054	9.40E-55	0.00%	0
	12	peptide biosynthetic process	0.435	0.114	6.06E-54	0.00%	0
	12	organonitrogen compound biosynthetic process	0.551	0.187	2.85E-53	0.00%	0
	3	mitochondrion organization	0.284	0.04	3.02E-50	0.00%	0
Molecular Function	12	structural constituent of ribosome	0.315	0.033	4.86E-85	0.00%	0
	3	structural constituent of ribosome	0.219	0.006	4.15E-36	0.00%	0
	9	electron transfer activity	0.235	0.025	2.24E-20	0.00%	0
	9	active transmembrane transporter activity	0.324	0.025	1.56E-17	0.00%	0
	18	snoRNA binding	0.086	0.04	4.65E-17	0.00%	0
Cellular Component	3	mitochondrion	0.781	0.174	3.79E-125	0.00%	0
	12	cytosolic ribosome	0.329	0.023	3.36E-119	0.00%	0
	3	mitochondrial protein-containing complex	0.34	0.03	1.51E-87	0.00%	0
	12	ribonucleoprotein complex	0.463	0.091	1.23E-79	0.00%	0
	18	preribosome	0.35	0.024	8.68E-64	0.00%	0
	12	intracellular non-membrane-bounded organelle	0.581	0.21	1.72E-53	0.00%	0

Biological Significance of Gene Cluster

Table 5.6.represents the information about some biological significant gene clusters in terms of Molecular function (MF), Biological process (BP) and Cellular component (CC) ontological terms for GDS2003 dataset generated by the GO Term Finder [152] tool via applying it gene clusters produced by HCFPC algorithm. Gene clusters are deemed significant clusters if their p-values are less than 0.05 for every given cluster of genes.

5.5. Discussion

From the results-based data analysis, it has been found that the proposed HCFPC framework partitions gene expression datasets in presence of inherent noise and also after incorporating additional noise into good clusters compared to other partition-based clustering algorithms in terms of different indices. The proposed framework also generates better clusters than other category-based clustering algorithms like SOM and CLICK for normal datasets. Apart from this, from Table 6, it can be written that the proposed clustering framework generates qualitative biologically significant clusters.

5.6. Conclusion

In this paper, a new gene clustering framework is developed by integrating different partition-based clustering algorithms in a novel manner. The main novelty of this framework is that, it can work in presence of noise and after detecting noisy genes, it can eliminate them and generates good qualitative clusters with small set of significant genes. Apart from this instead of random centroid selection it selects centroid in a novel manner.

The proposed framework with the above-mentioned features performs significantly better than other partition-based clustering algorithms and other types of clustering algorithms for all microarray datasets (in presence of additional noise also) in terms of different quantitative indices and also provides biologically significant clusters.

In the next chapter we have proposed a new ensemble machine learning model for cancer sample classification from gene expression data.

Chapter 6

An Ensemble Machine Learning Model based on Multiple Filtering and Supervised Attribute Clustering Algorithm for Classifying Cancer Samples

6.1 Introduction

Cancer is one of the most fatal diseases around the globe [153, 154]. According to the World Health Organization report, Cancer is marked as the second most deadly disease and an estimated 9.7 million deaths around the world in 2018 have occurred due to this signature disease [154]. Generally, one in every 6 deaths all over the world, occurs due to cancer. So, within 2030, the number of new cancer patients per year will increase approximately by 25 million [154, 155]. Although several advanced techniques are already developed for the detection of cancer, the proper prognosis of cancer patients, till date, is very poor and the survival rate is also very low [153, 154, 156]. It has been already found that for very accurate cancer sample classification or prediction, adequate information is not available from the clinical, environmental, and behavioral characteristics of patients [153, 154, 156]. Recently, due to different types of bio-molecular data analysis, several genetic disorders with different biological characteristics have been revealed which are very helpful for early identification and prognosis of cancer and also to discern the responses for different types of treatment [157-165].

With the rapid advancements in genomic, proteomic, and imaging high-throughput technologies [157-165], now it is possible to accumulate huge amount (in the order of thousands) of different bio-molecular information of patients. Using this huge amount of information, researchers have been trying to develop more advanced techniques for early detection and proper prognosis of cancer, and also to improve cancer therapy for improvement of patients' survival rate. To analyze this huge amount of information, lab-based approaches are not adequate as these methods are costly and time-consuming. So, computational or in-silico methods like statistical methods, machine learning, deep learning, etc. have been being used extensively in this field.

It is well-known fact that in cancer-causing cells, gene expression is either overexpressed or under expressed [153]. So, measurement of gene expression in cancer cells can give adequate information to improve cancer diagnostic procedures. Nowadays, different developing countries have been using this procedure for cancer sample detection. It is already known that using DNA microarray technology it is possible to measure the expression level of a numerous number of genes for a single experiment/sample simultaneously. The outcome of DNA microarray technology is a gene expression data matrix. This matrix carries information about the expression level of a huge number of genes for a limited number of samples (such as diseased patient samples and normal samples). The presence of the limited number of samples in this data matrix is due to the lack of availability of samples. So, based on information of gene expression data matrix, cancer sample classification is one of the essential tasks in the field of cancer research [165-171].

Using computational or in-silico approaches, gene expression-based cancer sample classification task has been reviewed extensively in different papers [165-171]. However, the main difficulties in the sample classification task arise due to several factors. Firstly, in these data sets, a substantially small number of samples is available (generally in the order of hundreds) compared to the availability of a huge number of genes (generally in the order of thousands) [165-171]. For sample classification, genes are treated as features/attributes. So, the high-dimensional gene space is an overhead for most classification algorithms. Secondly, only a very few genes are informative (differentially expressed) and the rest of the section is non-informative (noisy) [165-171] for sample classification and responsible for degrading the classifier's performance. Gene dimension reduction by identification of informative genes as biomarkers can improve the classification accuracy of classifiers. Apart from the improvement of classification accuracy, the identification of informative biomarkers (here, informative genes) has great prospects from a biomedical point of view. These are beneficial for finding the biological reason for a disorder, assessing disease risk, and developing therapeutic targets. The third problem arises due to the small sample size which creates an overfitting problem in classifier construction. Another problem that degrades classifier performance is the sample class imbalance problem. This problem occurs due to the presence of more instances/samples of one class (majority class) with respect to other class(es) (minority class) in a dataset.

A fairly large number of works have been already developed for sample classification. These works are divided into two categories. In the first category [153, 165-171], the major emphasis is given to the selection of relevant genes for the reduction of feature space. Then based on this reduced feature space, predictive/classification accuracy of the samples is measured using different existing single classification models like naïve Bayes, support vector machine, relevance vector machine, K-nearest neighbor, decision tree, logistic regression, etc. As gene selection is a feature selection task, so based on feature selection techniques, these methods are divided into different categories. These are (1) filter methods (2) wrapper methods (3) embedded methods and (4) hybrid methods. Before we mention the second category of classification methods, let us first elaborate on the first category methods one by one.

Filter methods [153, 165-171] select a subset of features without taking any information from any classification model. These methods select features that are differentially expressed with respect to sample class labels. The filter methods rank individual features according to their class discrimination power based on some statistical score function and then select a number of high-ranked features to form a reduced and relevant feature subset. The popular statistical score functions used in filter methods are Fisher's score, Signal to Noise ratio (SNR), correlation coefficient, mutual information, Relief [172], etc. Filter methods are computationally simple, fast, and unbiased in favor of any specific classifier as these methods do not consider any knowledge from any classifier at the feature selection phase. The drawback of filter methods is that the number of selected features is based solely on the trial-error method.

Wrapper methods [153, 165-171], on the other hand, judge discrimination capability of a feature subset using classification error rate or prediction accuracy of a classifier as the feature evaluation function. It selects the most discriminative feature subset via minimizing the classification error rate or maximizing the classification accuracy of a classifier. The wrapper methods generally achieve better classification accuracy than the filter methods because the selection of feature subset is classifier-dependent. One drawback of these methods is that these are biased to used classifiers and another drawback is that these are computationally more expensive than the filter methods as generation of the best feature subset for the high-dimensional dataset is an NP-complete problem. Due to these reasons, these methods are not applicable for high-dimensional datasets.

In Embedded methods [153, 165-171], the optimal feature subset is selected through the unique learning procedure of a specific classifier at the time of classifier construction. Actually, in these methods, the optimal feature subset selection part is embedded as part of classifier construction. These methods are faster than wrapper methods but are biased to the specific classifier. In embedded approaches, the feature selection process is specific for a particular classifier and is not applicable to other classifiers. These are also computationally expensive. Due to these reasons for high-dimensional datasets, these methods are not applicable. On the other hand, recently hybrid feature selection methods [153, 165-171] are also developed. In hybrid methods, different category-based methods are combined to take advantage of all of these methods for improving classification accuracy.

Apart from these methods, clustering techniques [153, 165-171] are also used for feature selection purposes. Clustering techniques divide the data space in such a manner that objects in the same cluster are similar while in different clusters they are dissimilar. For the feature selection task, clustering methods (famous as attribute clustering in feature selection domain) (Au 2005) divide the features into several distinct clusters and then reduce the feature dimension by selecting a small number of significant features from each cluster. A lot of unsupervised gene (attribute) clustering algorithms (Au2005, Chin 2016, Hambali 2020) are already developed for this task. However, these methods are unsuccessful to find informative functional groups of genes for sample classification as in clustering genes, no supervised information from sample classes is considered [165-171, 173]. So, scientists have developed a number of supervised gene (attribute) clustering algorithms [174-177] in which genes are grouped using supervised information from sample classes and a reduced gene set is formed via selecting the most informative genes from each cluster.

All the above mentioned variants deliver comparable feature selection and classification accuracy. Quite often this type of classification models with only a few genes and with a limited number of training samples can classify the majority of training samples correctly, but the generalization capability of such classification models cannot be guaranteed [178-183]. So, the most important task for a medical diagnosis system is to improve the classification accuracy of unknown samples (generalization performance) which cannot be solved by this type of classification model.

Apart from this problem, the microarray data is related to several uncertainties due to fabrication, hybridization, and image processing procedure in microarray technology. These uncertainties introduce various types of noise in microarray data. Due to the presence of these uncertainties with a limited number of training samples, the conventional machine learning approaches face challenges to develop reliable classification models.

To overcome the above mentioned problems, it is therefore essential to develop general approaches and robust methods. In this regard, researchers are motivated to develop the second category-based model. These are the different robust ensemble classification models [178-183] which can overcome small sample size problems and are capable of removing uncertainties of gene expression data.

Ensemble methods [184] are a class of machine learning technique which combines multiple base learning algorithms to produce one optimal predictive model. Ensemble classification model refers to a group of individual/base classifiers that are trained individually on the trained dataset in a supervised classification system and finally, an aggregation method is used to combine the decisions produced by the base classifiers. These ensemble classification models have the potential to alleviate the small sample size problem by applying multiple classification models on the same training data or on bootstrapped samples (sampling with replacement) of the training data to decrease the chance of overfitting in the training data. In this way, the training dataset is utilized more efficiently, and as a consequence, the generalization ability is improved.

Although different category-based ensemble classification models exist in the literature but these ensemble models are not capable of addressing all the above-mentioned problems (small sample size, high dimensional feature space, and sample class imbalance problem) related to microarray data.

In this regard, here a new Multiple Filtering and Supervised Attribute Clustering algorithm-based ensemble classification model named MFSAC-EC is proposed. In this model, first, a number of bootstrapped versions of the original training dataset are created. At the time of the creation of bootstrapped versions, an oversampling technique [185] is adopted to solve the class imbalance problem. For every bootstrapped dataset a number of sub-datasets (each with a subset of genes) are generated using the proposed MFSAC method. The MFSAC is a hybrid method combining multiple filters with a new supervised attribute clustering method. Then for every sub dataset, a base classifier is constructed. Finally, based on the prediction accuracy of all these base classifiers of all sub-datasets for all bootstrapped datasets an ensemble classifier (EC) is formed using the majority voting technique.

The novelty of the proposed MFSAC-EC model is that here the emphasis is given simultaneously on the high dimensionality problem of gene expression data, small sample size problem as well as the class imbalance problem. All of these problems at the same time are not considered in any existing ensemble classification model. First of all, due to the use of bootstrapping method with a class balancing strategy, the proposed model can handle a small sample size and overfitting problem. Secondly, in MFSAC, different filter methods are used with their unique characteristics. So, different characteristics-based relevant gene subsets are selected via different filters to form different sub-datasets from every bootstrapped dataset. Finally, every gene subset is modified using a supervised attribute clustering algorithm. In this way, the high-dimensionality problem of gene expression data is handled here. Apart from this, from the

MFSAC generated sub-datasets, the frequency of occurrence is counted for every gene and informative genes are ranked accordingly. The prediction capability of the proposed model is experimented with over different microarray datasets and compared with the existing well-known models. Experimental outputs demonstrate the superiority of the proposed model over existing models.

6.2 Proposed Work

The proposed MFSAC-EC model is composed of different filter score functions, a new supervised attribute clustering method, and an ensemble classification method. In the following subsections, first, a brief overview is given on different filter score functions and then the proposed MFSAC-EC model is described.

Preliminaries

In this paper, a data set (here, a microarray gene expression data set) is represented by a data matrix,$K_{U\times V}$, with U data objects (samples) and V features (genes). The set of objects or samples is represented as $\mathrm{E} = \{E_1, E_2, \ldots\ldots, E_s, \ldots E_U\}$ while the set of genes is represented as $G = \{G_1, G_2, \ldots\ldots, G_t, \ldots. G_V\}$. Here, each sample is a V-dimensional feature vector containing V number of gene expression values. Similar way, every gene is a U-dimensional vector containing U number of sample values. Here, $C_{U\times 1}$is a class vector representing the associated class label for every sample. The class label is taken from a set $DC = \{d_1, d_2, \ldots\ldots, d_j, \ldots d_N\}$ with N distinct class labels.

Brief overview of Filter score functions used in MFSAC

The filter score functions used in the proposed MFSAC-EC model are modified Fisher score [186], modified T-test [187], Chi-square [172], Mutual information [172], Pearson correlation

Table 6.1. Description of Different Filter Methods

Filter Method	Definition	Description
Modified Fisher score (Gu 2011)	$F(G_t) = \frac{\sum_{j=1}^{N} n_j \left(m_t^j - m_t\right)^2}{(\sigma_t)^2}$ (1)	Here modified Fisher score of feature G_t is presented. Here n_j represents the number of samples of class j, m_t^j and σ_t^j are the mean and standard deviation of t th feature for j th class respectively, while m_t and σ_t are the mean and standard deviation of t th feature, $where\ (\sigma_t)^2 = \sum_{j=1}^{N} \left(\sigma_t^j\right)^2$.
Modified T-test (Zhou 2007)	$\mathrm{t}(G_t) = \max\{ \frac{m_t^j - m_t}{M_j S_t}, j = 1, \ldots., N$ (2)	Here modified t-test of feature G_t is presented. Here, m_t^j and m_t are the mean of tth feature for jth class and mean of tth feature respectively. S_t is the sum of within class standard deviation and is represented as, $S_t^2 = \frac{1}{U-N} \sum_{j=1}^{N} \sum_{l \in j} \left(x_{lt} - m_t^j\right)^2,$ While $M_j = sqrt(\frac{1}{n_j} + \frac{1}{U})$, n_j is the number of samples for jth class, $x_{lt}$is lth sample's value for t th feature in jth class. Here U and N are total number of samples and total number of classes respectively.
Chi Square (Das 2019)	$\chi^2(G_t, C) = \sum_{l \in L} \sum_{k=1}^{N} \frac{\left((observed(l, d_j) - expected(l, d_j)\right)^2}{expected(l, d_j)}$ (3)	Chi Square is a statistical test which is commonly used to compare observed data with the expected data according to a specific hypothesis. This test is also used as a measure to find class discrimination capability of a gene with respect to class vector. The chi square value of every gene vector is calculated in equation (3). Here, L is the set of all distinct values present in a gene, if the gene is discretized. $observed(l, d_j)$and$expected(l, d_j)$ represent the number of observed and expected co-occurrence of value l and d_j appeared in gene G_t and class vector C respectively.
Mutual Information (Das 2019)	$MInfo(G_t, C) = \sum_{d_k \in DC} \sum_{l \in L} P(l, d_j) log \frac{P(l, d_j)}{(P(l)P(d_j))}$ (4) Mutual information is also represented using entropy measure shown in equation (5). $MInfo(G_t, C) = H(G_t) - H(G_t \mid C)$ (5)	Mutual information is an important measure based on information theory. Mutual information between a gene and a class vector can be defined as in equation (4). Here, $H(G_t)$is the entropy or amount of information present inG_t and $H(G_t \mid C)$ is the entropy or amount of information present in G_t in presence of C. Here, L is the set of all distinct values present in a gene vector, if the gene vector is discretized.
Relief-F (Das 2019)	$RE(G_t) = \mathrm{RE}(G_t) - \sum_{m=1}^{K} \frac{diff(G_t, R_i, H_m)}{w.K} + \sum_{c_j \neq class(R_i)} \left[\frac{P(c_j)}{1 - P(class(R_i))} \sum_{m=1}^{K} diff(G_t, R_i, M_m)\right] \Big/ w.K$ (6)	Relief-F measures class discrimination power of a feature according to its distinguishing capability between near instances. For any random instance R_i, ReliefF finds the nearest instance from same class to find a hit (H) and a miss (M) from different class and according to that $RE(G_t)$ is increased or decreased using equation (6). This process is run for w number of times for different instances.
Pearson Correlation Coefficient (Leung 2010)	$\rho(\mathrm{x}, \mathrm{y}) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 (y_i - \bar{y})^2}}$ (7)	Pearson Correlation is used for detecting the linear relationship between two vectors. The equation (7) is used to calculate the PC (ρ) between the independent vector x and dependent vector y. For this paper x is considered as G_t and y as class vector C.
Signal-to-noise ratio (Leung 2010)	$S/R(G_t) = \frac{\sum_{j=1}^{N} (m_j^t - m_t)}{\sigma_t}$ (8)	The signal to noise ratio test is a feature selection method that selects significant features according to their expression levels using S/R test. Here m_t^j and σ_t^j are the mean and standard deviation of t th feature for j th class respectively, while m_t and σ_t are the mean and standard deviation of t th feature, $where\ \sigma_t = \sum_{j=1}^{N} \sigma_t^j$

coefficient [188], SNR [188], and Relief-F [172]. A summary of these 7 filters used in the MFSAC-EC model is given in the Table 6.1

Proposed MFSAC-EC Model

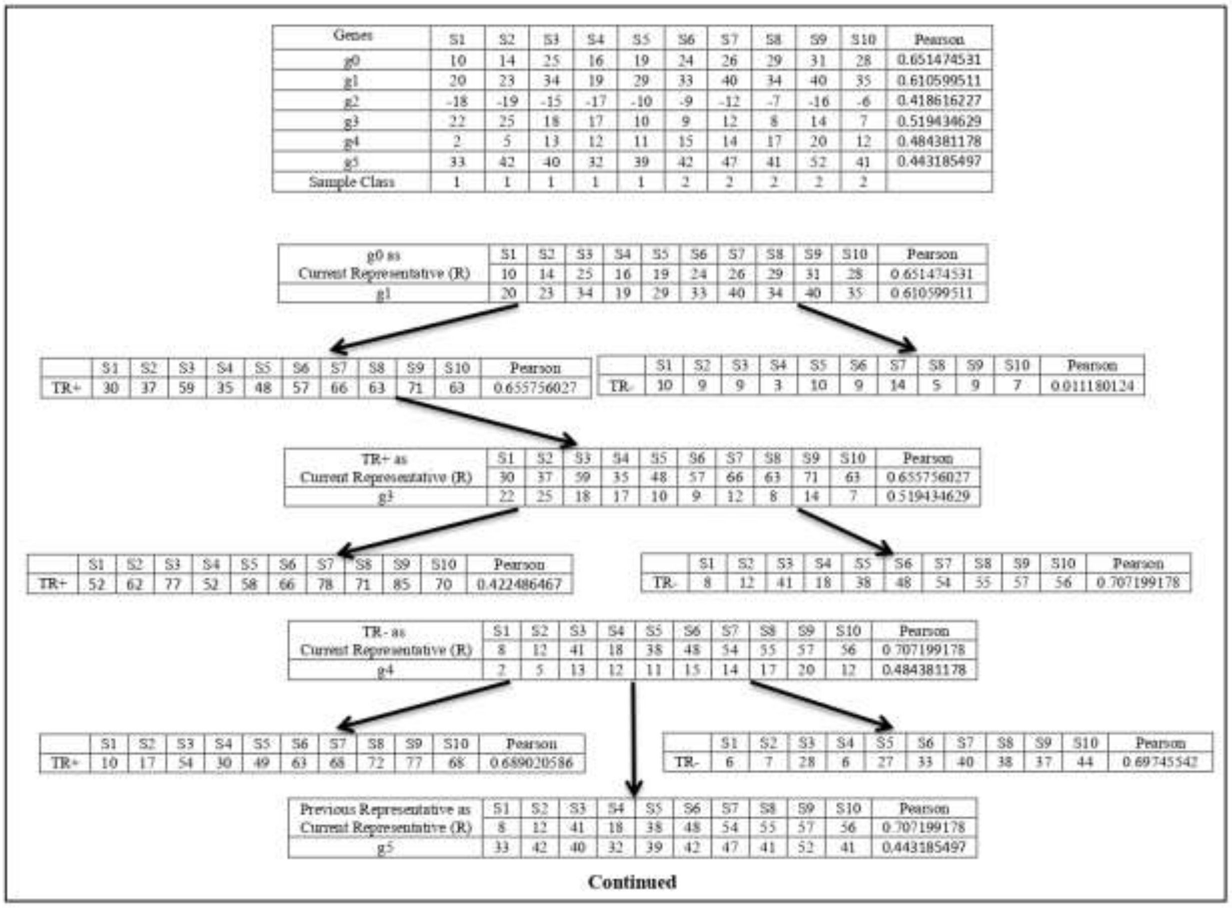
In the proposed MFSAC-EC model, initially, bootstrapping (sampling with replacements) with a class balancing procedure of samples is applied on training dataset K to create D number of different bootstrapped versions from the training dataset. Here, every bootstrapped dataset with U samples is formed by random sampling with replacements U times from the original dataset K. After that oversampling procedure is applied to each minority class to achieve data balance.

Oversampling consists of increasing the minority class instances by their random replication to exactly balance the cardinality of the minority and majority classes in each bootstrapped dataset. Due to oversampling each bootstrapped dataset will contain more instances than the original dataset.

The MFSAC method of the MFSAC-EC model, which is an integration of multiple filters and a new supervised attribute (gene) clustering method, is applied on every newly created bootstrapped (BK_l) training dataset. The proposed MFSAC method first calculates the class relevance score of every gene present in the bootstrapped training dataset using each filter score function (FT_x), $x = 1\ to$ 7 mentioned above. Then for each filter score function, a sub dataset SD_{lx} with a gene subset (GS_{lx}) is created by selecting a predefined number (let P) of the most relevant genes from the full gene set G. So, $|GS_{lx}| = P$. After that on every gene subset (GS_{lx}) of every sub dataset SD_{lx}, the SAC (Supervised Attribute Clustering) method is applied and a set of clusters CGS_{lx} and corresponding cluster representatives (considered as modified features) are formed. Finally, Q numbers of most relevant cluster representatives are selected as modified features and a reduced sub dataset RSD_{lx} of the sub dataset $SD_{lx}$is formed. How the SAC method works on $GS_{lx}$of every sub dataset $SD_{lx}$is discussed below.

