Unsupervised Pattern Recognition in a Fuzzy Set Theoretic Framework

Thesis submitted by

Kaushik Sarkar

Doctor of Philosophy (Engineering)

Department of Instrumentation and Electronics Engineering
Faculty Council of Engineering and Technology
Jadavpur University
Kolkata, India
2024

To my family ...

INDEX NO. 117/17/E

1. Title of the Thesis: Unsupervised Pattern Recognition in a Fuzzy Set Theoretic Framework

2. Name, Designation & Institution of the Supervisors:

i. **Prof. Rajanikanta Mudi**
Professor
Department of Instrumentation and Electronics Engineering
Jadavpur University
Kolkata-700098

ii. **Prof. Nikhil Ranjan Pal**
Professor
Electronics and Communication Sciences unit
Indian Statistical Institute
Calcutta-700108

List of Publications

i. Journal Publications

1. **Kaushik Sarkar**, Rajani K Mudi and Nikhil R Pal, "Weighted Fuzzy C-Means: Unsupervised Feature Selection to Realize a Target Partition", *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 32, No. 8, pp. 1111-1134, 2024.
2. Nikhil R Pal and **Kaushik Sarkar**, "What and When Can We Gain From the Kernel Versions of C-Means Algorithm?," *IEEE Transactions on Fuzzy Systems*, Vol 22, Issue 2, pp. 363-379, 2014.
3. **Kaushik Sarkar** and Rajani K Mudi , "Fuzzy Clustering Exploiting Neighbourhood Information for Non-image Data", *International Journal of Computer Sciences and Engineering*, vol. 12, issue 2, pp. 1-8, 2024.
4. **Kaushik Sarkar**, Prantik Chatterjee and Nikhil R Pal, "Finding Synergy Networks From Gene Expression Data: A Fuzzy-Rule-Based Approach", *IEEE Transactions on Fuzzy Systems*, Vol 24, Issue 6, pp. 1488-1499, 2016.

ii. Conference Publications

1. **Kaushik Sarkar** and Nikhil Ranjan Pal, "Is It Rational To Partition A Data Set Using Kernel-Clustering?," *IEEE International Conference on Fuzzy Systems*, FUZZ-IEEE 2011, Taipei, Taiwan.

Statement of Originality

I Kaushik Sarkar registered on March 3, 2017 do hereby declare that this thesis entitled "**Unsupervised Pattern Recognition in a Fuzzy Set Theoretic Framework**" 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 5%.

[illegible]
....
Signature of Candidate:
Date: 29/2/24

Certified by Supervisors:
(Signature with date, seal)

[illegible] 29/02/24
....
(Prof. Rajanikanta Mudi)

Professor
Dept. of Instrumentation & Electronics Engg.
Jadavpur University
Saltlake, 2nd Campus
Kolkata-700098

[illegible] 29-02-24
....
(Prof. Nikhil Ranjan Pal)

Professor Nikhil R. Pal
Electronics and Communication Sciences Unit
Indian Statistical Institute
203, B.T. Road, Calcutta-700108
India

CERTIFICATE FROM THE SUPERVISORS

This is to certify that the thesis entitled "**Unsupervised Pattern Recognition in a Fuzzy Set Theoretic Framework**" submitted by Kaushik Sarkar, who got his name registered on March 3, 2017 for the award of Ph.D. (Engg.) degree of Jadavpur University is absolutely based upon his own work under the supervision of Prof. Rajanikanta Mudi and Prof. Nikhil Ranjan Pal 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.

[illegible] 29/02/24
....
(Prof. Rajanikanta Mudi)

Professor
Dept. of Instrumentation & Electronics Engg.
Jadavpur University
Saltlake, 2nd Campus
Kolkata-700098

NRanjaPal 29-02-24
....
(Prof. Nikhil Ranjan Pal)

Professor Nikhil R. Pal
Electronics and Communication Sciences Unit
Indian Statistical Institute
203, B.T. Road, Calcutta-700108
India

Acknowledgements

I am immensely indebted to my supervisors, Prof. Rajanikanta Mudi and Prof. Nikhil Ranjan Pal for their kind and invaluable supervision. I am thankful to all the faculty members and staffs of the Department of Instrumentation and Electronics Engineering, Jadavpur University, Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata and Department of Electronics and Communication Engineering, Narula Institute of Technology, Agarpara, Kolkata. I am grateful to all my friends. Special thanks go to Kaustav, Prantik, Suvrajyoti, Suchismita and Rupan. Last but not the least, I want to acknowledge my family members for everything.

[illegible]
29/2/24
Kaushik Sarkar

Abstract

Pattern recognition primarily involves three major tasks: clustering, classification and feature selection/dimensionality reduction. The success of clustering and classification heavily depends on representation of data, i.e., the features used. It is important to note that more features does not necessarily imply better clustering and classification performance because there may be derogatory features which hinder the performance of clustering and classification. To address these three problems, a computational framework can be established using statistical approaches, neural networks, and/or fuzzy set theoretic methods. Development of systems for tasks like classification or prediction always demands some data with the target values (such as class labels). Thus, learning of such systems is called supervised learning. Clustering finds "homogeneous" subgoups in the data without knowing any class labels. Consequently, clustering is an unsupervised learning. On the other hand, feature selection/dimensionality reduction can be done under both frameworks, supervised and unsupervised. This thesis primarily solves unsupervised pattern recognition problems using fuzzy set theoretic framework.