For any sub dataset SD_{lx}, the SAC method starts by selecting the gene from the subset (GS_{lx}) with the highest FT_x value. Let gene$G_{li} \in GS_{lx}$with the highest FT_x value be selected as the first member (let $FT_x(G_{li}, C) = A$) and it also becomes the initial cluster representative $R(R = G_{li})$) of the first cluster C_1GS_{lx} and $G_{li}$is deleted from GS_{lx}. In effect, $G_{li} \in C_1GS_{lx}$, and $GS_{lx} = GS_{lx} - \{G_{li}\}$and so$FT_x(R, C) = A$. This cluster is then grown up in parallel with the cluster representative refinement process which is described next. In this process, the gene (let G_{lm}) with next highest FT_x value is taken from GS_{lx} subset and is merged with the current cluster representative R. The merging is done in two ways. Firstly the expression profile of $G_{lm}$is directly added with R and a temporary augmented representative TR^+ is formed and its FT_x value (let B_1) is calculated. The second one is that the sign-flipped value of the expression profile of G_{lm} is added with R and another temporary augmented representative TR^- is formed and its FT_xvalue (letB_2) is calculated. If $FT_x(TR^+, C) \geq FT_x(TR^-, C)$ that is $B_1 \geq B_2$ then TR^+ is chosen else TR^- is chosen. Let TR^+ is chosen. Now if $FT_x(TR^+, C) > FT_x(R, C)$then $R = TR^+$otherwise, R is unaltered. Similar way if TR^- is chosen and if

$FT_x(TR^-, C) > FT_x(R, C)$then $R = TR^-$otherwise, R remains unchanged. If R is modified then the gene$G_{lm}$is included in the cluster and G_{lm} is deleted from GS_{lx}. In effect, $G_{lm} \in C_1 GS_{lx}$, and $GS_{lx} = GS_{lx} - \{ G_{lm}\}$. So, the next chosen gene is included in the current cluster if it improves the class relevance value of the current cluster representative. The merging process is described in Figure 6.1.



Genes	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
g0	10	14	25	16	19	24	26	29	31	28	0.651474531
g1	20	23	34	19	29	33	40	34	40	35	0.610599511
g2	-18	-19	-15	-17	-10	-9	-12	-7	-16	-6	0.418616227
g3	22	25	18	17	10	9	12	8	14	7	0.519434629
g4	2	5	13	12	11	15	14	17	20	12	0.484381178
g5	33	42	40	32	39	42	47	41	52	41	0.443185497
Sample Class	1	1	1	1	1	2	2	2	2	2	

g0 as Current Representative (R)	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
	10	14	25	16	19	24	26	29	31	28	0.651474531
g1	20	23	34	19	29	33	40	34	40	35	0.610599511

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
TR+	30	37	59	35	48	57	66	63	71	63	0.655756027

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
TR-	10	9	9	3	10	9	14	5	9	7	0.011180124

TR+ as Current Representative (R)	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
	30	37	59	35	48	57	66	63	71	63	0.655756027
g3	22	25	18	17	10	9	12	8	14	7	0.519434629

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
TR+	52	62	77	52	58	66	78	71	85	70	0.422486467

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
TR-	8	12	41	18	38	48	54	55	57	56	0.707199178

TR- as Current Representative (R)	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
	8	12	41	18	38	48	54	55	57	56	0.707199178
g4	2	5	13	12	11	15	14	17	20	12	0.484381178

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
TR+	10	17	54	30	49	63	68	72	77	68	0.689020586

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
TR-	6	7	28	6	27	33	40	38	37	44	0.69745542

Previous Representative as Current Representative (R)	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Pearson
	8	12	41	18	38	48	54	55	57	56	0.707199178
g5	33	42	40	32	39	42	47	41	52	41	0.443185497

Continued

Figure 6.1: Cluster Representative Refinement Procedure

Here g0 represents the current cluster representative (R) and its class relevance score $((FT_x, R)$, here Pearson score), is shown. Now among all the genes g1, g2, g3, g4, and g5, the Pearson score of g1 is the highest. So, g1 is chosen for the merging process. Then g1 is added with R to create the temporary augmented representative $(TR^+ = R + g1)$ and also its sign-flipped value is added with the R to form the temporary augmented representative $(TR^- = R - g1)$. The Pearson score of TR^+ is greater than the Pearson score of TR^-, so TR^+ is chosen. Now the Pearson score of $TR^+$is greater than the Pearson score of R, so $TR^+$is considered as the current cluster representative and $R = TR^+$. This process is continued for all other genes. Now, g3 is chosen as it is the gene with the next highest Pearson value. g3 and its sign-flipped value are added individually with current cluster representative R to form $TR^+ = R + g3$ and $TR^- = R - g3$ respectively. In this case, Pearson score of $TR^-$is greater than the Pearson score of TR^+. So, TR^- is chosen. Then Pearson score of $TR^-$is Checked with the Pearson score of R and here

Pearson score of $TR^-$is greater than the Pearson score of R. So, TR^- is considered as current cluster representative and $R = TR^-$. In this way, cluster representative is refined. This process is repeated for every member of GS_{lx} subset.

After the formation of the first cluster and its corresponding augmented representative, R is assigned toAR_{lx1} that means $AR_{lx1} = R,$ and the supervised clustering process is repeated to form the second cluster with the gene (let G_{lz}) with next highest FT_x value fromGS_{lx} subset. In this way a set of clusters $CGS_{lx} = \{C_1GS_{lx}, C_2GS_{lx}, \dots.., C_kGS_{lx}, \dots.\}$ and their corresponding augmented cluster representatives $AR_{lx} = \{AR_{lx1}, \dots\dots\dots, AR_{lxk}, \dots\dots\}$are formed. After that Q number of most powerful augmented cluster representatives are chosen (as modified features) according to their FT_xvalue from the generated clusters and with these Q number of modified features, a reduced sub dataset RSD_{lx} of sub dataset $SD_{lx}$is formed.

In this way, for every bootstrapped version (BK_l) of the training dataset, 7 number of RSD_{lx} sub-datasets are created and for every RSD_{lx} an individual classifier is constructed using any existing classifier and finally, an ensemble classifier (EC) is formed by combining all these classifiers of all bootstrapped versions using the majority voting technique. To classify every sample using this ensemble classifier, each classifier votes or classifies the sample for a particular class, and the class for which the highest number of votes is obtained is considered as the output class.

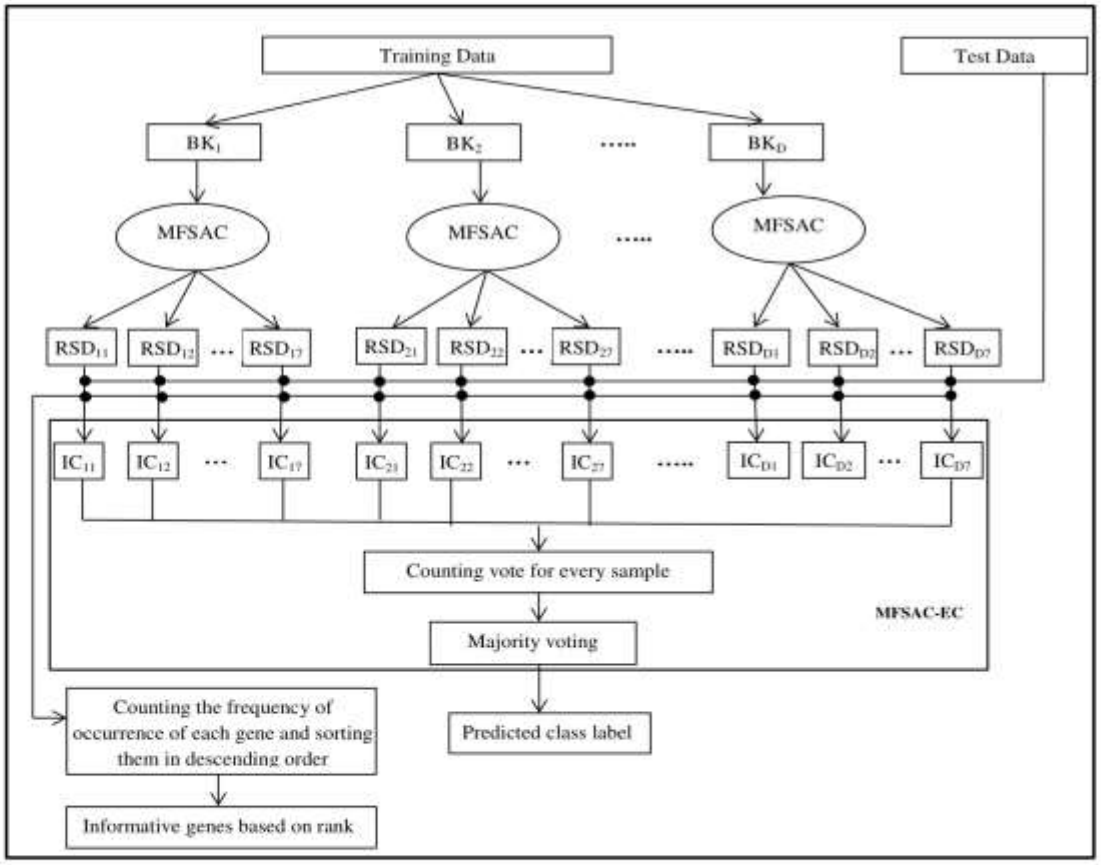


Figure 6.2: Block Diagram of the Proposed MFSAC-EC Model

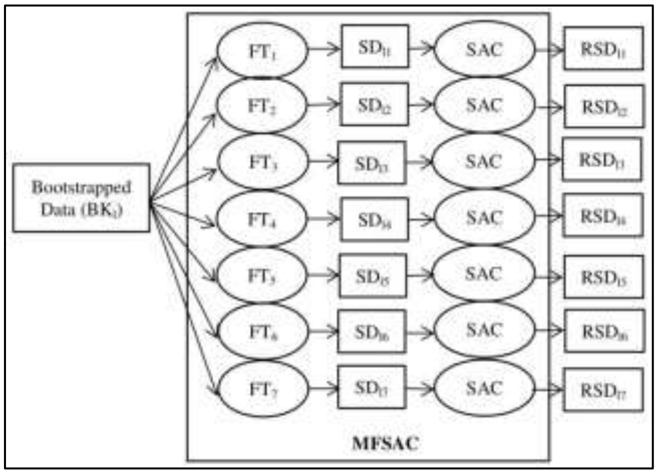


Figure 6.3: Block Diagram of MFSAC Method

MFSAC method based Informative Attribute Ranking

For every gene (feature/attribute), the frequency of occurrence that means the total number of times it appears in all sub-datasets generated by the MFSAC method for all bootstrapped versions is calculated. Then according to their frequency of occurrence, those genes are ordered or ranked. The top-ranked genes with the highest occurrence frequency are considered the most informative cancer-related genes. The block diagram of the proposed MFSAC-EC model is shown in Figure 6. 2, while the block diagram of the MFSAC method is shown in Figure 6.3. The algorithm of the proposed model is described below.

Algorithm: MFSAC-EC

Input: A $K_{U\times V}$ data matrix (here, gene expression data matrix) containing U number of data objects (here, cancer samples) and V number of attributes (here, genes).

Output: An ensemble classifier MFSAC-EC is formed to classify test samples. From MFSAC generated sub-datasets, informative genes are selected according to their rank. Every gene is ranked according to its frequency of occurrence.

Definitions:

$E = \{E_1, E_2, \dots\dots, E_s, \dots E_U\}$ is the set of objects or samples of $K_{U\times V}$ data matrix. Every sample E_s is a V dimensional vector.

$G = \{G_1, G_2, \dots\dots, G_t, \dots. G_V\}$is the set of features or genes of $K_{U\times V}$ data matrix. Every gene $G_t$is a U dimensional vector.

$BK = \{BK_1, BK_2, \dots, BK_l, \dots\dots, BK_D\}$is a set of the bootstrapped version of the original training dataset. In every bootstrapped dataset the number of samples varies from the original dataset but the number of features is the same as the original dataset.

$C_{U\times 1}$is a class vector representing the associated class label for every sample. For a data matrix N distinct class labels exist and class labels are taken from a set $= \{d_1, d_2, \dots\dots, d_k, \dots d_N\}$.

$FT_x(G_t, C)$isx^{th} filter score function which returns the class relevance value of G_t gene with respect to class vector C using FT_xscore function, for $x = 1\ to\ 7$ as 7 represents the total number of filtering score functions used here.
$GS_{lx}(GS_{lx} = P)$is a set of top-ranked genes of G selected using FT_xscore function and $SD_{lx}$is corresponding sub dataset of BK_l. Here SD_{lx} is a data matrix containing P number of genes.
$CGS_{lx} = \{C_1GS_{lx}, C_2GS_{lx}, \dots\dots, C_kGS_{lx}, \dots.\}$ And $AR_{lx} = \{AR_{lx1}, \dots\dots\dots, AR_{lxk}, \dots\dots\}$ are the set of clusters and corresponding cluster representatives respectively generated from the corresponding subset GS_{lx} of SD_{lx} . Here every AR_{lxk} is a vector.
TR^+, TR^-, R are vectors similar to a gene vector.
$RSD_l = \{RSD_{l1}, RSD_{l2}, \dots\dots\dots, RSD_{lx}, \dots\dots RSD_{l7}\}$is a set of sub-datasets each containing Q number of most relevant cluster representatives formed for every bootstrapped dataset BK_l.
$CF_l = \{IC_{l1}, IC_{l2}, \dots\dots\dots, IC_{lx}, \dots\dots IC_{l7}\}$is a set of classifiers formed for every bootstrapped dataset.

```
1. Create $D$ number bootstrapped version of training dataset $K$.
2. For Every bootstrapped dataset $BK_l$repeat step 3
3. Repeat for $x = 1\ to\ 7$
    A. Repeat for $t = 1\ to\ V$
        a) Calculate class relevance score $FT_x(G_t, C)$of $G_t$ gene, where $G_t \in G$, with
           respect to class vector $C$
    B. Select $P$ number of top-ranked genes from $G$ based on $FT_x$ score function and
       form $GS_{lx}$ gene subset with corresponding $SD_{lx}$ sub dataset
    C. Set $k = 0$
    D. Repeat until $GS_{lx} = \emptyset$
        a) Set $k = k + 1$
        b) Set $AR_{lxk} = 0$, $R = 0$, and $i = 0$
        c) Select the gene (let $G_{li}$ ) whose $FT_x$ score value is maximum among all
           genes of $GS_{lx}$and set $R = G_{li}$
        d) Add $G_{li}$ to $C_kGS_{lx}$, and delete $G_{li}$ from $GS_{lx}$
        e) Set count =1
        f) Repeat for $j = 1\ to\ |GS_{lx}|$
            I.   Compute first augmented representatives $TR^+$ by
                 adding$G_{lj} \in GS_{lx}$ with $R$ that means $TR^+ = R + G_{lj}$
            II.  Compute second augmented representatives $TR^-$ by
                 adding sign-flipped version of $G_{lj} \in GS_{lx}$ with $R$ that
                 means $TR^- = R - G_{lj}$
            III. Compute class relevance value $FT_x(TR^+, C)$and
                 $FT_x(TR^-, C)$ using $FT_x$ score function
            IV.  If $FT_x(TR^+, C) \geq FT_x(TR^-, C)$then
                     If $FT_x(TR^+, C) > FT_x(R, C)$ then
```

```
                    • Set $R = R + G_{lj}$ and add $G_{lj}$ to $C_kGS_{lx}$ and
                      delete $G_{lj}$ from $GS_{lx}$
                    • count = count +1
        V.   If $FT_x(TR^-, C) > FT_x(TR^+, C)$ then
                 If$FT_x(TR^-, C) > FT_x(R, C)$ then
                    • Set $R = R - G_{lj}$ and add $G_{lj}$ to $C_kGS_{lx}$ and
                      delete $G_{lj}$ from $GS_{lx}$
                    • count = count + 1
      g) Set $R = R/count$
      h) Set $AR_{lxk} = R$
   E. Select $Q$ number of most relevant cluster representatives according to $FT_x$ score
      from $AR_{lx}$ set and form $RSD_{lx}$ sub data set.
   F. Construct a classifier $C_{lx}$ for $RSD_{lx}$ sub data set
4. Apply a test sample over all the classifiers of all bootstrapped dataset and calculate the
   prediction accuracy of each classifier
5. Apply simple voting over all predictions to form an ensemble classifier $EC$ and get final
   prediction.
6. Calculate number of occurrences for every gene for all $RSD_{lx}$ sub datasets across all
   bootstrapped versions and rank them according to their count.
7. Select a number of top-ranked genes as informative genes.
8. End
```

6.3 Results

To assess the performance of the proposed MFSAC-EC model, four well-known existing classifiers named K-Nearest Neighbor [34], Naive Bayes [34], Support vector machine [189], and Decision tree(c4.5) [34] are applied independently in this model and four different ensemble classification models are formed. To prove the superiority of the proposed model, it is compared with existing well-known filter methods (used here) and existing recognized gene selection methods [167, 173, 177] and also with different existing ensemble classifiers [178, 180, 190, 181, 182]. To analyze the performance, the methods are applied to different publicly available cancer and other disease-related gene expression datasets. The major metrics used here for evaluations of the performance of the proposed classifier are the Cross-validation method (LOOCV, 5-fold, and 10-fold), ROC Curve, and Heat map.

Description and Preprocessing of the Datasets

The experimentation has been carried out over ten publicly available different gene expression binary class and multi-class datasets. Among these datasets, eight datasets are cancer datasets and two arthritis datasets. The eight cancer datasets are Leukemia [170], Colon [191], Prostate [192], Lung [193], RBreast [194], Breast [195], MLL [196], and SRBCT [197]. To show the accuracy of the proposed model with respect to other than cancer datasets here two arthritis datasets RAHC [198] and RAOA [198] are also considered. The summary of the datasets is represented in Table 6.2.

Table 6.2. Description of Cancer Gene Expression Datasets

Dataset	Data Dimension Gene × Sample (Original)	Data Dimension Gene × Sample (Used)	Sample Class Labels	Dataset	Data Dimension Gene × Sample (Original)	Data Dimension Gene × Sample (Used)	Sample Class Labels
Leukemia	7129 × 72	7070 × 72	2	Breast	7129 × 49	7129 × 49	2
Colon	2000 × 62	2000 × 62	2	MLL	12582 × 72	12582 × 72	3
Prostate	12600 × 136	12600 × 136	2	SRBCT	2308 × 63	2308 × 63	4
Lung	12533 × 181	12533 × 181	2	RAHC	41057 × 50	41057 × 50	2
Rbreast	24481 × 97	24188 × 97	2	RAOA	18433 × 30	18433 × 30	2

In the Leukemia dataset [170], the gene expression data matrix is prepared using Affymetrix oligonucleotide arrays. The original dataset consists of two datasets: the training dataset and the testing dataset. The training dataset consists of 38 samples (27 Acute Lymphoblastic Leukemia (ALL) and 11 Acute Myeloid Leukemia (AML)) while the test dataset consists of 34 samples (20 Acute Lymphoblastic Leukemia (ALL) and 14 Acute Myeloid Leukemia (AML)), each with 7129 probes from 6817 genes. For the Leukemia dataset, training and test datasets are merged here and genes with missing values are removed and finally, the dataset with 7070 genes and 72 samples is prepared.

In the Colon cancer dataset [191], gene expression of 6500 genes for 62 samples is measured using Affymetrix oligonucleotide arrays. Among these 62 samples, 40 are Colon cancer samples and 22 are normal samples. Among these 6500 genes, 2000 genes are selected based on the confidence of measured expression levels.

Prostate cancer dataset [192] also consists of training and testing datasets. In the training dataset, among 102 samples, 50 are normal samples and 52 are prostate cancer samples. In the test dataset among 34 samples, 25 are prostate cancer samples and 9 are normal prostate samples. Gene expression of every sample is measured with respect to 12600 genes using Affymetrix chips. Here, training and test datasets are merged, and a dataset with 12600 genes and 136 samples is formed.

The Lung cancer dataset [193] consists of 181 samples. Among these samples, 31 are malignant pleural mesothelioma and rest150 adenocarcinoma of lung cancer. Each sample is represented by 12533 genes and the gene expression of every sample is measured using Affymetrix human U95A oligonucleotide probe arrays.

In Rbreast data set [194], the patients, who are considered as breast cancer patients after 5 years intervals of initial diagnosis, fall under the category of relapse and rest as no relapse of metastases. 97 samples have been provided in which 46 patients developed distance metastases within 5 years and they are considered as relapse while the remaining remained healthy and are labeled as non-relapse. This dataset comprises 24481 genes and among them, 293 are removed.
In the Breast cancer dataset [195], the gene expression of 49 samples is measured using HuGeneFL Affymetrix microarray arrays. Breast tumors are positive or negative in the presence

or absence of estrogen receptors (ER). In this dataset, 25 samples are ER+ tumors and 24 samples are ER- tumors.

MLL [196] is a type of dataset which comprises of training data set of 57 leukemia samples including 20 ALL, 17 MLL, and 20 AML and the test dataset including 4 ALL, 3 MLL, and 8 AML samples. For MLL cancer dataset training and test, datasets are merged here and finally, the dataset with 12582 genes and 72 samples are prepared.

SRBCT dataset [197] is introduced as a dataset comprising of gene-expression for identifying small round blue-cell tumors of childhood SRBCT and samples of this dataset are further divided into four class which are neuroblastoma, rhabdomyosarcoma, non-Hodgkin lymphoma, and Ewing family of tumors and they are obtained from cDNA microarrays. A training set consisting of 63 SRBCT tissues, a test set consisting of 20 SRBCT and 5 non-SRBCT samples are available. Here we have considered only the training dataset. Each tissue sample is already standardized to zero mean value and has a unit variance across the genes.
RAHC commonly known as Rheumatoid Arthritis versus Healthy Controls is a data set [198] which comprises of gene expression characterizing as peripheral blood cells of 32 patients with RA, 3 patients with probable RA, and 15 age with sex-matched healthy controls performed under microarrays with a complexity of 26000 unique genes of 46000 elements.

RAOA commonly known as Rheumatoid Arthritis versus Osteoarthritis is a dataset [198] that includes the gene expression of thirty patients in which 21 of them are with RA and the remaining 9 of them are with OA. The Cy3 labelled common reference sample and the Cy5 labelled experimental cDNA were combined, then hybridised to the lymphochips. (consisting of 18000 cDNA spots which symbolize immunology in the genes of relevance).

Tools Used

The algorithms are implemented using Python programming language and Scikit-learn libraries [199] which are explained in [200] for ML algorithms. The programs are executed on an online Colab platform with 12 GB RAM and Intel(R) Xeon(R) processor available in the "CPU" Runtime. Figures and tables are generated in the Matplotlib library [201] and also in Microsoft Excel. The python codes used here are available at https://github.com/NSECResearchCD-SLB/PEERJ_MFSAC_EC.

In the following subsections, first, the different types of metrics used here are discussed, and then the performance of the proposed MFSAC-EC model is verified with respect to these metrics. This is followed by comparing the classification performance of the proposed model with different existing methods in terms of 10-fold cross-validation. The proposed model does not only perform the task of classification but also ranks every attribute or gene in descending order based on its information present in the dataset. To show the effectiveness of this ranking procedure topmost eight genes from Colon cancer and Leukemia cancer datasets are represented with their corresponding names, symbols, and references in significant cancer-related journals to demonstrate their significant roles in these cancers.

Evaluation Metrics

The performance of the proposed MFSAC-EC classifier is established with respect to the following measures.

Cross-Validation method

The first well-known metric used here to evaluate the classification model performance is the k-fold cross-validation method [182]. In the k-fold cross-validation method, the dataset is randomly divided into k number of folds and k-1 folds are used for training and one fold is used for testing. The process is repeated for k number of times and average classification accuracy is taken. When k is set at 1 that means the fold size is equal to the size of the dataset (training dataset size is equal to one less than the number of samples in the dataset and validation is done using the remaining sample) then it is considered as Leave one out cross-validation method (LOOCV). For k is equal to 2, the cross-validation method is named the household method. It has been found that when k is set at a very small value that means the fold size is large then the accuracy of the classification model is affected by low bias and high variance problems. On the other hand, if k is set at a high value that means the fold size is not so large then the classification accuracy of the classification model has a high bias but low variance. It has been found that 10-fold cross-validation method outperforms the LOOCV method [202, 203, 204] and it has been also endorsed that the 10-fold cross-validation method as a better measure for classification.
In training-testing random splitting the dataset is initially randomly partitioned into training set (2/3rd of the dataset) and testing set (1/3rd of the dataset) with 50 runs.

ROC curve analysis

The performance of the proposed classifier for two-class datasets is also judged using Receiver Operator Characteristic (ROC) analysis [182]. It is a visual method for evaluating binary classification models. Under this analysis, the following measures are considered to judge the binary classification model.
Classification accuracy (Acc) is defined as,

$$Acc = \frac{TP+TN}{TP+FP+TN+FN} \qquad 0 \leq Acc \leq 1$$

The sensitivity (SN) or True Positive Rate (TPR) can be defined as,

$$SN = TPR = \frac{TP}{TP+FN}$$

The specificity (SP) or True Negative Rate (TNR) can be defined as,

$$SP = TNR = \frac{TN}{TN+FP}$$

The False Positive Rate (FPR) can be defined as:

$$FPR = (1 - specificity) = \frac{FP}{FP+TN}$$

The Positive Predicted Value (PPV) can be defined as:

$$PPV = \frac{TP}{TP + FP}$$

The Negative Predicted Value (NPV) can be defined as:

$$NPV = \frac{TN}{TN + FN}$$

Where TP, TN, FP, FN are true positive, true negative, false positive, and false negative respectively.

The ROC curve is plotted considering TPR along the y-axis and FPR along the x-axis. The area under the ROC curve (AUC) is used to represent the performance of the binary classification model. The higher AUC value of a ROC curve for a particular classification model signifies the better performance of the classification model in differentiating positive and negative examples. The range of AUC value is 0<=AUC<=1.

Heat map analysis

A heatmap is a data representation diagram in which the values for a variable of interest are portrayed using a data matrix. In this data matrix, the values of the variable are represented across two-axis variables as a grid of colored squares. The axis variables are divided into ranges and each cell's color represents the intensity of that variable for the particular ranges of values of axis variables.

Here, the performance of the proposed classifier for multi-class datasets is judged using Heat map representation of confusion matrix [205], where a confusion matrix is a tabular representation to visualize the performance of a classification model in terms of true positive, true negative, false positive and false negative.

Parameter Estimation

Before running the MFSAC-EC, the parameters are settled down. In MFSAC-EC the input training dataset is bootstrapped. The proposed MFSAC-EC model is run here varying the number of bootstrapped datasets (D) from 5 to 30 and the classification accuracy of this model is more or less the same from 10 to the rest of the range. So, the number of bootstrapped datasets for every training dataset for this model is set at 10.

In MFSAC method, initially P number of genes is selected by each filter method. Here in Table 2, the classification accuracy of the proposed model is shown with respect to different values of P. From Table 6.3, it has been found that the proposed model gives the best result for P=100 for RAOA and RAHC datasets. In case of Breast cancer, Lung cancer, MLL and SRBCT datasets it gives the best result at P = 200. For Leukemia datasets it gives the best result at P= 500. So, it can be said that MFSAC-EC gives best result for P value within 200 to 500 in all cases for all datasets except Colon and Prostate. In Colon and Prostate, it shows the best result for P = 1500.

Here we have used SVM, DT (C4.5), NB, and KNN classifiers individually for forming different ensemble classification models. All the classifiers are implemented using Scilit-learn libraries of Python. For all classifiers, we have set parameters with default parameter values. For DT as default setting we have used splitting function = Gini, Splitting criterion = best, height = none (

Table 6.3: Classification Accuracy of MFSAC-EC depending on varying number of genes selected by each Filter

Dataset	Evaluation Metric	MFSAC-EC																							
		P=100				P=200				P=500				P=1000				P=1200				P=1500			
		NB	KNN	DT	SVM	NB	KNN	DT	SVM	NB	KNN	DT	SVM	NB	KNN	DT	SVM	NB	KNN	DT	SVM	NB	KNN	DT	SVM
Leukemia	LOOCV	98.6	98.6	98.6	98.6	98.6	98.6	98.6	98.6	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	98.6	98.6	98.6	98.6	98.6	98.6	98.6	98.6	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
RAHC	LOOCV	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
MLL	LOOCV	98.6	100	100	97.2	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	97.2	100	100	97.2	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
RAOA	LOOCV	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
SRBCT	LOOCV	98.4	98.4	100	98.4	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	100	98.4	98.4	98.4	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
Breast	LOOCV	98	95.9	93.9	95.9	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	100	95.9	95.9	98	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
Lung	LOOCV	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
Rbreast	LOOCV	92.6	93.7	93.7	95.8	99	97.9	100	99	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	10 Fold	91.6	96.8	95.8	97.9	97.9	99	99	99	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
COLON	LOOCV	91.9	91.9	93.6	91.9	91.9	91.9	96.8	91.9	98.4	98.4	96.8	98.4	98.4	98.4	98.4	98.4	98.4	98.4	98.4	100	100	100	98.4	100
	10 Fold	91.9	91.9	93.6	91.9	91.9	91.9	95.2	91.9	98.4	96.8	96.8	96.8	98.4	100	98.4	98.4	100	98.4	98.4	100	100	100	98.4	100
Prostrate	LOOCV	83.8	90.4	92.7	86.8	88.2	92.7	92.7	88.2	91.2	95.6	97.1	91.9	94.9	97.1	97.8	94.1	98.5	98.5	97.8	98.5	98.5	99.3	98.5	99.3
	10 Fold	85.3	88.2	91.9	86.8	88.2	91.9	93.4	89.7	91.9	95.6	97.8	91.2	95.6	97.8	97.8	94.1	99.3	98.5	97.8	98.5	99.3	99.3	97.8	99.3

Table 6.4: Total Execution Time (Bootstrapping, Feature Selection, Training, Testing for LOOCV, 5-Fold, 10-Fold, Random Splitting) in a single run of MFSAC-EC on Different Datasets

	Leukemia	RAHC	MLL	RAOA	SRBCT	Breast	Lung	Rbreast	COLON	Prostrate
No. of Feature selected for best result	500	100	200	100	200	200	100	500	1200	3000
Total Time Taken	8mins 23secs	7mins 32secs	7mins 54secs	4mins 43secs	5mins 17secs	4mins 2secs	11mins 14secs	10mins 22secs	17mins 40secs	1hr 18mins 41secs
Time Taken for only 10 fold	35secs	30secs	41secs	36secs	30secs	32secs	36secs	33secs	30secs	36secs

that means for every sample it reaches a leaf/class node). For SVM, we have used the RBF kernel function. For KNN we have chosen K (number of nearest neighbor) value from 3 to 7.

The overall execution time of a single run of the MFSAC-EC model (considering bootstrapped dataset creation, feature selection using MFSAC, and then generating classification accuracy of test samples using LOOCV, 5-fold, 10-fold, and random splitting) and testing time using only 10-fold are shown for different datasets in Table 6.4.

Classification Performance of the Proposed MFSAC-EC Classifier

In Table 6.5, using the LOOCV method, the classification accuracy of our proposed MFSAC-EC model is 100% for different datasets (Leukemia, Breast, RBreast, Lung, RAOA, and RAHC) for all cases. In the Prostate dataset, we did not get 100% accuracy using our model with respect to any type of existing classifier. In MLL, Colon, and SRBCT it also gives 100% accuracy using all types of ensemble classifiers.

Table 6.5.Classification Accuracy of the proposed MFSAC-EC model with respect to LOOCV

Dataset	Proposed Model		Cluster Representatives			Dataset	Proposed Model		Cluster Representatives		
			1	2	3				1	2	3
COLON	MFSAC-EC	NB	100	98.39	98.39	MLL	MFSAC-EC	NB	100	100	100
		KNN	98.39	100	100			KNN	100	100	100
		DT	98.39	98.39	98.39			DT	100	100	100
		SVM	100	98.4	98.4			SVM	100	100	100
Prostate		NB	97.06	97.79	98.53	SRBCT		NB	96.83	100	100
		KNN	97.79	97.79	98.53			KNN	96.83	100	100
		DT	97.79	98.53	97.79			DT	96.83	98.41	100
		SVM	98.53	99.26	99.26			SVM	82.54	98.41	100
Leukemia		NB	100	100	100	Lung		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100
RAOA		NB	100	100	100	RAHC		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100
Breast		NB	100	100	100	RBreast		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100

In Table 6.6 and Table 6.7, it has been shown that using 5 -fold and 10-fold cross-validation, MFSAC-EC does not provide 100% accuracy only for Colon and Prostate cancer datasets. For other datasets, it provides 100% accuracy with respect to all types of ensemble classifiers.

To show the generalization property of the proposed ensemble classifiers, the classification accuracy of these classifiers is also measured repeatedly with respect to the random splitting of the dataset into a training set (2/3 data of original dataset) and test set (1/3 data of original dataset). Random splitting is done with care such that class proportion is alike in the training set and test set. In Table 6.8, the classification accuracy of the above mentioned four different types of ensemble classifiers for the different number of cluster representatives is shown in different datasets which are based on the best result of 50 random splitting of the dataset into a training set (2/3 data of original dataset) and test set (1/3 data of original dataset).

Table 6.6.Classification Accuracy of the proposed MFSAC-EC model with respect to 5-Fold Cross Validation

Dataset	Proposed Model		Cluster Representatives			Dataset	Proposed Model		Cluster Representatives		
			1	2	3				1	2	3
COLON	MFSAC-EC	NB	96.77	96.77	96.77	MLL	MFSAC -EC	NB	100	100	100
		KNN	98.39	96.77	96.77			KNN	98.61	100	100
		DT	98.39	96.77	98.39			DT	98.61	100	100
		SVM	98.39	96.77	96.77			SVM	100	100	100
Prostate		NB	97.06	97.79	98.53	SRBCT		NB	98.41	100	100
		KNN	97.79	97.79	99.26			KNN	96.83	100	100
		DT	97.06	97.79	94.85			DT	96.83	98.41	100
		SVM	97.79	98.53	99.26			SVM	96.83	100	100
Leukemia		NB	100	100	100	Lung		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	99.44	100
		SVM	100	100	100			SVM	100	100	100
RAOA		NB	100	100	100	RAHC		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100
Breast		NB	100	100	100	RBreast		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100

Table 6.7Classification Accuracy of the proposed MFSAC-EC model with respect to 10-Fold Cross Validation

Dataset	Proposed Model		Cluster Representatives			Dataset	Proposed Model		Cluster Representatives		
			1	2	3				1	2	3
COLON	MFSAC-EC	NB	98.39	98.39	98.39	MLL	MFSAC-EC	NB	100	100	100
		KNN	98.39	98.39	100			KNN	100	100	100
		DT	98.39	98.39	98.39			DT	100	100	100
		SVM	98.39	98.39	98.39			SVM	100	100	100
Prostate		NB	97.06	97.79	98.53	SRBCT		NB	96.83	96.83	100
		KNN	97.79	97.79	99.26			KNN	92.06	100	100
		DT	97.06	97.79	94.85			DT	95.24	96.83	100
		SVM	97.79	98.53	99.26			SVM	80.95	92.06	100
Leukemia		NB	100	100	100	Lung		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100
Breast		NB	100	100	100	RBreast		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100
RAOA		NB	100	100	100	RAHC		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100

From the results of Table 6.5 to Table 6.8, it has been observed that classification accuracy in the LOOCV method, 5-fold cross-validation, and 10-fold cross-validation methods is higher than the random splitting of the dataset, and the overall generalization performance of the proposed classification model is also good.

Table 6.8. Classification Accuracy of the proposed MFSAC-EC model with respect toRandom Splitting of the Datasets

Dataset	Proposed Model		Cluster Representatives			Dataset	Proposed Model		Cluster Representatives		
			1	2	3				1	2	3
COLON		NB	98.39	98.39	98.39	MLL		NB	100	100	100
		KNN	98.39	98.39	98.39			KNN	100	100	100
		DT	98.39	98.39	98.39			DT	98.61	100	98.61
		SVM	98.39	100	98.39			SVM	100	100	100
Prostate		NB	94.68	95.74	93.62	SRBCT		NB	95	85	95
		KNN	97.87	96.81	92.55			KNN	95	100	90
		DT	94.68	94.68	94.68			DT	80	90	95
		SVM	94.68	96.81	94.68			SVM	65	75	95
Leukemia	MFSAC-EC	NB	100	100	100	Lung	MFSAC-EC	NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	100
		SVM	100	100	100			SVM	100	100	100
RAOA		NB	100	100	100	RAHC		NB	100	100	100
		KNN	100	100	100			KNN	100	100	100
		DT	100	100	100			DT	100	100	81.25
		SVM	100	100	100			SVM	100	100	81.25
Breast		NB	100	100	100	RBreast		NB	91.94	91.94	91.94
		KNN	100	100	100			KNN	85.48	87.10	83.87
		DT	100	100	100			DT	83.87	79.03	80.65
		SVM	100	100	100			SVM	93.55	91.94	91.94

From the results of Table 6.5 to Table 6.8, it has been observed that classification accuracy in the LOOCV method, 5-fold cross-validation, and 10-fold cross-validation methods is higher than the random splitting of the dataset, and the overall generalization performance of the proposed classification model is also good.

The performance of the proposed model for different two-class datasets with respect to different parameters like SN, SP, PPV, NPV, FPR is shown in Table 8. From this table, it is found that the performance of the proposed model is very good with respect to all these parameters for all two-class datasets.

In Figures 6.4, the ROC curve is shown for different two-class datasets. In Figures 6.4a, 6.4b, and 6.4c, the ROC curves are shown for Breast cancer using LOOCV, for Colon cancer using 5-fold cross validation, and for RAHC dataset using 10-fold cross-validation respectively. The ROC curves for Leukemia Cancer, and Lung cancer datasets using LOOCV are given in Figures 6.5a and 6.5b respectively. For Breast cancer, Leukemia cancer, and Lung cancer, the AUC value is equal to 1.0 in every case. The ROC curves are shown for RAOA, and RBreast cancer datasets using 5-fold cross-validation in Figures 6.6a, and 6.6b respectively. For these datasets also the prediction accuracy using 5-fold cross validation is very high according to the AUC value. In Supplemental Figures 6.6c, the ROC curves are shown for Prostate cancer using 10-fold cross-validation. From these curves of 10-fold cross validation, it may be seen that except for Prostate cancer, for all other datasets the AUC value is 1 and for Prostate cancer, the AUC value is close to 1.

Table 6.9. Evaluation of MFSAC-EC classifier based on SN, SP, PPV, NPV, FPR for two class data sets with respect to LOOCV

Dataset	Proposed Model		SN	SP	PPV	NPV	FPR	Dataset	Proposed Model		SN	SP	PPV	NPV	FPR
Leukemia	MFSAC-EC	NB	100	100	100	100	0	Breast	MFSAC-EC	NB	100	100	100	100	0
		KNN	100	100	100	100	0			KNN	100	100	100	100	0
		DT	100	100	100	100	0			DT	100	100	100	100	0
		SVM	100	100	100	100	0			SVM	100	100	100	100	0
Prostate		NB	98.7	98.3	98.7	98.3	1.7	Rbreast		NB	100	100	100	100	0
		KNN	98.7	98.3	98.7	98.3	1.7			KNN	100	100	100	100	0
		DT	100	96.61	97.46	100	3.4			DT	100	100	100	100	0
		SVM	100	98.3	98.7	100	1.7			SVM	100	100	100	100	0
Colon		NB	100	100	100	100	0	Lung		NB	100	100	100	100	0
		KNN	100	100	100	100	0			KNN	100	100	100	100	0
		DT	100	100	100	100	0			DT	100	100	100	100	0
		SVM	100	100	100	100	0			SVM	100	100	100	100	0
RAHC		NB	100	100	100	100	0	RAOA		NB	100	100	100	100	0
		KNN	100	100	100	100	0			KNN	100	100	100	100	0
		DT	100	100	100	100	0			DT	100	100	100	100	0
		SVM	100	100	100	100	0			SVM	100	100	100	100	0

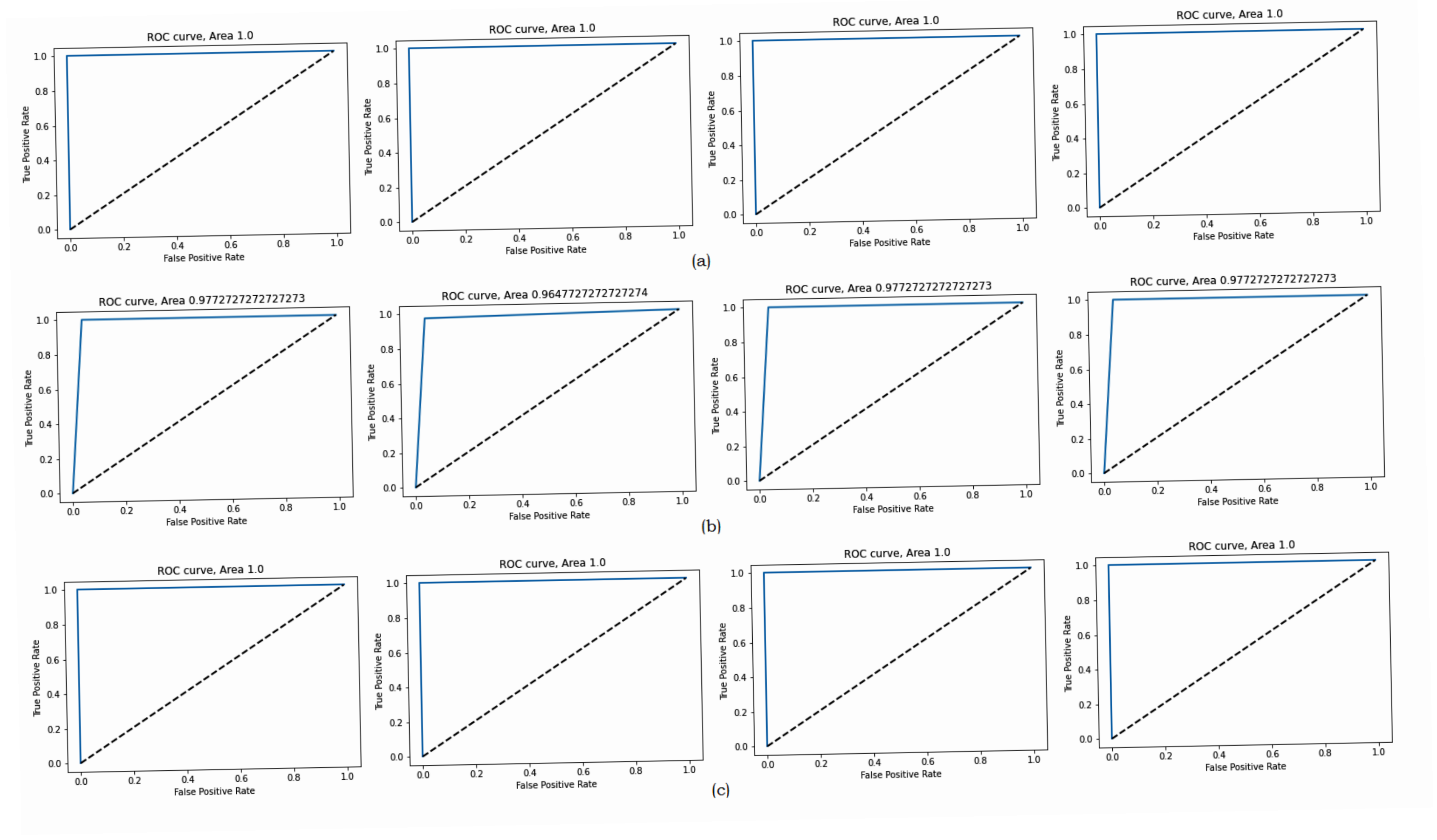


Figure 6.4: AUC using MFSAC-EC+ KNN, MFSAC-EC+NB, MFSAC-EC+ DT and MFSAC-EC+ SVM

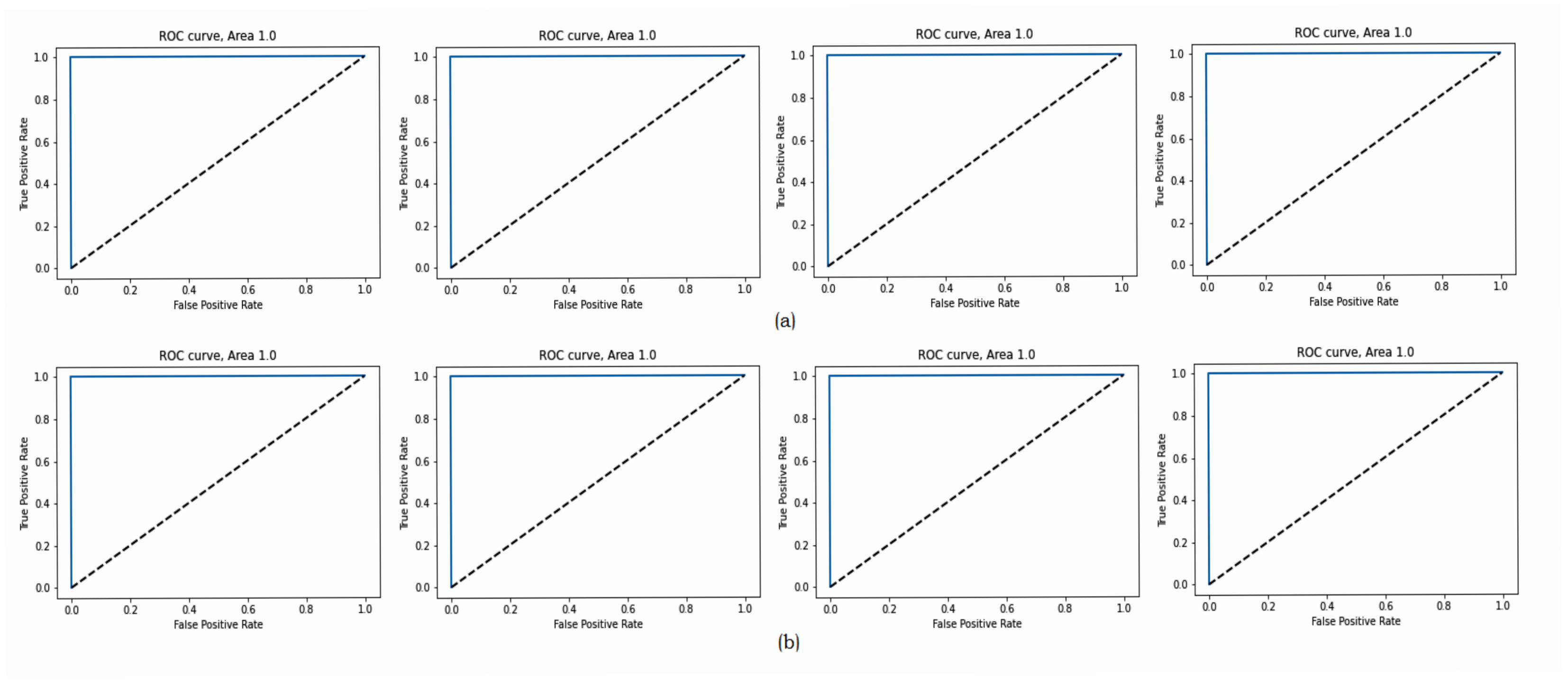


Figure 6.5: AUC using MFSAC-EC+ KNN, MFSAC-EC+NB, MFSAC-EC+ DT AND MFSAC-EC+ SVM

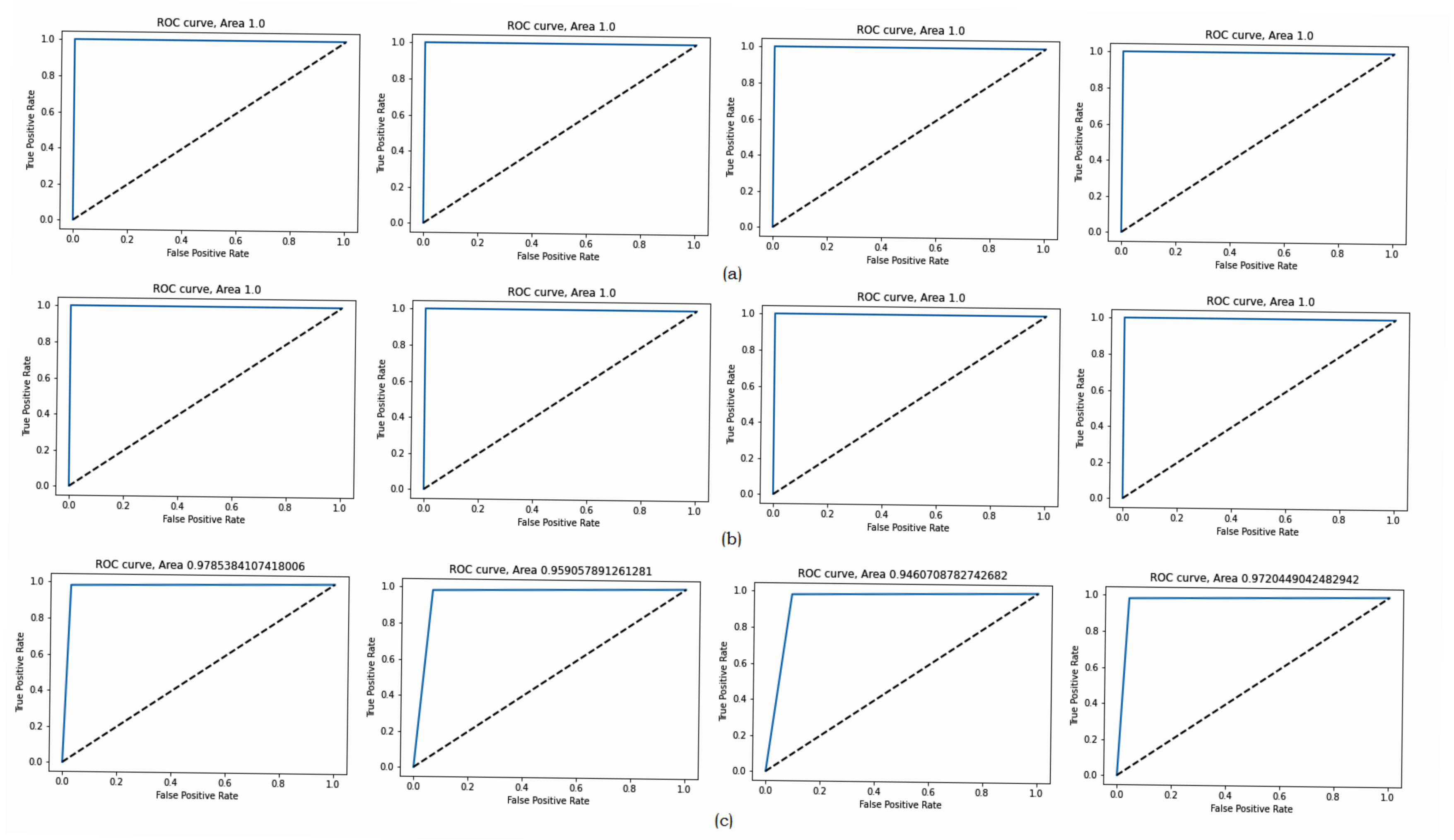


Figure 6.6 AUC using MFSAC-EC+ KNN, MFSAC-EC+NB, MFSAC-EC+ DT AND MFSAC-EC+ SVM

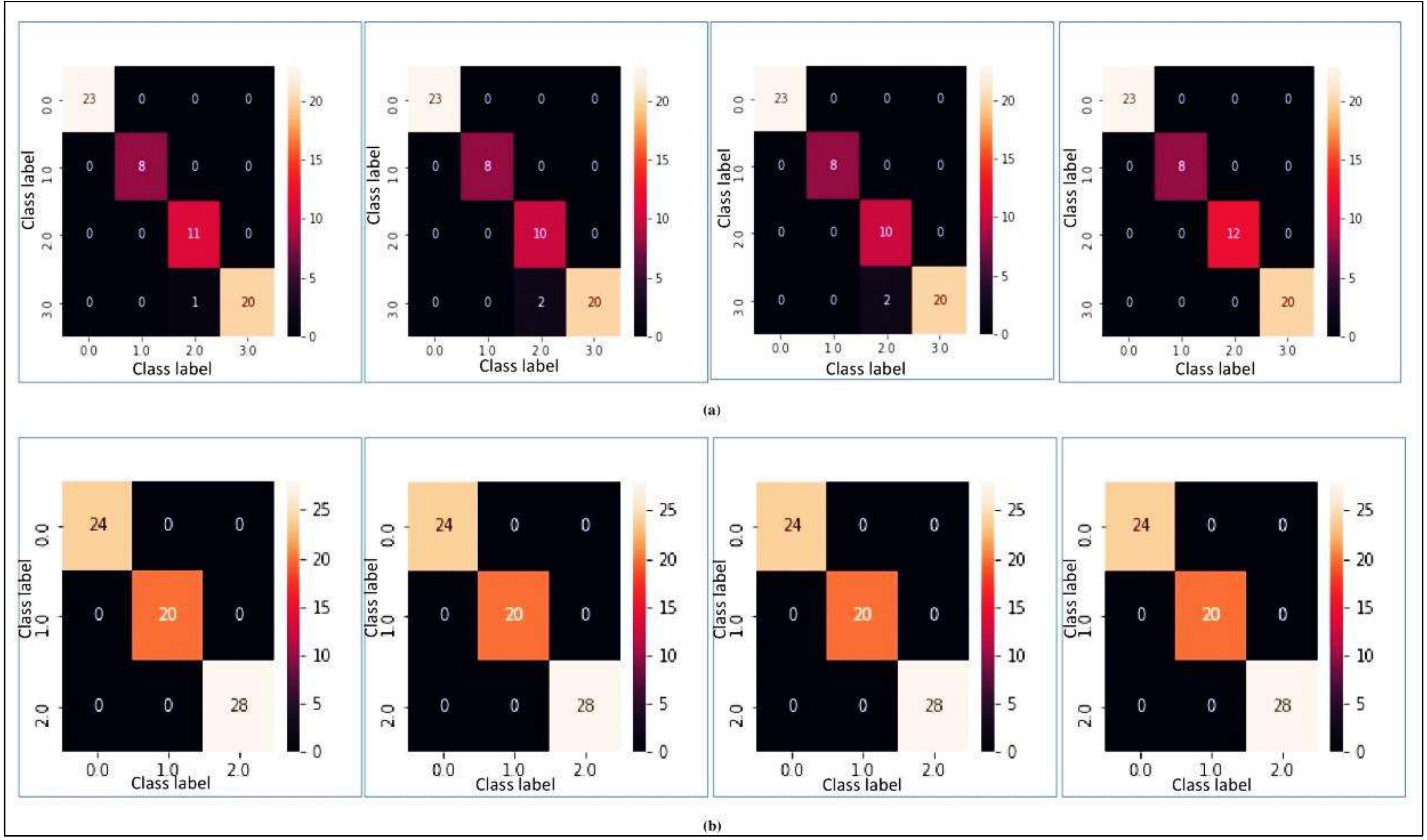


Figure 6.7: Heatmap of MFSAC-EC with base classifiers NB, KNN, DT and SVM respectively

In Figures 6.7a and 6.7b, heatmap representations of the confusion matrix are shown for multi-class datasets: SRBCT and MLL with respect to 5-fold cross-validation, and 10-fold cross-validation respectively. From these figures, it is clear that for the proposed model prediction accuracy is accurate in most cases.

Comparison of MFSAC-EC Model with Well-Known Existing Filter Methods used in this model

In Figure 6.8, the proposed MFSAC-EC model in combination with different existing classifiers is compared with different filter methods used in this model with respect to SRBCT, RAHC, Prostate, and Colon datasets in terms of 10-fold cross-validation. In all cases, the performance of the proposed model is significantly better with respect to all filters.

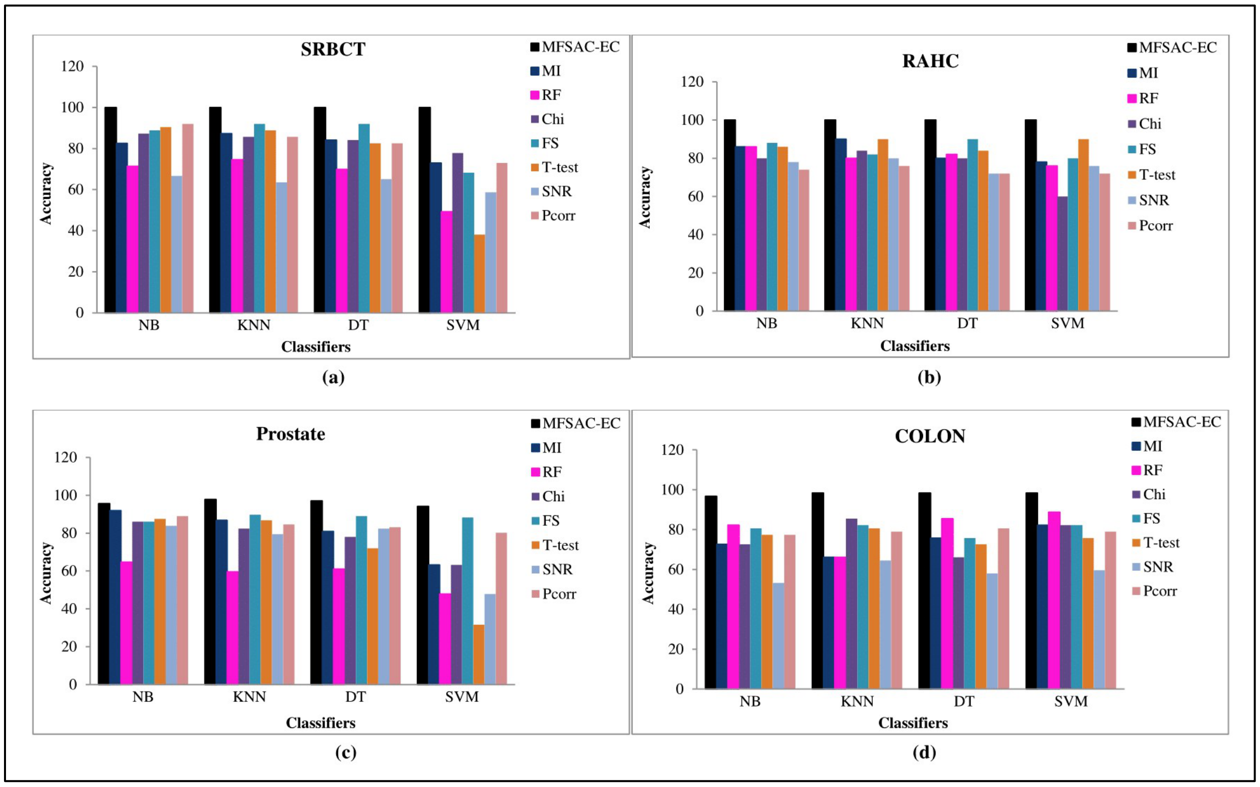


Figure: 6.8: Classification Accuracy Comparison of MFSAC-EC with different filter methods using Ten Fold Cross-Validation.

Comparison of MFSAC-EC Model with Well-Known Existing Gene Selection Methods

In Figure 6.9, the MFSAC-EC model with different existing classifiers as base classifiers are compared with existing well-known supervised gene selection methods named mRMR (minimum redundancy maximum relevance framework) [167], MSG (mutual information based supervised gene clustering algorithm) [177], CFS (Correlation-based Feature Selection) [206], and FCBF(Fast Correlation-Based Filter) [206] with respect to different classifiers using 10-fold cross-validation method. From these results, it has been found that the proposed model outperforms in most of the cases.

In Figure 6.10, the MFSAC-EC model is compared with well-known existing unsupervised gene selection methods named MGSACO [208], UFSACO [207], RSM[209], MC [210], RRFS [211], TV [212], and LS [213] with respect to DT, SVM, NB classifiers using random splitting method. From these results, it can be said that the MFSAC-EC model outperforms in all cases.

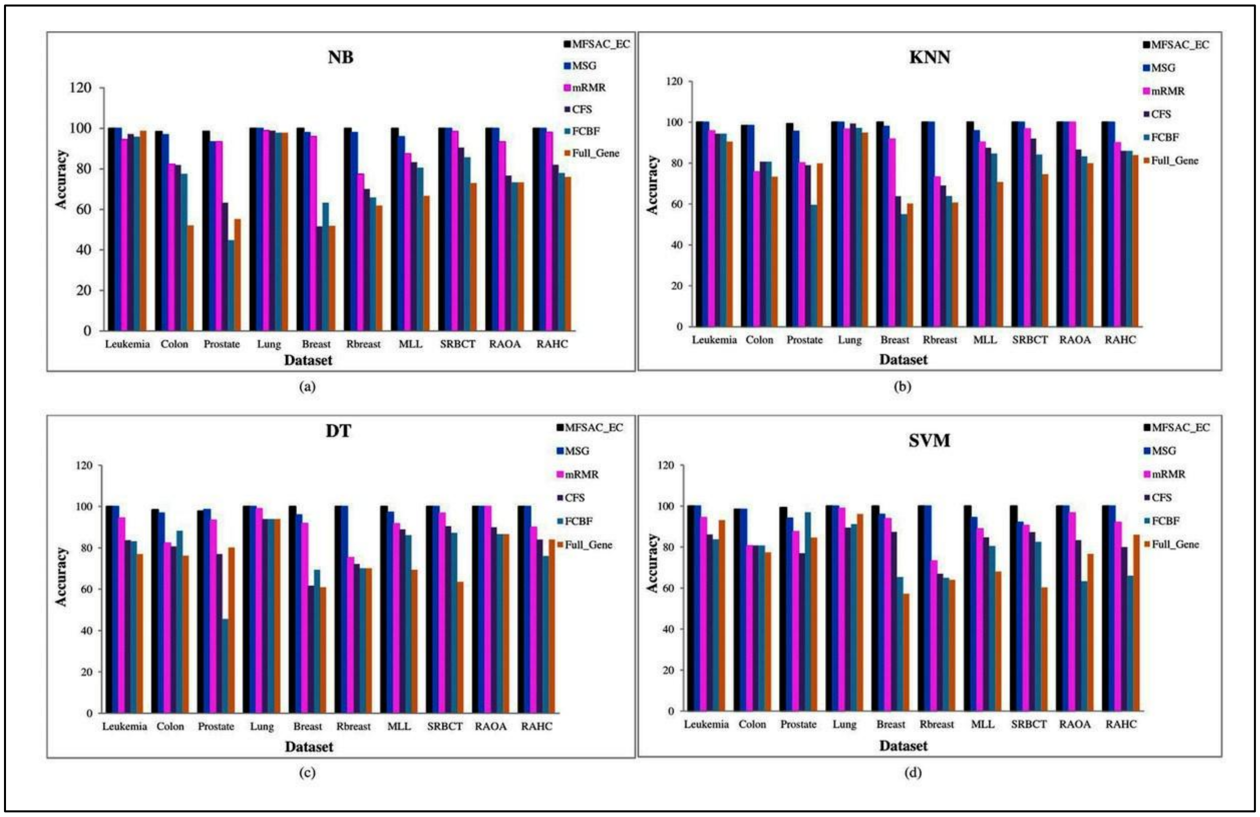


Figure 6.9 Comparison of MFSAC-EC with other well-known supervised gene selection methods and full gene set in terms of fivefold cross validation for all datasets.

Comparison of MFSAC-EC Model with Well-Known Existing Ensemble Classification and DEEP learning Models

In Table 6.10, the proposed MFSAC-EC model using the DT classifier is compared with well-known existing ensemble classification models with respect to 10-fold cross-validation. These models are PCA-based RotBoost [190], ICA-based RotBoost [190], AdaBoost [190], Bagging [190], Arcing [190], Rotation Forest [190], EN-NEW1 [181], and EN-NEW2 [181]. From Table 6.10, it is clear that the proposed model using DT classifier outperforms in all cases.

In Table 6.11, the proposed MFSAC-EC model using DT, NB, KNN as base classifiers are compared with different existing ensemble classifiers with respect to 10-fold cross-validation. These classifiers are Bagging based ensemble classifier [180], Boosting based ensemble classifier [180], Stacking based ensemble classifier [180], Heuristic breadth-first search-based ensemble classifier (HBSA) [182], Sd_Ens [180], and Meta_Ens [180]. In Table 6.12 our model using SVM and KNN as base classifiers is compared with auto-encoder-based deep learning models [214] in terms of random splitting. Here, results are shown only for the datasets for which results are available in the literature, and all other fields are marked as “Not Found”. In all cases, the MFSAC-EC model outperforms all the well-known existing ensemble models (except for the Colon cancer dataset) and deep learning models which in turn validates the usefulness of the proposed model.

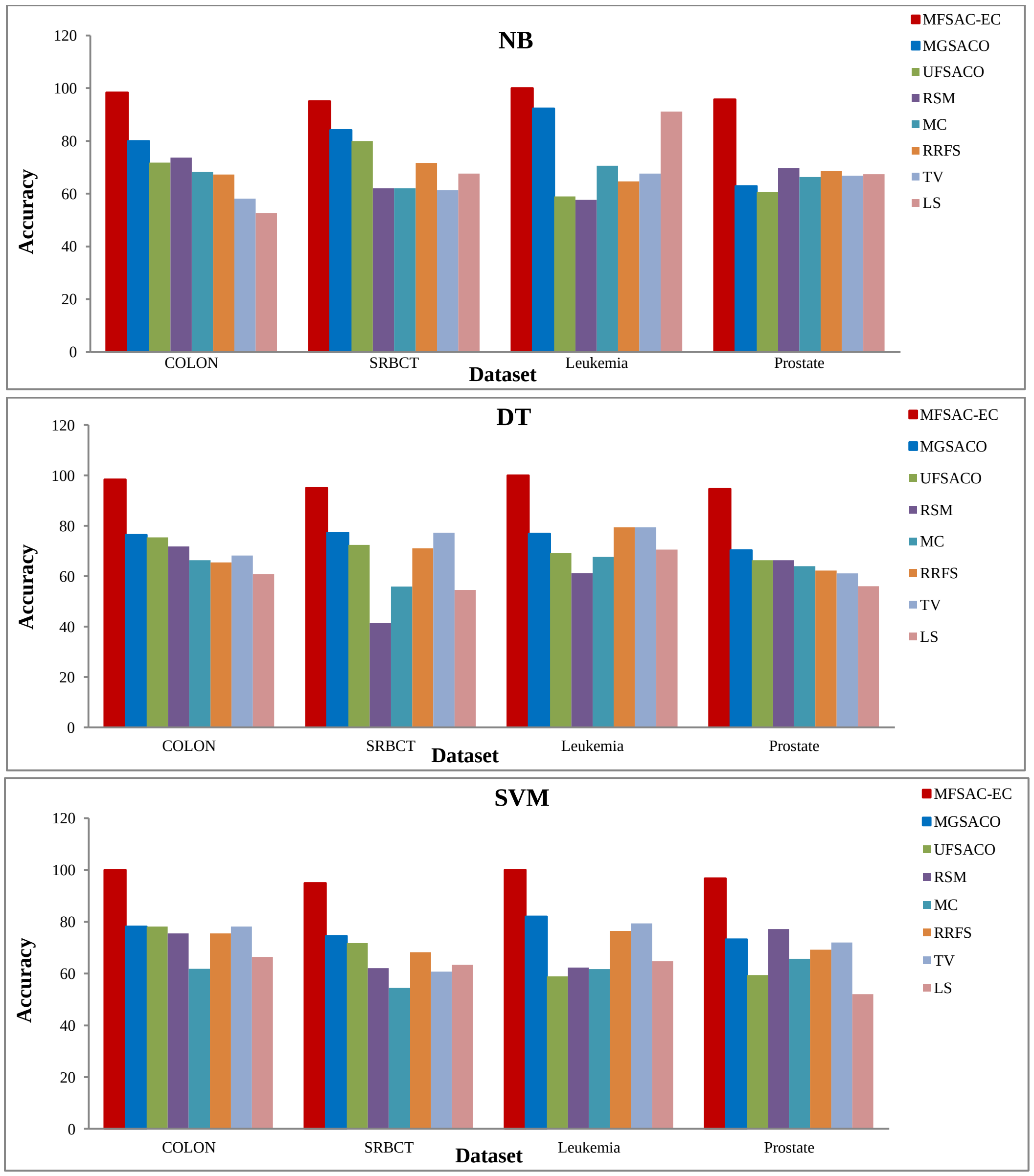


Figure 6.10: Comparison of MFSAC-EC with other well-known unsupervised gene selection methods in terms of random splitting for different datasets.

Comparison of MFSAC-EC Model with Well-Known Existing Ensemble Classification and DEEP learning Models

In Table 6.10, the proposed MFSAC-EC model using the DT classifier is compared with well-known existing ensemble classification models with respect to 10-fold cross-validation. These models are PCA-based RotBoost [190], ICA-based RotBoost [190], AdaBoost [190], Bagging [190], Arcing [190], Rotation Forest [190], EN-NEW1 [181], and EN-NEW2 [181]. From Table 6.10, it is clear that the proposed model using DT classifier outperforms in all cases.

In Table 6.11, the proposed MFSAC-EC model using DT, NB, KNN as base classifiers are compared with different existing ensemble classifiers with respect to 10-fold cross-validation. These classifiers are Bagging based ensemble classifier [180], Boosting based ensemble classifier [180], Stacking based ensemble classifier [180], Heuristic breadth-first search-based ensemble classifier (HBSA) [182], Sd_Ens [180], and Meta_Ens [180]. In Table 6.12 our model using SVM and KNN as base classifiers is compared with auto-encoder-based deep learning models [214] in terms of random splitting. Here, results are shown only for the datasets for which results are available in the literature, and all other fields are marked as "Not Found". In all cases, the MFSAC-EC model outperforms all the well-known existing ensemble models (except for the Colon cancer dataset) and deep learning models which in turn validates the usefulness of the proposed model.

Table 6.10. Comparison of MFSAC-EC using DT with different existing Ensemble Classifiers using DT in terms of 10-Fold Cross Validation

	MFSAC-EC	**PCA-based RotBoost**	**ICA-based RotBoost**	**AdaBoost**	**Bagging**	**Arcing**	**Rotation Forest**	**EN-NEW1**	**EN-NEW2**
Colon	**98.39**	95.48	96.1	94.97	94.92	69.35	95.21	79.03	83.87
Leukemia	**100**	98.75	98.77	98.22	97.47	Not Found	97.97	Not Found	Not Found
Breast	**100**	94.39	97.88	98.89	92.74	80.41	98.6	94.85	95.88
Lung	**100**	98.11	99.54	96.3	97.08	97.24	97.56	98.34	99.45
Prostate	**97.79**	Not Found	Not Found	90.44	94.12	87.5	Not Found	94.85	97.06
MLL	**100**	98.86	99.31	97.63	97.11	91.67	97.61	93.06	98.61
SRBCT	**100**	99.5	99.59	98.16	96.46	Not Found	97.44	Not Found	Not Found

Table 6.11. Comparison of MFSAC-EC using DT, KNN, NB, SVM with different existing Ensemble Classifiers using DT, KNN, NB, SVM in terms of 10-Fold Cross Validation

Dataset	MFSAC-EC				Bagging			Boosting			Stacking			HBSA		SD_Ens	Meta_Ens
	DT	NB	KNN	SVM	DT	NB	KNN	DT	NB	KNN	DT	NB	KNN	KNN	SVM		
Leukemia	**100**	**100**	**100**	**100**	94.12	88.23	73.53	91.18	88.24	75.53	91.18	91.18	91.18	88.46	88.46	92.45	94.12
Colon	98.39	98.39	**100**	98.39	95.16	66.13	90.32	98.39	87.1	91.94	98.39	93.59	93.59	75	85	94.4	99.21
Prostate	97.79	**99.26**	**99.26**	**99.26**	26.47	26.47	38.24	26.47	26.47	52.94	26.47	26.47	52.94	85.29	97.06	52.94	52.94
Lung	**100**	**100**	**100**	**100**	91.28	96.64	97.32	81.88	95.3	97.99	97.99	97.99	96.64	Not Found	Not Found	81.88	97.99
Breast	**100**	**100**	**100**	**100**	78.95	36.84	68.42	68.42	36.84	68.42	68.42	68.42	68.42	Not Found	Not Found	73.49	79.87

Table 6.12. Comparison of MFSAC-EC using SVM and KNN with respect to different existing deep learning Classifiers using random splitting

Dataset	SVM			KNN		
	MFSAC-EC	Folded Autoencoder	Autoencoder	MFSAC-EC	Folded Autoencoder	Autoencoder
Colon	**100**	90.15	73.11	**98.39**	81.09	56.97
Prostate	**96.81**	84.16	64.3	**97.87**	76.48	52.1
Leukemia	**100**	93.62	84.12	**100**	85.24	77.13

Biological Significance Analysis

The top 8 genes selected by the MFSAC-EC model for Colon cancer and Leukemia are listed in Table 6.13. For every gene, the name and symbol of the gene as well as the Accession number of the Affymetrix chip are listed. Apart from this information, to validate those genes, biomedical literature of the genes is searched and for every gene, the corresponding reference about its role and significance for a particular disease is provided.

Table 6.13. List of genes selected by MFSAC-EC model for Colon and Leukemia cancer Datasets

Dataset	Gene Name	Accession Number	Description	Validation of Genes
Colon	TPM1	Hsa.1130	Human tropomyosin isoform mRNA, complete cds.	[215], [216], [217]
	IGFBP4	Hsa.1532	Human insulin-like growth factor binding protein-4 (IGFBP4) gene, promoter and complete cds.	[218], [219], [220]
	MYL9	Hsa.1832	Myosin Regulatory Light Chain 2, Smooth Muscle Isoform (Human); contains element TAR1 repetitive element	[221], [222]
	ALDH1L1	Hsa.10224	Aldehyde Dehydrogenase, Mitochodrial X Precursor (Homo sapiens)	[223], [224]
	KLF9	Hsa.41338	Human mRNA for GC box binding protein/ Kruppel Like Factor 9, complete cds	[225], [226], [227]
	MEF2C	Hsa.5226	Myocyte-Specific Enhancer Factor 2, Isoform MEF2 (Homosapiens)	[228], [229], [230]
	GADPH	Hsa.1447	Glyceraldehyde 3-Phosphate Dehydrogenase	[231], [232]
	TIMP3	Hsa.11582	Metalloproteinase Inhibitor 3 Precursor	[233], [234]
Leukemia	TXN	X77584_at	TXN Thioredoxin	[235], [236], [237]
	CSF3R	M59820_at	CSF3R Colony stimulating factor 3 receptor (granulocyte)	[238], [239], [240], [241]
	MPO	M19508_xpt3_s_at	MPO from Human myeloperoxidase gene	[242], [243], [244], [245]
	LYZ	M21119_s_at	LYZ Lysozyme	[159],[246], [247]
	CST3	M27891_at	CST3 Cystatin C (amyloid angiopathy and cerebral hemorrhage)	[248]
	ZYX	X95735_at	Zyxin	[248], [249]
	CTSD	M63138_at	CTSD Cathepsin D (lysosomalaspartyl protease)	[246]
	CD79A/ MB-1 gene	U05259_rna1_at	MB-1 membrane glycoprotein	[246]

6.4 Discussion

In this paper, a new Multiple Filtering and Supervised Attribute Clustering algorithm-based ensemble classification model named MFSAC-EC is proposed. The main motivation behind this work is to develop a machine learning-based ensemble classification model to overcome the over-fitting problem which arises due to the presence of sample class imbalance problem, small sample size problem, and also high dimensional feature set problem in the microarray gene expression dataset, to enhance the prediction capability of the proposed model.

Nowadays, in designing machine learning models, the use of ensemble methodology has been increasing day by day as it incorporates multiple learning algorithms and also training datasets in different efficient manners to improve the overall prediction accuracy of the model. Due to the inclusion of prediction accuracy of multiple learning models and also the use of different bootstrapping datasets, the chances of potential overfitting in training data is greatly reduced in the ensemble models, and as a consequence the prediction accuracy increases. One necessary condition of the superior performance of an ensemble classifier with respect to its individual member/base classifier is that every base classifier should be very accurate and diverse [190]. A classifier is considered accurate if its generalization capability is high and two classifiers satisfy diverse property if their prediction in classifying the same unknown samples varies from each other. The general principle of ensemble methods is to rearrange training datasets in different ways (either by resampling or reweighting) and build an ensemble of base classifiers by applying a base classifier on every rearranged training dataset [190].

In our proposed ensemble model, at first, a number of bootstrapped datasets of the original training dataset is created. In every bootstrapped dataset, the class imbalance problem is solved using the oversampling method. Then for every bootstrapped dataset, a number of sub-datasets are created using the MFSAC method (which is a hybrid method combining multi-filters and a new supervised attribute/gene clustering method), and then for every generated sub dataset, a base classifier is constructed using any existing classification model. After that, a new ensemble classifier (EC) is formed using the majority voting scheme by combining the prediction accuracy of all those base classifiers.

The prediction accuracy of the proposed model is verified by applying it to high-dimensional microarray gene expression data From Figure 6.9, and Figure 6.10 it has been found that the classification accuracy of the MFSAC-EC model is much better than the well-known existing gene selection methods. From Table 6.10, Table 6.11, and Table 6.12, it has been also found that the proposed MFSAC-EC classification model is superior to the existing ensemble classification models in almost every case. The superior performance of the proposed model is due to the following reasons:

- The generation of the different bootstrapped versions of training data and also the use of the oversampling procedure to balance the cardinality of majority class and minority class in every bootstrapped dataset reduces the chances of the overfitting problem of a classifier.
- Different types of filter methods are used in the MFSAC method. It has been already observed that one filter gives better performance for one dataset while the same gives poor results for other datasets. This is because every filter uses separate metrics and so the choice for a filter for a specific dataset is a very complex task. As different filter methods are used in the MFSAC method, so different sub-datasets with different characteristics-based attributes/genes are formed from each dataset. This is shown using Venn diagram in Supplemental Figures 6.11a and 6.11b. Here for Leukemia and Prostate cancer datasets, the first twenty genes, selected by each filter are shown. In case of Leukemia dataset, Relief measure generates non-overlapping gene subset while using other filter metrics presence of a small number of overlapping genes in different gene subsets are observed. In Prostate cancer dataset, Relief generates non-overlapping gene subset and also maximum number of genes are non-overlapping in gene subsets formed by Fisher score, MI (mutual information). From these figures, it is clear that using different filter methods different subsets of genes are selected and different sub datasets are formed. It shows diversity of those filter methods. As a consequence, the base

classifiers prepared on these diverse datasets are become diverse. This diversity increases the power of ensemble classifier.

- Moreover, the genes selected by different filter methos are good biomarker also. In Table 6.13, the top ranked 8 genes selected by MFSAC-EC model are shown for Leukemia and Colon cancer datasets. Among these genes, gene MPO (with column number 1720), CST3 (with column number 1823), ZYX (with column number 4788), CTSD (with column number 2062), CD79A/MB-1(with column number 2583), LYZ (with column number 6738) in Leukemia dataset are important biomarkers as these are selected by different filter methods mentioned in Supplemental Figure 6.11.

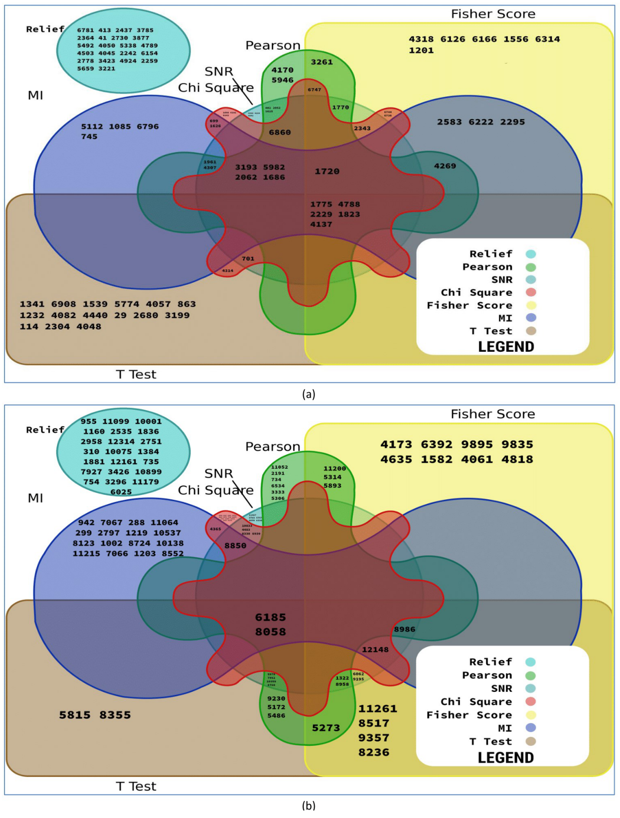


Figure 6.11. Venn Diagram for top 20 genes Selected by each Filter for two Microarray Datasets.:

- In MFSAC, at first, a sub dataset of the most relevant genes is selected by each filter method. Then on each sub dataset, the proposed supervised gene clustering algorithm is applied and a reduced sub dataset of modified attributes/features in the form of augmented cluster representatives is generated. In this method, at the time of cluster formation, genes are augmented based on their supervised information. In other words, such augmentation is considered where it increases the class discrimination power. Thus effectively, the class relevance of any augmented cluster representative is greater than that of any single gene involved in that process. So, this modified sub dataset containing a reduced feature set in the form of augmented cluster representatives is more powerful according to class discrimination power than the sub dataset containing a subset of the most relevant genes. Apart from this, it is well known fact in gene expression data that two genes are functionally similar if they are pattern-based similar (either positively co-expressed or negatively co-expressed) [109]. So, at the time of the augmentation procedure, two types of augmentations are considered here. One is that a gene is added with its original value with the current cluster representative and another one is that the gene is added with its sign-flipped value with the current cluster representative. This is because if the current cluster representative and a gene are positively co-expressed then normal addition is considered but if they are negatively co-expressed then normal addition will hamper the addition process and in that case, sign-flipping of that gene will give proper result. The effect of augmentation with respect to every filter method is shown in Figure 6.12. In Figure 6.12, for the Breast cancer dataset, at the time of supervised cluster formation from each filter generated subset, the original gene, and its corresponding class relevance value, and also augmented gene and its corresponding class relevance are shown. From Figure 8, it is clear that for every filter method the class relevance score of every original gene is increased with respect to that filter after augmentation. In Figure 6.12, different class labels are distinguished by different colors.

- Finally, for each sub dataset with modified attributes in the form of augmented cluster representatives, a classifier is constructed using any existing classifier, and these classifiers are combined using the majority voting technique to form an ensemble classifier (EC). The use of different sub-datasets with optimal gene subsets in the form of augmented cluster representatives and the formation of a classifier for every sub dataset can solve the overfitting problem of any single classifier. This is due to the reason that not all sub-datasets can consistently perform well on all types of cancer datasets (due to inherent characteristics of the datasets), but due to the use of majority voting in ensemble classifiers, this problem can be solved or reduced.

Another outcome of our proposed model is to rank informative genes for every cancer dataset. For this task, the frequency of occurrence of each gene present in the form of augmented cluster representatives in every sub dataset is counted and these genes are ranked according to the counted value to measure the importance of those genes for any specific disease, here cancer. To establish the biological significance of those selected genes for every cancer dataset, their contribution has been confirmed by other existing studies where they are referred already. From these existing studies, it is clear that the selected genes are important for cancer class discrimination and also are important as cancer biomarkers for molecular treatment targets.

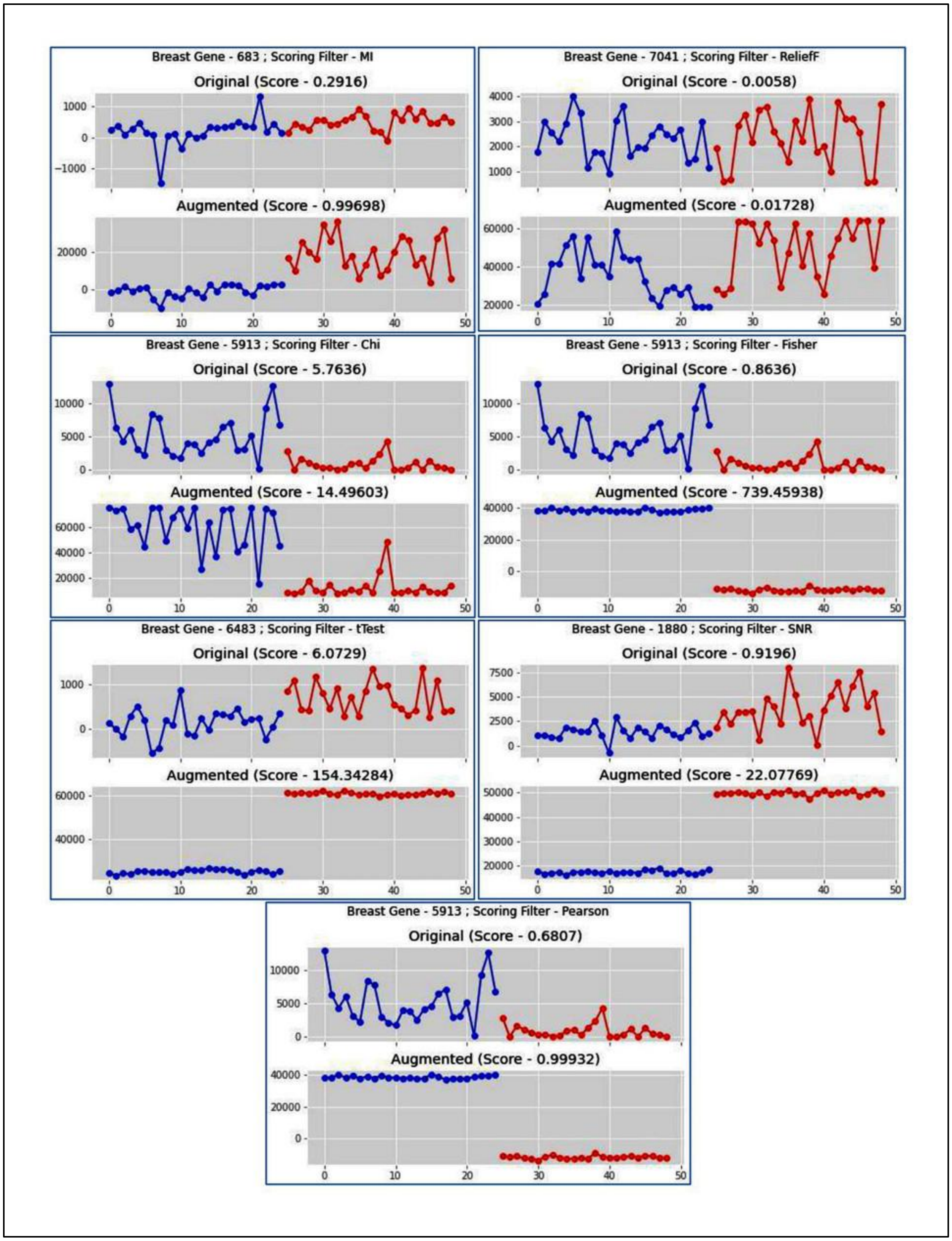


Figure 6.12 Original gene (different class label with different color) and corresponding Augmented gene with respect to different filter methods for Breast Cancer dataset.

6.5 Conclusions

Many machine learning and statistical learning-based classifiers for sample classification already exist in the literature, but these methods are prone to suffer from overfitting due to small sample size problems, class imbalance problems, and the curse of the high dimensionality of microarray data. Although some of the existing methods can mitigate these issues to quite an extent, the problems have still not been satisfactorily overcome. Due to this reason, here a novel feature selection-based ensemble classification model named MFSAC-EC is proposed. It has been shown that the proposed model can handle the above-mentioned issues present in existing models. To check the performance of the proposed MFSAC-EC model, this classifier is applied to test sample classification accuracy in high dimensional microarray gene expression data, a domain that will be beneficial in the field of cancer research. From the experimental results, it has been found that the proposed model outperforms all other well-known existing classification models combined with the different recognized feature selection methods and also the newly developed ensemble classifiers for all types of cancer datasets mentioned here. Apart from this classification task, the proposed model can also rank informative attributes according to their importance. The efficiency of the proposed model in this task is vindicated by finding the most informative genes for Colon cancer and Leukemia cancer datasets using this model. These genes are biologically validated based on other well-known existing studies. Consequently, it is clear that the selected genes are vital for sample class discrimination and are also important biomarkers for molecular treatment targets of deadly diseases.

Chapter 7.

Conclusion and Future Directions

The main objective of this thesis is to develop machine learning techniques based new data mining methodologies, which preprocess and analyze gene expresssion data more accurately. In this regard, certain problems of gene expression data and the solutions of these problems using the proposed methodologies are discussed in this thesis.

In the first work we have developed a novel framework for better neighbourhood formation in KNNimpute and its several versions to improve their prediction accuracy. From experimental results it has been found that prediction accuracy of the *K*NN and its several versions has been greatly improved after using this framework. This method can be applied in RNA expression data, protein expression data for prediction of missing values in future.

In the second work we have developed another imputation method via integrating clustering and numerical method as it is already known that numerical methods are robust although these methods has several limitations. In this work we have tried to overcome all these drawbacks.The proposed model has given better performance than well-konwn existing numerical methods. This method can be applied in RNA expression data, protein expression data and also in other areas apart from bioinformatics for prediction of missing values in future.

In the third work, we have proposed a new framework based on partition based clustering for grouping functionally similar genes in unsupervised manner from microarray gene expression data. This is the first framework where we have eliminated noise using different partition based clustering methods and tries to overcome the drawbacks of different partition based clustering methods. Experimental results show superioty of the proposed method. One limitation of this method is that to check gene gene similarity we have used Euclidean distance. In gene expression data it is already revealed that pattern based similar genes are functionally similar. Using Euclidean distance value wise closer genes can only be considered which may be pattern based similar. So, in our proposed framework we are unable to extract pattern based similar genes only for cluster formation. In future work we will solve this problem.

In the fourth work we have proposed a new ensemble classification model named MFSAC-EC for sample classification in microarray gene expression data. The proposed model has two components. One is gene/feature selection to reduce feature dimenstion and second one sample classification. For feature selection we have applied multi filters based supervised attribute/gene clustering algorithm and for classification we have applied a modified bagging model. The proposed model shows its superioty for different cancer datasets. In future we will modify this model via applying deep learning techniques and apply it for prediction of different diseases from high dimensional microarray gene expression and RNA expression datasets.

Specifically, both HCFPC method and MFSAC method select relevant genes in unsupervised and supervised manner respectively. Pathway analysis of identified relevant genes can give valuable information for Cancer disease as Cancer is still now deadly disease in advanced stage. Analysis of gene expression data to identify relevant biomarker prediction for early diagnosis of neurodegenerative disease can be another area of interest in future. Analysis of multimodal data including gene expression data for identification of relevant biomarkers for cancer, diabetics, and neurodegenerative disease prediction will be the promising area of interest for future research.

Finally, it can be concluded that different machine learning based data mining schemes reported in this thesis can be extended to model other complex problems of bioinformatics and also in other areas. We will do these works in future.

References

1. W J S Diniz and F Canduri, “REVIEW-ARTICLE Bioinformatics: an overview and its applications”, Genetics and molecular research, vol. 16(1), pp. 1-21, 2017. DOI:10.4238/gmr16019645

2. N M Luscombe, D Greenbaum and M Gerstein, “What is bioinformatics? An introduction and overview”, Yearbook of Medical Informatics, vol. 1, pp. 83-99, 2001. PMID: 27701604. DOI:10.1055/s-0038-1638103

3. P Baldi and S Brunak, “Bioinformatics: The Machine Learning Approach”, Cambridge, MA: MIT Press, ISBN 0-262-02506-X, 1998. https://dl.acm.org/doi/10.5555/500801

4. I H Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions”, SN Computer Science, vol. 2, pp. 1-21, 2021. https://doi.org/10.1007/s42979-021-00592-x

5. S Angra and S Ahuja, “Machine learning and its applications: A review”, International Conference on Big Data Analytics and Computational Intelligence (ICBDAC), pp. 57-60, 2021. https://doi.org/10.1109/ICBDACI.2017.8070809

6. A Dey, “Machine Learning Algorithms: A Review”, International Journal of Computer Science and Information Technologies, vol. 7(3), pp. 1174-1179, 2016. https://ijcsit.com/docs/Volume%207/vol7issue3/ijcsit2016070332.pdf

7. X Bai, E R Hancock, R C Wilson and T K Ho, “Special issue on recent advances in statistical, structural and syntactic pattern recognition”, Pattern Recognition Letters, vol. 131, pp. 46-48, 2020. https://doi.org/10.1016/j.patrec.2019.12.004.

8. B Omarov and Y I Cho, “Machine learning based pattern recognition and classification framework development”, 17th International Conference on Control, Automation and Systems (ICCAS), pp. 1-5, 2017. DOI: 10.23919/ICCAS.2017.8204282

9. A W Olthof, P Shouche, F F A IJpma, R H C Koolstra, V M A Stirler, P M A Ooijen and L J Cornelissen, “Machine learning based natural language processing of radiology reports in orthopedic trauma”, Computer Methods and Programs in Biomedicine, vol. 8, pp. 1-9, 2021. https://doi.org/10.1016/j.cmpb.2021.106304

10. A Pratap and N Sardana, “Machine learning-based image processing in materials science and engineering: A review”, Materials today Proceedings, vol. 62(14), pp. 7341-7347, 2022. https://doi.org/10.1016/j.matpr.2022.01.200

11. S J Lee and K Siau, “A Review of Data Mining Techniques”, Industrial Management & Data Systems, vol. 101(1-2), pp. 41-46, 2001. http://dx.doi.org/10.1108/02635570110365989

12. J Han and M Kamber, “Data Mining Concepts and Techniques”, Elsevier, ISBN 978-0-12-381479-1, 2000.

13. H Causton, J Quackenbush and A Brazma, “Microarray Gene Expression Data Analysis: A Beginner’s Guide”, Wiley-Blackwell, ISBN: 978-1-405-10682-5, 2003.

14. A Zhang, “Advanced analysis of gene expression microarray data”, Singapore: World Scientific, ISBN : 9789812566454, 2006.

15. S Selvaraj and J Natarajan, “Microarray data analysis and mining tools”, Bioinformation, vol. 6 (3), pp. 95-99, 2011. https://doi.org/10.6026%2F97320630006095

16. A. Brazma and J Vilo, “Minireview: Gene Expression Data Analysis”, Federation of European Biochemical Societies Letters, vol. 480(1), pp. 17-24, 2000. https://doi.org/10.1016/S0014-5793(00)01772-5

17. D Jiang, C Tang and A Zhang, “Cluster Analysis for Gene Expression Data: A Survey”, IEEE Transactions on Knowledge and Data Engineering, vol. 16(11), pp. 1370-1386, 2004. https://doi.org/10.1109/TKDE.2004.68

18. E Domany, “Cluster Analysis of Gene Expression Data”, Journal of Statistical Physics, vol. 110(3-6), pp. 1117-1139, 2003. https://doi.org/10.1023/A:1022148927580

19. J Oyelade, I Isewon, F Oladipupo, O Aromolaran, E Uwoghiren, F Ameh, M Achas and E Adebiyi, “Clustering Algorithms: Their Application to Gene Expression Data”, Bioinformatics and Biology Insights, vol. 10, pp. 237-253, 2016. doi: 10.4137/BBI.S38316

20. S Tavazoie, J D Hughes, M J Campbell, R J Cho and G M Church, “Systematic Determination of Genetic Network Architecture”, Nature Genetics, vol. 22(3), pp. 281-285, 1999. DOI: 10.1038/10343

21. D P Berrar, C S Downes and W Dubitzky, “Multiclass cancer classification using gene expression profiling and probabilistic neural networks”, Pacific Symposium on Biocomputing, pp. 5-16, 2003. PMID: 12603013, DOI:10.1142/9789812776303_0002

22. H Zhang, C Y Yu and B Singer, “Cell and tumor classification using gene expression data: Construction of forests”, PNAS, vol. 100(7), pp. 4168-4172, 2003. https://doi.org/10.1073/pnas.0230559100

23. R Majji, G Nalinipriya, Ch Vidyadhari and R Cristin, “Jaya Ant lion optimization-driven Deep recurrent neural network for cancer classification using gene expression data”, Medical & Biological Engineering & Computing, vol. 59, pp. 1005–1021, 2021. https://doi.org/10.1007/s11517-021-02350-w

24. J Khan, J S Wei, M Ringnér, L H Saal, M Ladanyi, F Westermann, F Berthold, M Schwab, C R Antonescu, C Peterson and P S Meltzer, “Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks”, Nature Medicine, vol. 7, pp. 673-679, 2001. https://doi.org/10.1038/89044

25. D Singh, P G Febbo, K Ross, D G Jackson, J Manola, C Ladd, P Tamayo, A A Renshaw, A V D'Amico, J P Richie, E S Lander, M Loda, P W Kantoff, T R Golub and W R Sellers, “Gene expression correlates of clinical prostate cancer behavior”, Cancer Cell, vol.1(2), pp. 203–209, 2002. DOI: 10.1016/s1535-6108(02)00030-2

26. K Shedden, J M G Taylor, S A Enkemann, M Tsao, T J Yeatman, W L Gerald, S Eschrich, I Jurisica, T J Giordano, D E Misek, A C Chang, C Q Zhu, D Strumpf, S Hanash, F A Shepherd, K D L Seymour, K Naoki, N Pennell, B Weir, R Verhaak, C L Acosta, T Golub, M Gruidl, A Sharma, J Szoke, M Zakowski, V Rusch, M Kris, A Viale, N Motoi, W Travis, B Conley, V E Seshan, M Meyerson, R Kuick, K K Dobbin, T Lively, J W Jacobson and D G Beer, “Gene expression–based survival prediction in lung adenocarcinoma: A multi-site, blinded validation study”, Natural Medicine, vol. 14(8), pp. 822-827, 2008. https://doi.org/10.1038/nm.1790

27. J D Hainsworth, M S Rubin, D R Spigel, R V Boccia, S Raby, R Quinn and F A Greco, “Molecular gene expression profiling to predict the tissue of origin and direct site-specific therapy in patients with carcinoma of unknown primary site: a prospective trial of the Sarah Cannon research institute”, Journal of Clinical Oncology, vol. 31(2): pp. 217-223, January, 2013. DOI: 10.1200/JCO.2012.43.3755

28. K Manjang, S Tripathi, O Y Harja, M Dehmer, G Glazko and F E Streib, “Prognostic gene expression signatures of breast cancer are lacking a sensible biological meaning”, Nature, Sci Rep 11, pp. 1-18, 2021. https://doi.org/10.1038/s41598-020-79375-y

29. L Kumar and R Greiner, “Gene expression based survival prediction for cancer patients—A topic modeling approach”, PLoS One. vol. 14(11), 2019. doi: 10.1371/journal.pone.0224446

30. S Liu, W Wang, Y Zhao, K Liang and Y Huang, “Identification of Potential Key Genes for Pathogenesis and Prognosis in Prostate Cancer by Integrated Analysis of Gene Expression Profiles and the Cancer Genome Atlas”, Frontiers in Oncology, vol. 10, pp. 1-13, 2020. DOI: 10.3389/fonc.2020.00809

31. A W Liew, N F Law and H Yan, “Missing value imputation for gene expression data, Computational techniques to recover missing data from available information”, Briefings in Bioinformatics, vol. 12(5), pp. 498-513, 2011. https://doi.org/10.1093/bib/bbq080

32. Z M Hira and D F Gillies, “A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data”, Advances in Bioinformatics, Hindwai, vol. 2015, pp. 1-13, 2015. https://doi.org/10.1155/2015/198363

33. F P Shah and V Patel, “A review on feature selection and feature extraction for text classification”, International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), pp. 2264-2268, 2016. DOI:10.1109/WISPNET.2016.7566545

34. R O Duda and P E Hart, “Pattern Classification and Scene Analysis”, New York: Wiley, 1974. DOI:10.2307/1573081

35. P Swain and H Hauska, “The Decision Tree Classifier Design and Potential”, IEEE Transactions on Geoscience and Electronics, vol. 15(3), pp.142–147, 1977. https://doi.org/10.1109/TGE.1977.6498972

36. S Haykin, “Neural Networks. A Comprehensive Foundation (2nd Edition)”, Pearson Education (Singapore), ISBN: 81-7808-300-0, 2005.

37. R Lippmnn, “An Introduction to Computing with Neural Nets”, IEEE ASSP Magazine, vol. 4(2), pp. 4-22, 1987. https://doi.org/10.1109/MASSP.1987.1165576

38. B Scholkpf, K K Sung, C J C Burges, F Girosi, P Niyogi, T Poggio and V Vapnik, “Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers”, IEEE Transactions on Signal Processing, vol. 45(11), pp. 2758–2765, 1997. https://doi.org/10.1109/78.650102

39. D Xu and Y Tian, “A Comprehensive Survey of Clustering Algorithms“, Annals of Data Science, vol. 2, pp. 165-193, 2015. https://doi.org/10.1007/s40745-015-0040-1

40. T Kohonen, “Self-Organizing Maps”, Springer Series in Information Sciences, vol. 30, 1995. https://doi.org/10.1007/978-3-642-56927-2

41. A K H Tung, J Hou and J Han, “Spatial Clustering in the Presence of Obstacles”, Proceedings of 17th International Conference on Data Engineering (ICDE’01), Heidelberg, Germany, pp. 359–367, 2001. https://doi.org/10.1109/ICDE.2001.914848

42. W R Pearson and D J Lipman, “Improved Tools for Biological Sequence Comparison”, Proceedings of National Academy of Science USA, vol. 85(8), pp. 2444–2448, 1998. https://doi.org/10.1073/pnas.85.8.2444

43. S F Altschul, T L Madden, A A Schaffer, J Zhang, Z Zhang, W Miller and D J Lipman, “Gapped-BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs”, Nucleic Acids Research, vol. 25(17), pp. 3389–3402, 1997. https://doi.org/10.1093/nar/25.17.3389

44. D P Clark and L D Russell, “Molecular Biology Made Simple and Fun”, The American Journal of Human Genetics, vol. 60(6), 1568-1568, 1997. http://dx.doi.org/10.2307/4450527

45. P J Russell, “IGenetics”, New York: Benjamin Cummings, ISBN: 0805345531, 2001.

46. W Saenger, “Principles of Nucleic Acid Structure”, New York: Springer-Verlag, 1984. https://doi.org/10.1007/978-1-4612-5190-3

47. S B Needleman and C D Wunch, “A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins”, Journal of Molecular Biology, vol. 48(3), pp. 443–453, 1970. https://doi.org/10.1016/0022-2836(70)90057-4

48. T F Smith and M S Waterman, “Identification of Common Molecular Sequences”, Jornal of Molecular Biology, vol. 147(1), pp. 195–197, 1981. https://doi.org/10.1016/0022-2836(81)90087-5

49. C Notredame and D G Higgins, “SAGA: Sequence Alignment by Genetic Algorithm”, Nucleic Acids Research, vol. 24(8), pp. 1515–1524, 1996. https://doi.org/10.1093%2Fnar%2F24.8.1515

50. P Clote and R Backofen, “Computational Molecular Biology: An Introduction”, Wiley, Wiley Series in Mathematical & Computational Biology, pp. 1-300, ISBN: 978-0-471-87252-8, 2000.

51. R Farber, A Lapedes and K Sirotkin, “Determination of Eukaryotic Protein Coding Regions Using Neural Networks and Information Theory”, Journal of Molecular Biology, vol. 226(2), pp. 471–479, 1992. https://doi.org/10.1016/0022-2836(92)90961-i

52. J Fickett, “Recognition of Protein Coding Regions in DNA Sequences”, Nucleic Acids Research, vol. 10(17), pp. 5303–5318, 1982. https://doi.org/10.1093/nar/10.17.5303

53. J Fickett and C S Tung, “Assessment of Protein Coding Measures”, Nucleic Acids Research, vol. 20(24), pp. 6441–6450, 1992. https://doi.org/10.1093/nar/20.24.6441

54. E Uberbacher and R Mural, “Locating Protein-Coding Regions in Hu- man DNA Sequences by a Multiple Sensor-Neural Network Approach”, Proceedings of National Academy of Science, USA, vol. 88, pp. 11261–11265, 1991. https://doi.org/10.1073%2Fpnas.88.24.11261

55. S Knudsen, “Promoter 2.0: For the Recognition Pol II Promoter Sequences”, Bioinformatics, vol. 15(5), pp. 356–361, 1999. https://doi.org/10.1093/bioinformatics/15.5.356

56. Y Zhu, F Li, D Xiang, T Akutsu, T Song and C Jia, “Computational identification of eukaryotic promoters based on cascaded deep capsule neural networks”, Briefings in Bioinformatics, vol. 22(4), pp. 1-11, 2021. https://doi.org/10.1093%2Fbib%2Fbbaa299

57. Y D Cai and K C Chou, “Artificial Neural Network Model for Predicting HIV Protease Cleavage Sites in Protein”, Advances in Engineering Software”, vol. 29(2), pp. 119–128, 1998. https://doi.org/10.1016/S0965-9978(98)00046-5

58. A Narayanan, X K Wu and Z R Yang, “Mining Viral Protease Data to Extract Cleavage Knowledge”, Bioinformatics, vol. 18, pp. 5–13, 2002. https://doi.org/10.1093/bioinformatics/18.suppl_1.s5

59. N Qian and T J Sejnowski, “Predicting the Secondary Structure of Globular Proteins Using Neural Network Models”, Journal of Molecular Biology, vol. 202(4), pp. 865–884, 1988. https://doi.org/10.1016/0022-2836(88)90564-5

60. P Arrigo, F Giuliano and G Damiani, “Identification of A New Motif on Nucleic Acid Sequence Data Using Kohonen’s Self-Organising Map”, CABIOS, vol. 7(3), pp. 353–357, 1991. https://doi.org/10.1093/bioinformatics/7.3.353

61. P Baldi and P F Baisnee, “Sequence Analysis by Additive Scales: DNA Structure for Sequences and Repeats of All Lengths”, Bioinformatics, vol. 16, pp. 865–889, 2000. https://doi.org/10.1093/bioinformatics/16.10.865

62. A S Rifaioglu, T Doğan, M J Martin, R C Atalay and V Atalay, “ DEEPred: Automated Protein Function Prediction with Multi-task Feed-forward Deep Neural Networks”, Nature, Scientific Reports, vol. 9(7344), pp. 1-16, 2019. https://doi.org/10.1038/s41598-019-43708-3

63. Z R Yang, “Prediction of Caspase Cleavage Sites Using Bayesian Bio-Basis Function Neural Networks”, Bioinformatics, vol. 21(9), pp. 1831–1837, 2005. https://doi.org/10.1093/bioinformatics/bti281

64. Z R Yang and K C Chou, “Predicting the Linkage Sites in Glycoproteins Using Bio-Basis Function Neural Networks”, Bioinformatics, vol. 20(6), pp. 903–908, 2004. https://doi.org/10.1093/bioinformatics/bth001

65. Z R Yang, R Thomson, P McNeil and R Esnouf, “RONN: Use of the Bio-Basis Function Neural Network Technique for the Detection of Natively Disordered Regions in Proteins”, Bioinformatics, vol. 21(16), pp. 3369–3376, 2005. https://doi.org/10.1093/bioinformatics/bti534

66. Z R Yang and R Thomson, “Bio-Basis Function Neural Network for Prediction of Protease Cleavage Sites in Proteins”, IEEE Transactions on Neural Networks, vol. 16(1), pp. 263–274, 2005. https://doi.org/10.1109/tnn.2004.836196

67. Z R Yang, “Biological Application of Support Vector Machines”, Briefings in Bioinformatics, vol. 5(4), pp. 328–338, 2004. https://doi.org/10.1093/bib/5.4.328

68. A Kanhere and M Bansal, “An assessment of three dinucleotide parameters to predict DNA curvature by quantitative comparison with experimental data”. Nucleic Acids Research, vol. 31(10), pp. 2647-58, 2003. https://doi.org/10.1093%2Fnar%2Fgkg362.

69. C C Chen and Y M Chan, “REDfold: accurate RNA secondary structure prediction using residual encoder-decoder network”, BMC Bioinformatics, vol. 24(122), pp. 1-13, 2023. https://doi.org/10.1186/s12859-023-05238-8

70. K C Wiese and E Glen, “A Permutation-Based Genetic Algorithm for the RNA Folding Problem: A Critical Look at Selection Strategies, Crossover operators, and Representation issues”, Biosystems, vol. 72(1-2), pp. 29–41, 2003. https://doi.org/10.1016/S0303-2647(03)00133-3

71. M Zuker and P Striegler, “Optimal Computer Folding of Large RNA Secondary Sequences Using Thermodynamics and Auxiliary Information”, Nucleic Acids Research, vol. 9(1), pp. 133–148, 1981. https://doi.org/10.1093%2Fnar%2F9.1.133

72. Z Sun, X Xia, Q Guo and D Xu, “Protein Structure Prediction in a 210-Type Lattice Model: Parameter Optimization in the Genetic Algorithm Using Orthogonal Array”, Journal of Protein Chemistry, vol. 18(1), pp. 39–46, 1999. https://doi.org/10.1023/A:1020643331894

73. R Unger and J Moult, “On the Applicability of Genetic Algorithms to Protein Folding”, Proceedings Hawaii International Conference System Sciences, vol. 1, pp. 715–725, 1993. https://doi.org/10.1109/HICSS.1993.270669

74. J T Pedersen and J Moult, “Protein folding simulations with genetic algorithms and a detailed molecular description”, Journal of Molecular Biology, vol. 269(2), pp. 240-259, 1997. https://doi.org/10.1006/jmbi.1997.1010.

75. P Chou and G Fasman, “Prediction of the Secondary Structure of Proteins from Their Amino Acid Sequence”, Advances in Enzymology and Related areas of Molecular Biology, vol. 47, pp. 145– 148, 1978. https://doi.org/10.1002/9780470122921.ch2

76. J Garnier, J F Gibrat and B Robson, “GOR Method for Predicting Protein Secondary Structure from Amino Acid Sequence” Methods in Enzymology, vol. 266, pp. 540–553, 1996. https://doi.org/10.1016/s0076-6879(96)66034-0

77. S K Riis and A Krogh, “Improving Prediction of Protein Secondary Structure Using Structured Neural Networks and Multiple Sequence Alignments”, Journal of Computational Biology, vol. 3, pp. 163–183, 1996. https://doi.org/10.1089/cmb.1996.3.163

78. S Salzberg and S Cost, “Predicting Protein Secondary Structure with a Nearest-Neighbor Algorithm”, Journal of Molecular Biology, vol. 227(2), pp. 371–374, 1992. https://doi.org/10.1016/0022-2836(92)90892-n

79. J M Yang and C Y Kao, “A Family Competition Evolutionary Algorithm for Automated Docking of Flexible Ligands to Proteins”, IEEE Transactions on Information Technology in Biomedicine, vol. 4(3), pp. 225–237, 2000. https://doi.org/10.1109/4233.870033

80. A Skourikhine, “Phylogenetic Tree Construction Using Self-Adaptive Genetic Algorithm”, IEEE International Symposium on Bioinformatics and Biomedical Engineering, vol. 1, pp. 129–134, 2000. http://dx.doi.org/10.1109/BIBE.2000.889599

81. T Hawkins, M Chitale, S Luban and D Kihara, “PFP: Automated Prediction of Gene Ontology Functional Annotations with Confidence Scores Using Protein Sequence Data”, Proteins, vol. 74(3), pp. 566-582, 2008. https://doi.org/10.1002/prot.22172

82. F Suna, W Zhanga, G Xionga, M Yanb, Q Qian, J Lia and Y Wang, “Identification and Functional Analysis of the MOC1 Interacting Protein 1”, Journal of Genetics and Genomics, vol. 37(1), pp. 69–77, 2010. https://doi.org/10.1016/s1673-8527(09)60026-6

83. P Vizn, S STena, G AVizn, M Soler, R Messeguer, M D Pujol, W N P Lee and M Cascante, "Characterization of the Metabolic Changes underlying Growth Factor Angiogenic Activation: Identification of New Potential Therapeutic Targets", Carcinogenesis, vol. 30(6), pp. 946–952, 2009. https://doi.org/10.1093/carcin/bgp083

84. Y Fu, Z Ling, H Arabnia and Y Deng, "Current trend and development in bioinformatics research", BMC Bioinformatics, vol. 21(9), pp. 1-3, 2020. https://doi.org/10.1186/s12859-020-03874-y

85. M Acunzo, G Romano, D Wernicke and C M Croce, "MicroRNA and cancer –A brief overview", Advances in Biological Regulation, vol. 57, pp. 1–9, 2015. https://doi.org/10.1016/j.jbior.2014.09.013

86. S M Hammond, "An overview of MicroRNAs", Advanced Drug Delivery Reviews, vol.87, pp. 3-14, 2015. https://doi.org/10.1016/j.addr.2015.05.001

87. K Moorthy, M S Mohamad and S Deris, "A review on missing value imputation algorithms for microarray gene expression data", Current Bioinformatics, vol. 9(1), pp. 18-22, 2014. http://dx.doi.org/10.2174/1574893608999140109120957

88. O Alter, P Brown and D Botstein, "Singular value decomposition for genome-wide expression data processing and modeling", PNAS, vol. 97(18), pp. 10101-10106, 2000. https://doi.org/10.1073/pnas.97.18.10101

89. S Oba, M A Sato, I Takemasa, M Monden, K Matsubara and S Ishii, "A Bayseian Missing Value Estimation Method for Gene Expression Profile Data", Bioinformatics, vol. 19(16), pp. 2088-2096, 2003. https://doi.org/10.1093/bioinformatics/btg287

90. O Tryosanka, M Cantor, G Sherlock, P Brown, T Hastie, R Tibshirani, D Bostein and R B Altman, "Missing Value Estimation Methods for DNA Microarrays", Bioinformatics, vol. 17(6), pp. 520-525, 2001. https://doi.org/10.1093/bioinformatics/17.6.520

91. K Y Kim, B J Kim and G S Yi, "Reuse of Imputed Data in Microarray Analysis Increases Imputation Efficiency", BMC Bioinformatics, vol. 5, pp. 1-9, 2004. https://doi.org/10.1186/1471-2105-5-160

92. L P Bras and J C Menezes, "Improving Cluster-based Missing Value Estimation of DNA Microarray Data", Biomolecular Engineering, vol. 24(2), pp. 273–282, 2007. https://doi.org/10.1016/j.bioeng.2007.04.003

93. K O Cheng , B Law and W C Siu, "Use of biclustering for missing value imputation in gene expression data", Artificial Intelligence Research, vol. 2(2), pp. 96-108, 2013. http://dx.doi.org/10.5430/air.v2n2p96

94. T H Bo, B Dysvik and I Jonassen, "LSimpute: accurate estimation of missing values in microarray data with least squares methods", Nucleic Acids Research, vol. 32(3), pp. 1-8, https://doi.org/10.1093/nar/gnh026

95. H Kim, G H Golub and H Park, “Missing value Estimation for DNA Microarray Expression Data: Local Least Square Imputation”, Bioinformatics, vol. 21(2), pp. 187-198, 2005. https://doi.org/10.1093/bioinformatics/bth499

96. Z Cai, M Heydari and G Lin, “Iterated local least squares microarray missing value imputation”, Journal of Bioinformatics and Computational Biology, vol. 4(5), pp. 935-957, 2006. https://doi.org/10.1142/s0219720006002302

97. X Zhang, X Song, H Wang and H Zhang, “Sequential local least squares imputation estimating missing value of microarray data”, Computers in Biology and Medicine, vol. 38(10), pp. 1112-1120, 2008. https://doi.org/10.1016/j.compbiomed.2008.08.006

98. M S Sehgal, I Gondal and L S Dooley, “Collateral missing value imputation: a new robust missing value estimation algorithm for microarray data”, Bioinformatics, vol. 21(10), pp. 2417-2423, 2005. https://doi.org/10.1093/bioinformatics/bti345

99. I Scheel, M Aldrin, I K Glad, R Sorum, H Lyng and A Frigessi, “The influence of missing value imputation on detection of differentially expressed genes from microarray data”, Bioinformatics, vol. 21(23), pp. 4272-4279, 2005. https://doi.org/10.1093/bioinformatics/bti708

100. K O Cheng, N F Law and W C Siu, “Iterative bicluster-based least square framework for estimation of missing values in microarray gene expression data”, Pattern Recognition, vol. 45(4), pp. 1281-1289, 2012. https://doi.org/10.1016/j.patcog.2011.10.012

101. X Pan, Y Tian, Y Huang and H Shen, “Towards better accuracy for missing value estimation of epistatic miniarray profiling data by a novel ensemble approach”, Genomics, vol. 97(5), pp. 257–264, 2011. https://doi.org/10.1016/j.ygeno.2011.03.001

102. J Tuikkala, L Elo, O S Nevalainen and T Aittokallio, “Improving missing value estimation in microarray data with gene ontology”, Bioinformatics, vol. 22(5), pp. 566-572, 2006. https://doi.org/10.1093/bioinformatics/btk019

103. D W Kim, K Y Lee, K H Lee and D Lee, “Towards clustering of incomplete microarray data without the use of imputation”, Bioinformatics, vol. 23(1), pp. 107-113, 2007. https://doi.org/10.1093/bioinformatics/btl555

104. J Hu, H Li, M S Waterman and X J Zhou, “Integrative missing value estimation for microarray data”, BMC Bioinformatics, vol. 7, article 449, pp. 1-14, 2006. https://doi.org/10.1186/1471-2105-7-449

105. V G Tusher, R Tibshirani and G Chu, “Significance analysis of microarrays applied to the ionizing radiation response”, PNAS, vol. 98(9), pp. 5116–5121, 2001. https://doi.org/10.1073/pnas.091062498

106. R Tibshirani, T Hastie, D Narasimhan and G Chu, “Diagnosis of multiple cancer types by shrunken centroids of gene expression”, PNAS, vol. 99(10), pp. 6567–6572, 2002. https://doi.org/10.1073/pnas.082099299

107. M K Kerr, M Martin and G A Churchill, “Analysis of variance for gene expression microarray data”, Journal of Computational Biology, vol. 7(6), pp. 819–837, 2000. https://doi.org/10.1089/10665270050514954

108. A Bhattacharya and R K De, “Divisive Correlation Clustering Algorithm (DCCA) for grouping of genes: detecting varying patterns in expression profiles”, Bioinformatics, vol. 24(11), pp. 1359-1366, 2008. https://doi.org/10.1093/bioinformatics/btn133

109. C Das, S Bose, M Chattopadhyay and S Chattopadhyay, “A novel distance based iterative sequential KNN algorithm for estimation of missing values in microarray gene expression data”, International Journal of Bioinformatics Research and Application(IJBRA), vol. 12(4), pp. 312-342, 2016. https://doi.org/10.1504/IJBRA.2016.080719

110. S Bose, C Das, K Ghosh, M Chattopadhyay and S Chattopadhyay, “A Framework for Neighborhood Configuration to Improve the KNN based Imputation Algorithms on Microarray Gene Expression Data”, International Journal of Bioinformatics Research and Applications (IJBRA), Inderscience, vol. 1, pp. 141-190, 2021. https://doi.org/10.1504/IJBRA.2022.124989

111. Y Cheng and G M Church, “Biclustering of Expression Data”, Proceedings of Eighth Int’l Conference on Intelligent Systems for Molecular Biology (ISMB ’00), pp. 93-103, 2000. https://pubmed.ncbi.nlm.nih.gov/10977070/

112. J S A Ruiz, “Shifting and scaling patterns from gene expression data”, Bioinformatics, vol. 21(20), pp. 3840-3845, 2005. https://doi.org/10.1093/bioinformatics/bti734

113. J Tuikkala, L L Elo, O S Nevalainen and T Aittokallio, “Missing value imputation improves clustering and interpretation of gene expression microarray data”, BMC Bioinformatics, vol. 9, pp. 1-14, 2008. https://doi.org/10.1186/1471-2105-9-202

114. P T Spellman, G Sherlock, M Q Zhang, V R Iyer, K Anders, M B Eisen, P O Brown, D Botstein and B Futcher, “Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization”, Molecular Biology of the Cell, vol. 9(12), pp. 3273-97, 1998. https://doi.org/10.1091/mbc.9.12.3273

115. A P Gasch, P T Spellman, C M Kao, O C Harel, M B Eisen, G Storz, D Botstein and P O Brown, “Genomic expression programs in the response of yeast cells to environmental changes”, Molecular Biology of the Cell, vol. 11(12), pp. 4241-4257, 2000. https://doi.org/10.1091/mbc.11.12.4241

116. D T Ross, U Scherf, M B Eisen, C M Perou, C Rees, P Spellman, V Iyer, S S Jeffrey, M V Rijn, M Waltham, A Pergamenschikov, J C Lee, D Lashkari, D Shalon, T G Myers, J N Weinstein, D Botstein and P O Brown, ”Systematic variation in gene expression patterns in human cancer cell Lines”, Nature Genetics, vol. 24(3), pp. 227-35, 2000. https://doi.org/10.1038/73432

117. T R Golub, D K Slonim, P Tomayo, C Huard, M Gaasenbeek, J P Mesirov, H Coller, M L Loh, J R Downing, M A Caligiuri, C D Bloomfield and E S Lander, “Molecular

classification of cancer: class discovery and class prediction by gene expression monitoring", Science, vol. 286, pp.531–537. https://doi.org/10.1126/science.286.5439.531

118. T Yu, H Peng and W Sun, "Incorporating nonlinear relationships in microarray missing value imputation", IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 8(3), pp. 723-731, 2011. https://doi.org/10.1109/tcbb.2010.73

119. M K Kerr, M Martin and G A Churchill, "Analysis of variance for gene expression microarray data", Journal of Computational Biology, vol. 7, pp. 819–837, 2000. https://doi.org/10.1089/10665270050514954

120. D N Baldwin, V Vanchinathan, P O Brown and J A Theriot, "A gene expression program reflecting the innate immune response of cultured intestinal epithelial cells to infection by Listeria monocytogenes", Genome Biology, vol. 4(R2), pp. 1-14, 2003. https://doi.org/10.1186/gb-2002-4-1-r2

121. H Yashimoto, K Saltsman, A P Gasch, L X Hong, N Ogawa, D Bostein, P O Brown and M S Cyert, "Genome-wide analysis of gene expression regulated by the calcineurin/Crz1p signaling pathway in Saccharomyces cerevisiae", Journal of Biology and Chemistry, vol. 277(34), pp. 31079-31088, 2002. https://doi.org/10.1074/jbc.m202718200

122. T V D Bulcke, K V Leemput, B Naudts, P V Remortel, H Ma, A Verschoren, B D Moor and K Marchal, "SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms", BMC Bioinformatics, vol. 7, article 43, pp. 1-12, 2006. https://doi.org/10.1186/1471-2105-7-43

123. G N Brock, J R Shaffer, R E Blakesley, M J Lotz and G C Tseng, "Which missing value imputation method to use in expression profiles: a comparative study and two selection schemes", BMC Bioinformatics, vol. 9, article 12, pp. 1-12, 2008. https://doi.org/10.1186/1471-2105-9-12

124. H Kuhn, "The Hungarian method for the assignment problem", Naval Research Logistics Quarterly, vol. 2(1-2), pp. 83-97, 1955. https://doi.org/10.1002/nav.3800020109

125. X Gan, A W Liew and H Yan, "Discovering biclusters in gene expression data based on high-dimensional linear geometries", BMC Bioinformatics, vol. 9, article 209, pp. 1-15, 2008. https://doi.org/10.1186/1471-2105-9-209

126. C Das and P Maji, "Possibilistic Biclustering Algorithm for Discovering Value-Coherent Overlapping ∂-Biclusters", International Journal of Machine Learning and Cybernetics, Springer, vol. 6(1), pp. 95-107, January 2015. http://dx.doi.org/10.1007%2Fs13042-013-0211-3

127. W C Tjhi and L Chen, "A partitioning based algorithm to fuzzy co-cluster documents and words", Pattern Recognition Letters, vol. 27(3), pp. 151-159, 2006. https://doi.org/10.1016/j.patrec.2005.07.012

128. F Meng, C Cai and H Yan, “A Bicluster-Based Bayesian Principal Component Analysis Method for Microarray Missing Value Estimation”, IEEE journal of biomedical and health informatics, vol. 18(3), pp. 863-871, 2014. https://doi.org/10.1109/jbhi.2013.2284795

129. C Das, S Bose, S Chattopadhyay, M Chattopadhyay and A Hossain, “A Bicluster-based Sequential Interpolation Imputation Method for Estimation of Missing Values in Microarray Gene Expression Data", Current Bioinformatics, Bentham Science, vol. 12(2), pp. 118-130, 2017. https://doi.org/10.2174/1574893612666170106102019

130. J H Mathews, “Numerical methods for mathematics, science and engineering” Edition 2, Prentice-Hall, U. S. A, ISBN: 812030845X, 9788120308459, 1992.

131. P Maji and S Paul, “Rough-Fuzzy Clustering for Grouping Functionally Similar Genes from Microarray Data”, IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 10(2), pp. 286-299, 2013. https://doi.org/10.1109/tcbb.2012.103

132. D Jiang, J Pei and A Zhang, “DHC: A Density-Based Hierarchical Clustering Method for Time-Series Gene Expression Data,” Proceedings of IEEE Third Int’l Symposium on Bioinformatics and Bioengineering, pp. 393-400, 2003. https://doi.org/10.1109/BIBE.2003.1188978

133. B Karmakar, S Das, S Bhattacharya, R Sarkar and I Mukhopadhyay, “Tight clustering for large datasets with an application to gene expression data”, Scientific Reports, Nature Research, vol. 9, article 3053, pp. 1-12, 2019. https://doi.org/10.1038/s41598-019-39459-w

134. F Luo, K Tang and L Khan, “Hierarchical clustering of gene expression data”, , 2003. Proceedings of. Third IEEE Symposium on Bioinformatics and Bioengineering, pp. 328–35, 2003. https://doi.org/10.1109/BIBE.2003.1188970

135. J Herrero, A Valencia and J Dopazo, “A hierarchical unsupervised growing neural network for clustering gene expression patterns”, Bioinformatics, vol. 17(2), pp. 126-136, 2001. https://doi.org/10.1093/bioinformatics/17.2.126

136. Y Lu, S Lu, F Fotouhi, Y Deng and S J Brown, “Incremental genetic K-means algorithm and its application in gene expression data analysis”, BMC Bioinformatics, vol. 5, article 172, pp. 1-10, 2004. https://doi.org/10.1186/1471-2105-5-172

137. Y Lu, S Lu, F Fotouhi, Y Deng and S J Brown, “FGKA: A fast genetic k-means clustering algorithm”, Proceedings of the ACM Symposium on Applied Computing, pp. 622-623, 2004. https://doi.org/10.1145/967900.968029

138. Z Du, Y Wang and Z Ji, “PK-means: A new algorithm for gene clustering”, Computational Biology and Chemistry, vol. 32(4), pp. 243-247, 2008. https://doi.org/10.1016/j.compbiolchem.2008.03.020

139. A P Gasch and M B Eisen, “Exploring the Conditional Coregulation of Yeast Gene Expression through Fuzzy K-Means Clustering”, Genome Biology, vol. 3(11), pp. 1-22, 2002. https://doi.org/10.1186/gb-2002-3-11-research0059

140. L Fu and E Medico, “FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data”, BMC Bioinformatics, vol. 8, article 3, pp. 1-15, 2007. https://doi.org/10.1186/1471-2105-8-3

141. S Nasser, R Alkhaldi and G Vert, “A modified fuzzy k-means clustering using expectation maximization”, IEEE International Conference on Fuzzy Systems, pp. 231-235, 2006. https://doi.org/10.1109/FUZZY.2006.1681719

142. L Tari, C Baral and S Kim, “Fuzzy c-means clustering with prior biological knowledge”, Journal of Biomedical Informatics, vol. 42(1), pp. 74-81, 2009. https://doi.org/10.1016/j.jbi.2008.05.009

143. D Dembele and P Kastner, “Fuzzy C-Means Method for Clustering Microarray Data”, Bioinformatics, vol. 19(8), pp. 973-980, 2003. https://doi.org/10.1093/bioinformatics/btg119

144. Y Wang, M Angelova and A Ali, "Fuzzy clustering of time seri es gene expression data with cubic-spline", Journal of Biosciences and Medicines, vol. 1, pp. 16-21, 2013. http://dx.doi.org/10.4236/jbm.2013.13004

145. P Maji and S K Pal, “Rough Set Based Generalized Fuzzy C-Means Algorithm and Quantitative Indices,” IEEE Transactions on System, Man, and Cybernetics, Part B: Cybernetics, vol. 37(6), pp. 1529-1540, Dec. 2007. https://doi.org/10.1109/TSMCB.2007.906578

146. R Sharan, A M Katz and R Shamir, “ CLICK and EXPANDER: a system for clustering and visualizing gene expression data”, Bioinformatics, vol. 19(14). pp. 1787-1799, 2003. https://doi.org/10.1093/bioinformatics/btg232

147. B A Jamous and S Kelly, “Clust: automatic extraction of optimal co-expressed gene clusters from gene expression data”, Genome Biology, vol. 19, article 172, pp. 1-11, 2018. https://doi.org/10.1186/s13059-018-1536-8

148. R Krishnapuram and J M Keller, “A Possibilistic Approach to Clustering”, IEEE Transactions on Fuzzy Systems, vol. 1(2), pp. 98-110, 1993. https://doi.org/10.1109/91.227387

149. P Tamayo, D Slonim, J Mesirov, Q Zhu, S Kitareewan, E Dmitrovsky, E S Lander and T R Golub, “Interpreting Patterns of Gene Expression with Self-Organizing Maps: Methods and Application to Hematopoietic Differentiation”, PNAS, vol. 96(6), pp. 2907-2912, 1999. https://doi.org/10.1073/pnas.96.6.2907

150. P J Rousseeuw, “Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis”, Journal of Computational and Applied Mathematics, vol. 20(1), pp. 53-65, 1987. https://doi.org/10.1016/0377-0427(87)90125-7

151. J C Bezdek and N R Pal, “Some New Indexes for Cluster Validity”, IEEE Transactions on System, Man, and Cybernetics, Part B:Cybernetics, vol. 28(3), pp. 301-315, 1988. https://doi.org/10.1109/3477.678624

152. E I Boyle, S Weng, J Gollub, H Jin, D Botstein, J M Cherry and G Sherlock, “GO::Term Finder Open Source Software for Accessing Gene Ontology Information and Finding Significantly Enriched Gene Ontology Terms Associated with a List of Genes”, Bioinformatics, vol. 20(18), pp. 3710-3715, 2004. https://doi.org/10.1093/bioinformatics/bth456

153. T S Reinel, O A Simon, R C Victor, S B Vanesa, R S J Luis and J V C Felipe, “A comparative study of machine learning and deep learning algorithms to classify cancer types based on microarray gene expression data”, PeerJ Computer Science, vol. 6, e270, pp. 1-22, 2020. https://doi.org/10.7717/peerj-cs.270

154. M A Hambali, T O Oladele and K S Adewole, “Microarray cancer feature selection: Review, challenges and research directions”, International Journal of Cognitive Computing in Engineering, vol. 1, pp. 78-97. 2020. https://doi.org/10.1016/j.ijcce.2020.11.001

155. NIH 2019. National Cancer Institute (NCI), cancer statistics. Available from: https://www.cancer.gov/news-events/press-releases/2019

156. K Kourou, T P Exarcos, K P Exarcos, M V Karmouzis and D Fotaidis, “Machine learning applications in cancer prognosis and prediction”, Computational and Structural Biotechnology Journal, vol. 13, pp. 8–17, 2015. https://doi.org/10.1016/j.csbj.2014.11.005

157. M Colozza, F Cardoso, C Sotiriou, D Larismont and M J Piccart, “Bringing molecular prognosis and prediction to the clinic”, Clinical Breast Cancer, vol. 6(1), pp. 61-76, 2005. https://doi.org/10.3816/cbc.2005.n.010

158. LD Greller and F L Tobin, “Detecting selective expression of genes and proteins”, Genome Research, vol. 9(3), pp. 282–296, 1999. https://genome.cshlp.org/content/9/3/282.full.pdf

159. J Liu, Y Cheng, X Wang, L Zhang and Z J Wang, “Cancer Characteristic Gene Selection via Sample Learning Based on Deep Sparse Filtering”, Scientific Reports, Nature, vol. 8, article 8270, pp. 1-13, 2018. https://doi.org/10.1038/s41598-018-26666-0

160. Z Li, W Xie and T Liu, “Efficient feature selection and classification for microarray data”, Plos one, vol. 13(8), e0202167, pp. 1-21, 2018. https://doi.org/10.1371/journal.pone.0202167

161. Q Liu, A H Sung, Z Chen, J Liu, I Chen, M Qiao, Z Wang, X Huang and Y Deng, “Gene selection and classification for cancer microarray data based on machine learning and similarity measures”, BMC Genomics, vol. 12(5), S1, pp. 1-12, 2011. https://doi.org/10.1186/1471-2164-12-S5-S1

162. M J Pilling, A Henderson and P Gardner, “Quantum Cascade Laser Spectral Histopathology: Breast Cancer Diagnostics Using High Throughput Chemical Imaging”,

Analytical Chemistry, vol. 89(14), pp. 7348-7355, 2017. https://doi.org/10.1021/acs.analchem.7b00426

163. A I Su, J B Welsh, L M Sapinoso, S G Kern, P Dimitrov, H Lapp, P G Schultz, S M Powell, C A Moskaluk, H F Frierson and G M Hampton, “Molecular classification of human carcinomas by use of gene expression signatures”, Cancer Research, vol. 61, pp. 7388–7393, 2001. https://aacrjournals.org/cancerres/article/61/20/7388/508137/Molecular-Classification-of-Human-Carcinomas-by

164. A I Swan, A Mobasheri, D Allaway, S Liddell and, J Bacardit, “Application of Machine Learning to Proteomics Data: Classification and Biomarker Identification in Postgenomics Biology”, OMICS, vol. 17(12), pp. 595–610, 2013. https://doi.org/10.1089/omi.2013.0017

165. A J Chin, A Mirzal, H Haron and H N A Hamed, “Supervised, Unsupervised and Semi-supervised Feature Selection: A Review on Gene Selection”, IEEE Transactions on Computational Biology and Bioinformatics, vol. 13(5), pp. 971-989, 2015. https://doi.org/10.1109/TCBB.2015.2478454

166. M Dashtban and M Balafar, “Gene selection for microarray cancer classification using a new evolutionary method employing artificial intelligence concepts”, Genomics, vol. 109(2), pp. 91-107, 2017. https://doi.org/10.1016/j.ygeno.2017.01.004

167. C Ding and H Peng, “Minimum redundancy feature selection from microarray gene expression data”, Journal of Bioinformatics and Computational Biology, vol. 3(2), pp. 185–205, 2005. https://doi.org/10.1142/s0219720005001004

168. V Elyasigomari, D A Lee , H R C Screen and M H Shaheed, “Development of a two-stage gene selection method that incorporates a novel hybrid approach using the cuckoo optimization algorithm and harmony search for cancer classification”, Journal of Biomedical Informatics, vol. 67, pp. 11-20, 2017. https://doi.org/10.1016/j.jbi.2017.01.016

169. T S Furey, N Cristianini, N Duffy, D W Bednarski, M Schummer and D Haussler, “Support Vector Machine Classification and Validation of Cancer Tissue Samples Using Microarray Expression Data”, Bioinformatics, vol. 16(10), pp. 906-914, 2000. https://doi.org/10.1093/bioinformatics/16.10.906

170. T R Golub, D K Slonim, P Tamayo, C Huard, M Gaasenbeek, J P Mesirov, H Coller, H L Loh, J R Downing, M A Caligiuri, C D Bloomfield and E S Lander, “Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring”, Science, vol. 286(5439), pp. 531-537, 1999. https://doi.org/10.1126/science.286.5439.531

171. A Nada and H Alshamlan, “A survey on hybrid feature selection methods in microarray gene expression data for cancer classification”, IEEE Access, vol. 7, pp. 78533-78548, 2019. https://doi.org/10.1109/ACCESS.2019.2922987

172. C Das, S Bose, A Banerjee, S Dutta, K Ghosh and M Chattopadhyay, “Comparative Performance Analysis of Different Measures to Select Disease Related Informative Genes

from Microarray Gene Expression Data", International Conference on Innovation in Modern Science and Technology (ICIMSAT-2019), Springer, LAIS, vol. 12, pp. 912-922, 2019. https://doi.org/10.1007/978-3-030-42363-6_105

173. W H Au, K C C Chan, A K C Wong and Y Wang, "Attribute clustering for grouping, selection, classification of gene expression data", IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 2(2), pp. 83–101, 2005. https://doi.org/10.1109/tcbb.2005.17

174. M Dettling and P Buhlmann, "Supervised Clustering of Genes", Genome Biology, vol. 3(12), pp. 1-15, 2002. https://doi.org/10.1186/gb-2002-3-12-research0069

175. T Hastie, R Tibshirani, M B Eisen, A Alizadeh, R Levy, L Staudt, W C Chan, D Botstein and P Brown, "'Gene Shaving' as a Method for Identifying Distinct Sets of Genes with Similar Expression Patterns", Genome Biology, vol. 1(2), pp. 1-21, 2000. https://doi.org/10.1186/gb-2000-1-2-research0003

176. T Hastie, R Tibshirani, D Botstein and P Brown, "Supervised Harvesting of Expression Trees", Genome Biology, vol. 2(1), pp. 1-12, 2001. https://doi.org/10.1186%2Fgb-2001-2-1-research0003

177. P Maji and C Das, "Relevant and Significant Supervised Gene Clusters for Microarray Cancer Classification", IEEE Transactions on Nanobioscience, vol. 11(2), pp. 161-168, 2012. https://doi.org/10.1109/tnb.2012.2193590

178. V B Canedo, N S Maroño and A A Betanzos, "An ensemble of filters and classifiers for microarray data classification", Pattern Recognition, vol. 45(1), pp. 531-539, 2012. https://doi.org/10.1016/j.patcog.2011.06.006

179. S Ghorai, A Mukherjee, S Sengupta and P K Dutta, "Cancer Classification from Gene Expression Data by NPPC Ensemble", IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 8(3), pp. 659-671, 2011. https://doi.org/10.1109/TCBB.2010.36

180. S Nagi and D K Bhattacharyya, "Classification of microarray cancer data using ensemble approach", Network Modelling Analysis in Health Informatics and Bioinformatics, vol. 2, pp. 159-17, 2013. https://doi.org/10.1007/s13721-013-0034-x

181. C W Wang, "New Ensemble Machine Learning Method for Classification and Prediction on Gene Expression Data", Proceedings of the 28th IEEE EMBS Annual International Conference New York City, USA, vol. 2006, pp. 3478-3481, 2006. https://doi.org/10.1109/iembs.2006.259893

182. S L Wang, X L Li and J Fang, "Finding minimum gene subsets with heuristic breadth-first search algorithm for robust tumor classification", BMC Bioinformatics, vol. 13, article 178, pp. 1-26, 2012. https://doi.org/10.1186/1471-2105-13-178

183. P Yang, B B Zhou, Z Zhang and A Y Zomaya, “A multi-filter enhanced genetic ensemble system for gene selection and sample classification of microarray data”, BMC Bioinformatics, vol. 11, article S5, pp. 1-12, 2010. https://doi.org/10.1186/1471-2105-11-S1-S5

184. T G Dietterich, “An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization”, Machine Learning, vol. 40(2), pp. 139–157, 2000. https://doi.org/10.1023/A:1007607513941

185. J Błaszczyński, J Stefanowski and Ł Idkowiak, “Extending Bagging for Imbalanced Data”, Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, AISC vol. 226, pp. 269-278, 2013. https://doi.org/10.1007/978-3-319-00969-8_26

186. Q Gu, Z Li and J Han,”Generalized Fisher Score for Feature Selection”, UAI'11: Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, pp. 1-8, 2011. https://doi.org/10.48550/arXiv.1202.3725

187. N Zhou and L Wang, “A Modified T-test Feature Selection Method and Its Application on the Hap Map Genotype Data”, Genomics Proteomics & Bioinformatics, vol. 5(3-4), pp. 242-249, 2007. https://doi.org/10.1016/S1672-0229(08)60011-X

188. Y Leung and Y Hung, “A Multiple-Filter-Multiple-Wrapper Approach to Gene Selection and Microarray Data Classification”, IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 7(1), pp. 108-117, 2010. https://doi.org/10.1109/tcbb.2008.46

189. V Vapnik, “The Nature of Statistical Learning Theory”, New York: Springer, 1995. https://doi.org/10.1007/978-1-4757-3264-1

190. A Osareh and B Shadgar, “An Efficient Ensemble Learning Method for Gene Microarray Classification”, Hindawi Publishing Corporation BioMed Research International, vol. 2013, article ID 478410, pp. 1-11, 2013. https://doi.org/10.1155/2013/478410

191. U Alon, N Barkai, D A Notterman, K Gish, S Ybarra, D Mack and A J Levine, “Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by Oligonucleotide arrays”, PNAS, vol. 96(12), pp. 6745–6750, 1999. https://doi.org/10.1073/pnas.96.12.6745

192. D Singh, P G Febbo, K Ross, D G Jackson, J Manola, C Ladd, P Tamayo, A A Renshaw, A V D Amico, J P Richie, E S Lander, M Loda, P W Kantoff, T R Golub and W R Sellers, “Gene expression correlates of clinical prostate cancer behaviour”, Cancer Research, vol. 1, pp. 203–209, 2002. https://doi.org/10.1016/s1535-6108(02)00030-2

193. G J Gordon, R V Jensen, L L Hsiao, S R Gullans, J E Blumenstock, S Ramaswamy, W G Richards, D J Sugarbaker and R Bueno, “Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma”, Cancer Research, vol. 62, pp. 4963–4967, 2002. https://pubmed.ncbi.nlm.nih.gov/12208747/

194. L J V Veer, H Dai, M J V D Vijver, Y D. He, A A M Hart, M Mao, H L Peterse, K V D Kooy, M J Marton, A T Witteveen, G J Schreiber, R M Kerkhoven, C Roberts, P S Linsley, R Bernards and S H Friend, "Gene expression profiling predicts clinical outcome of breast cancer", Nature, vol. 415, pp. 530–536, 2002. https://doi.org/10.1038/415530a

195. M West, C Blanchette, H Dressman, E Huang, S Ishida, R Spang, H Zuzan, J A Olson, J R Marks and J R Nevins, "Predicting the clinical status of human breast cancer by using gene expression profiles", PNAS, vol. 98(20), pp. 11462–11467, 2001. https://doi.org/10.1073/pnas.201162998

196. S A Armstrong, J E Staunton, L B Silverman, R Pieters, M L den Boer, MaD Minden, S E Sallan, E S Lander, T R Golub and S J Korsmeyer, "MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia", Nature Genetics, vol. 30, pp. 41–47, 2001. https://doi.org/10.1038/ng765

197. F Alharbi and A Vakanski, "Machine Learning Methods for Cancer Classification Using Gene Expression Data: A Review", Bioengineering, vol. 10(2), pp. 1-26, 2023. https://doi.org/10.3390%2Fbioengineering10020173

198. T C T M V D P Kraan, C A Wijbrandts, L G M van Baarsen, A E Voskuyl, F Rustenburg, J M Baggen, S M Ibrahim, M Fero, B A C Dijkmans, P P Tak and C L Verweij, "Rheumatoid Arthritis Subtypes Identified by Genomic Profiling of Peripheral Blood Cells: Assignment of a Type I Interferon Signature in a Subpopulation of Patients. Annals of the Rheumatic Diseases", vol. 66(8), pp. 1008-1014, 2007. https://doi.org/10.1136/ard.2006.063412

199. F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot and É Duchesnay, "Scikit-learn: machine learning in python", Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011. https://dl.acm.org/doi/10.5555/1953048.2078195

200. B Komer, J Bergstra and C Eliasmith, "Hyperopt-sklearn: automatic hyperparameter configuration for scikit-learn", Proceedings of the 13th Python in Science Conference (SCIPY 2014), pp. 33–39, 2014. https://doi.org/10.25080/MAJORA-14BD3278-006

201. J D Hunter, "Matplotlib: a 2D graphics environment", Computing in Science & Engineering, vol. 9(3), pp. 90–95, 2007. https://ui.adsabs.harvard.edu/link_gateway/2007CSE.....9...90H/doi:10.1109/MCSE.2007.55

202. C Ambroise and G J McLachlan, "Selection bias in gene extraction on the basis of microarray gene-expression data", PNAS, vol. 99(10), pp. 6562–6566, 2002. https://doi.org/10.1073/pnas.102102699

203. M H Asyali, D Colak, O Demirkaya and M S Inan, "Gene expression profile classification: A review", Current Bioinformatics, vol. 1(1), pp.55–73, 2006. http://dx.doi.org/10.2174/157489306775330615

204. L Breiman and P Spector, "Submodel selection and evaluation regression – the X-random case", International Statistical Review, vol. 60(3), pp. 291–319, 1992. https://doi.org/10.2307/1403680

205. S Liu, S Dissanayake, S Patel, X Dang, T Mlsna, Y Chen and D Wilkins, "Learning accurate and interpretable models based on regularized random forests regression", BMC Systems Biology, vol. 8, S5, pp. 1-9, 2014. https://doi.org/10.1186/1752-0509-8-S3-S5

206. R Ruiz, J C Riquelme, J S A Ruiz, "Incremental wrapper-based gene selection from microarray data for cancer classification", Journal of Pattern Recognition, vol. 39(12), pp. 2383–2392, 2006. https://doi.org/10.1016/j.patcog.2005.11.001

207. S Tabakhi, P Moradi and F Akhlaghian, "An unsupervised feature selection algorithm based on ant colony optimization", Engineering Applications of Artificial Intelligence, vol. 32, pp. 112-123, 2014. https://doi.org/10.1016/j.engappai.2014.03.007

208. S Tabakhi, A Najafi, R Ranjbar and P Moradi, "Gene selection for microarray data classification using a novel ant colony optimization", Neurocomputing, vol. 168, pp. 1024-1036, 2015. https://doi.org/10.1016/j.neucom.2015.05.022

209. C Lai, M J T Reinders and L Wessels, "Random subspace method for multivariate feature selection", Pattern Recognition Letters, vol. 27, pp. 1067-1076, 2006. https://doi.org/10.1016/j.patrec.2005.12.018

210. M Haindl, P Somol, D Ververidis and C Kotropoulos, "Feature Selection Based on Mutual Correlation", Progress in Pattern Recognition, Image Analysis and Applications, Springer, vol. 4225, pp. 569-577, 2006. https://doi.org/10.1007/11892755_59

211. A J Ferreira and M A T Figueiredo, "An unsupervised approach to feature discretization and selection", Pattern Recognition, vol. 45, pp. 3048-3060, 2012. https://doi.org/10.1016/j.patcog.2011.12.008

212. S Theodoridis and K Koutroumbas, "Pattern Recognition, fourth ed.", Elsevier Science, Hardback ISBN: 9781597492720, eBook ISBN: 9780080949123, 2008.

213. B Liao, Y Jiang, W Liang, W Zhu, L Cai and Z Cao, "Gene selection using locality sensitive Laplacian score", IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 11(6), pp. 1146-1156, 2014. https://doi.org/10.1109/tcbb.2014.2328334

214. N Bhui, P K Ram and P Kuila, "Feature Selection from Microarray Data based on Deep Learning Approach", ICCCNT2020, IEEE 49239, pp. 1-5, 2020. https://doi.org/10.1109/ICCCNT49239.2020.9225353

215. P J Gardina, T A Clark, B Shimada, M K Staples, Q Yang, J Veitch, A Schweitzer, T Awad, C Sugnet, S Dee, C Davies, A Williams and Y Turpaz, "Alternative splicing and differential gene expression in colon cancer detected by a whole genome exon array",

BMC Genomics, vol. 7, article 325, pp. 1-18, 2006. https://doi.org/10.1186/1471-2164-7-325

216. K Thorsen, K D Sørensen, A S B Eskildsen, C Modin, M Gaustadnes, A M K Hein, M Kruhøffer, S Laurberg, M Borre, K Wang, S Brunak, A R Krainer, N Tørring, L Dyrskjøt, C L Andersen and T F Orntoft, "Alternative Splicing in Colon, Bladder, and Prostate Cancer Identified by Exon Array Analysis", Molecular & Cellular Proteomics, vol. 7(7), pp. 1214-1224, 2008. https://doi.org/10.1074/mcp.m700590-mcp200

217. I L Botchkina, R A Rowehl, D E Rivadeneira, M S K Jr, H Crawford, A Dufour, J Ju, Y Wang, Y Leyfman and G I Botchkina, "Phenotypic Subpopulations of Metastatic Colon Cancer Stem Cells: Genomic Analysis", Cancer Genomics & Proteomics, vol. 6(1), pp. 19-29, 2009. https://cgp.iiarjournals.org/content/6/1/19

218. R Durai, S Y Yang, A M Seifalian, G Goldspink and M C Winslet, "Role of insulin-like growth factor binding protein-4 in prevention of colon cancer", World Journal of Surgical Oncology, vol. 5, article 128, pp. 1-8, 2007. https://doi.org/10.1186%2F1477-7819-5-128

219. P Singh, B Dai, B Dhruva and S G Widen, "Episomal Expression of Sense and Antisense Insulin-like Growth Factor (IGF) binding Protein-4 Complementary DNA Alters the Mitogenic Response of a Human Colon Cancer Cell Line (HT-29) by Mechanisms That Are Independent of and Dependent upon IGF-11", Cancer Research, vol. 54(24), pp. 6563-6570, 1994. PMID: 7527300 https://pubmed.ncbi.nlm.nih.gov/7527300/

220. H Yu and T Rohan, "Role of the Insulin-Like Growth Factor Family in Cancer Development and Progression", JNCI: Journal of the National Cancer Institute, vol. 92(18), pp. 1472–1489, 2000. https://doi.org/10.1093/jnci/92.18.1472

221. Z Yan, J Li, Y Xiong, W XU and G Zheng, "Identification of candidate colon cancer biomarkers by applying a random forest approach on microarray data", Oncology Reports, vol. 28, pp. 1036-1042, 2012. https://doi.org/10.3892/or.2012.1891

222. K Zhu, Y Wang, L Liu, S Li and W Yu, "Long non-coding RNA MBNL1-AS1 regulates proliferation, migration, and invasion of cancer stem cells in colon cancer by interacting with MYL9 via sponging microRNA-412-3p", Clinics and Research in Hepatology and Gastroenterology, vol. 44(1), pp. 101-114, 2020. https://doi.org/10.1016/j.clinre.2019.05.001

223. H Feng, Y Liu and X Bian, "ALDH1A3 affects colon cancer in vitro proliferation and invasion depending on CXCR4 status", British Journal of Cancer, vol. 118, pp. 224–232, 2018. https://doi.org/10.1038/bjc.2017.363

224. L M V D Waals, I H M B Rinkes and O Kranenburg, "ALDH1A1 expression is associated with poor differentiation, 'right-sidedness' and poor survival in human colorectal cancer", PLOS ONE, vol. 13(10), pp. 1-17, 2018. http://dx.doi.org/10.1371/journal.pone.0205536

225. A R Brown, R C Simmen, V R Raj, T T Van, S L MacLeod and F A Simmen, "Krüppel-like factor 9 (KLF9) prevents colorectal cancer through inhibition of interferon-related signalling", Carcinogenesis, vol. 36(9), pp. 946-55, 2015. https://doi.org/10.1093%2Fcarcin%2Fbgv104

226. M Ying, J Tilghman, Y Wei, H G Cazares, A Q Hinojosa, H Ji and J Laterra, "Kruppel-like factor-9 (KLF9) inhibits glioblastoma stemness through global transcription repression and integrin α6 inhibition", Journal of Biology and Chemistry, vol. 289(47), pp. 32742-56, 2014. doi: 10.1074/jbc.M114.588988.

227. F A Simmen, Y Su, R Xiao, Z Zeng and R C Simmen, "The Krüppel-like factor 9 (KLF9) network in HEC-1-A endometrial carcinoma cells suggests the carcinogenic potential of dys-regulated KLF9 expression", Reproductive Biology and Endocrinology, vol. 6(41), pp. 1-11, 2008. https://doi.org/10.1186%2F1477-7827-6-41

228. X Chen, B Gao, M Ponnusamy, Z Lin and J Liu, "MEF2 signaling and human diseases", Oncotarget, vol. 8, pp. 112152-112165, 2017. https://doi.org/10.18632%2Foncotarget.22899

229. E D Giorgio, W W Hancock and C Brancolini, "MEF2 and the tumorigenic process, hic sunt leones", Biochimica et Biophysica Acta (BBA) - Reviews on Cancer, vol. 1870(2), pp. 261-273, 2018. https://doi.org/10.1016/j.bbcan.2018.05.007

230. L Su, Y Luo, Z Yang, J Yang, C Yao, F Cheng, J Shan, J Chen, F Li, L Liu, C Liu, Y Xu, L Jiang, D Guo, J Prieto, M A Ávila, J Shen and C Qian, "MEF2D Transduces Microenvironment Stimuli to ZEB1 to Promote Epithelial–Mesenchymal Transition and Metastasis in Colorectal Cancer", Molecular and Cellular Pathobiology, vol. 76(17), pp. 5054–5067, 2016. https://doi.org/10.1158/0008-5472.CAN-16-0246.

231. J Y Zhang, F Zhang, C Hong, A E Giuliano, X Cui, G Zhou, G Zhang and Y Cui, "Critical protein GAPDH and its regulatory mechanisms in cancer cells", Cancer Biology and Medicine, vol. 12(1), pp. 10-22, 2015. https://doi.org/10.7497/j.issn.2095-3941.2014.0019

232. Z Tang, S Yuan, Y Hu, H Zhang, W Wu, Z Zeng, J Yang, J Yun, R Xu and P Huang, "Over-expression of GAPDH in human colorectal carcinoma as a preferred target of 3-bromopyruvate propyl ester", Journal of Bioenergetics and Biomembranes, vol. 44(1), pp. 117-125. 2012. https://doi.org/10.1007%2Fs10863-012-9420-9

233. C W Su, C W Lin, W E Yang and S F Yang, " TIMP-3 as a therapeutic target for cancer", Therapeutic Advances in Medical Oncology, vol. 11, pp. 1-17, 2019. https://doi.org/10.1177/1758835919864247.

234. Y X Bai, J L Yi, J F Li and H Sui, “Clinicopathologic significance of BAG1 and TIMP3 expression in colon carcinoma”, World Journal of Gastroenterology, vol. 13(28), pp.3883-3885, 2007. https://doi.org/10.3748/wjg.v13.i28.3883

235. A M Kamal, N M Hamdy, H M Hegab and H O El-Mesallamy, “Expression of thioredoxin-1 (TXN) and its relation with oxidative DNA damage and treatment outcome in adult AML and ALL: A comparative study”, Hematology, vol. 21(10), pp. 567-575, 2016. https://doi.org/10.1080/10245332.2016.1173341

236. T C Karlenius and K F Tonissen, “Thioredoxin and Cancer: A Role for Thioredoxin in all States of Tumor Oxygenation”, Cancers (Basel), vol. 2(2), pp. 209-232, 2010. https://doi.org/10.3390/cancers2020209

237. T Léveillard and N Aït-Ali, “Cell Signaling with Extracellular Thioredoxin and Thioredoxin-Like Proteins: Insight into Their Mechanisms of Action”, Oxidative Medicine and Cellular Longevity, vol. 2017, pp. 1-12, 2017. https://doi.org/10.1155/2017/8475125

238. Y Zhang, F Wang, X Chen, Y Zhang, M Wang, H Liu, P Cao, X Ma, T Wang, J Zhang, X Zhang, P Lu and H Liu, “CSF3R Mutations are frequently associated with abnormalities of RUNX1, CBFB, CEBPA, and NPM1 genes in acute myeloid leukemia”, Cancer, vol. 124(16), pp. 3329-3338, 2018. https://doi.org/10.1002/cncr.31586

239. M Ritter, M Klimiankou, O Klimenkova, A Schambach, D Hoffmann, A Schmidt, L Kanz, D C Link, K Welte and J Skokowa, “Cooperating, congenital neutropenia-associated Csf3r and Runx1 mutations activate pro-inflammatory signaling and inhibit myeloid differentiation of mouse HSPCs”, Annals of Hematology, vol. 99(10), pp. 2329-2338, 2020. https://doi.org/10.1007%2Fs00277-020-04194-0

240. M Klimiankou, M Uenalan, S Kandabarau, R Nustede, I Steiert, S M Heineke, C Zeidler, J Skokowa and K Welte, “Ultra-Sensitive CSF3R Deep Sequencing in Patients With Severe Congenital Neutropenia”, Frontiers in Immunology, vol. 10, pp. 1-11, 2019. https://doi.org/10.3389%2Ffimmu.2019.00116.

241. A Lance, L Druhan, G Vestal, N M Steuerwald, A Hamilton, M Smith, A E Price, E Tjaden, A N Fox and B R Avalos, “Altered expression of CSF3R splice variants impacts signal response and is associated with SRSF2 mutations”, Nature, Leukemia, vol. 34(Aug), pp.1-11, 2020. https://www.nature.com/articles/s41375-019-0567-9

242. N Szuber and A Tefferi, “Chronic neutrophilic leukemia: new science and new diagnostic criteria”, Blood Cancer Journal, vol. 8, pp. 1-19, 2018. https://doi.org/10.1038/s41408-018-0049-8

243. Y Kim, SYoon, S J Kim, J S Kim, J W Cheong and Y H Min, “Myeloperoxidase expression in acute myeloid leukemia helps identifying patients to benefit from transplant”, Yonsei Medical Journal, vol. 53(3), pp. 530-536, 2012. https://doi.org/10.3349/ymj.2012.53.3.530

244. F A L Rangel, V C Valencia, M A G Guijosa and C C Penagos, “Acute Myeloid Leukemia-Genetic Alterations and Their Clinical Prognosis”, International Journal of Hematology-Oncology and Stem Cell Research, vol. 11(4), pp. 328-339, 2017. https://pubmed.ncbi.nlm.nih.gov/29340131/

245. L. Handschuh, “Not Only Mutations Matter: Molecular Picture of Acute Myeloid Leukemia Emerging from Transcriptome Studies”, Journal of Oncology, vol. 2019, Article ID 7239206, pp. 1-37, 2019. https://doi.org/10.1155%2F2019%2F7239206.

246. H Wang, H Hu, Q Zhang, Y Yang, Y Li, Y Hu, X Ruan, Y Yang, Z Zhang, C Shu, J Yan, E K Wakeland, Q Li, S Hu and X Fang, “Dynamic transcriptomes of human myeloid leukemia cells”, Genomics, vol. 102(4), pp.250–256, 2013. https://doi.org/10.1016/j.ygeno.2013.06.004

247. D L Tong and G Ball, “Exploration of Leukemia Gene Regulatory Networks Using A Systems Biology Approach”, IEEE International Conference on Bioinformatics and Biomedicine, pp. 68-73, 2014. http://dx.doi.org/10.1109/BIBM.2014.6999250

248. A H Chen, Y Tsau and C Lin, “Novel methods to identify biologically relevant genes for leukemia and prostate cancer from gene expression profiles”, BMC Genomics, vol. 11(274), pp. 1-21, 2010. https://doi.org/10.1186/1471-2164-11-274

249. Y Qi and X Wang, “Interval-valued analysis for discriminative gene selection and tissue sample classification using microarray data”, Genomics, vol. 101(1), pp. 38-48, 2013. https://doi.org/10.1016/j.ygeno.2012.09.004

250. P Maji and S Paul, “Scalable Pattern Recognition Algorithms: Applications in Computational Biology and Bioinformatics”, Springer, Warsaw, Poland, ISBN 978-3-319-05630-2, 2013. https://doi.org/10.1007/978-3-319-05630-2

Shilpi Bose.