In Chapter 1, the literature related to the problems addressed in this thesis is briefly reviewed. As already mentioned, the success of a pattern recognition system heavily depends on the features used. The first contributory chapter, Chapter 2, introduces an unsupervised feature selection method based on regularized weighted fuzzy C-means (WRFCM) clustering. When the target task is clustering, a good objective should be to select a subset of features that can generate the same/similar partition matrix to the partition matrix obtained on the original high dimensional data by a clustering algorithm. To achieve this, a novel objective function is proposed, considering the Fuzzy C-means (FCM) clustering algorithm. This approach realizes feature selection within the WRFCM framework,

emphasizing features to maintain the FCM-based target partition. The proposed method is evaluated using Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and KM-index. NMI, and ARI measure the agreement between clusters, i.e, the partition in the lower dimension and the partition of the original data. On the other hand, KM-index measures the disagreement between the two partitions. Experimental results on synthetic and real datasets demonstrate the efficacy of the proposed method in selecting informative features. This approach fills a crucial gap in unsupervised feature selection, making it valuable for real-world applications. To the best of our knowledge unsupervised feature selection to match a target partition has been done for the first time.

The core of this discourse is explored in the second contributory work, presented in Chapter 3, which emphasizes a pivotal concern: the disparity between clustering in the original feature space and the pursuit of clustering in the kernel space using fuzzy set theoretic framework; in particular, using KFCM-F (Kernel FCM with prototypes in the feature Space) and KFCM-K (Kernel FCM with prototypes in the kernel Space). In this work, the rationality behind such clustering approaches is critically examined. Using simple datasets, it is argued and demonstrated that it is not a good idea to find clusters in the kernel space when the objective is to look for clusters in the original data because in the kernel space the data may have a different geometry from that in the original feature space. In particular, the following points are demonstrated: (1) improper choice of the number of clusters may lead to very counterintuitive clusters (e.g., instead of merging nearby clusters, it may merge clusters that are far from each other) and (2) improper choice of kernel parameters has a significant effect on the extracted clusters and it can even impose arbitrary cluster structures that are undoubtedly absent in the original data. However, it is important to note that it is not implied that kernel clustering can never produce desirable results. In fact, kernel clustering could be useful provided we can choose right kernel parameters. But the process being unsupervised, As of now, no solution to this issue has been proposed. Yet, amidst these challenges, the work cautiously acknowledges the potential utility of kernel clustering, provided precise parameter choices—a Herculean task within unsupervised realms—can be achieved.

In the third contributory chapter, Chapter 4, this debate is further amplified, presenting a meticulous argument against the clustering of object data in projected spaces. It rigorously scrutinizes the fundamental objective of clustering algorithms: to unearth natural subgroups defined by inherent similarities within the data. The work contends that any transformation of data into a new space should, unequivocally, aid in the identification of the same clusters present in the original dataset to hold any utility. It firmly cautions against transformations that significantly alter the "structure"/"geometry" of the data, warning that such alterations can result in the emergence of clusters devoid of relevance to the original context. This exploration moves beyond the algorithmic realm, delving into the philosophical underpinnings, questioning the very essence of clustering in transformed spaces. While acknowledging the potential utility of kernel clustering in 2D/3D data, where visualization offers insights, it contends that in higher dimensions, its relevance remains enigmatic, intensifying the quest to determine its efficacy. Here are our line of arguments: (i) The objective of any clustering algorithm is to find natural subgroups in X, where the subgroups are defined by a measure of similarity between the vectors in X. (ii) If the data is transformed, X into Y in another space by a nonlinear transformation and try to find clusters in Y, then such clusters can be useful, if and only if Y helps to identify the same clusters that are present in X, because that is the objective. (iii) If Y maintains the same structure/topology as that of X, then the use of Y may not give any advantage. (iv) On the other hand, if Y changes the structure (i.e., imposes a new structure) on the data and that change makes the extraction of the desired clusters present in X, easier, then clustering of Y is useful. (v) But when Y imposes new (non-existent) structures, the clustering algorithm may find very strange clusters with no relation to the actual clusters present in X. (vi) Thus, when attempting to cluster in a transformed space, the issue is to know if it could help us to find the clusters present in X. To get any benefit from kernel clustering (or clustering in any other transformed space) this question must be answered first; otherwise, there is a risk of finding completely irrelevant clusters without realizing it and thereby making kernel clustering useless. (vii) This issue is a philosophical one and is neither dependent on the choice of clustering algorithm nor on the particular transformation (kernel function) used. (viii) Except for 2D/3D data, there is no way to answer the question in (vi) and for 2D/3D data since it is possible to examine the data directly, without the need for kernel clustering. So there is no benefit from kernel clustering. The claims are demonstrated and justified through detailed analysis using both synthetic and real data sets with visual assessment as well as with Normalized Mutual Information (NMI), Adjusted Rand Index (ARI) and cluster instability. It is proposed to use Sammon's nonlinear projection method to obtain a crude visual representation of the data in the kernel space. The issue of how to choose appropriate parameters for the kernel function is discussed, though no solution to this problem is provided. Finally, the interaction between the kernel parameters and the algorithmic parameters is examined.

The fourth contributory chapter, Chapter 5 presents an extension of the conventional Fuzzy C-Means (FCM) algorithm, unlocking the potential of neighbourhood information in non-image datasets that reside in the Euclidean space. This approach strategically incorporates local/ contextual information, demonstrating enhanced clustering performance compared to conventional FCM. The utility and validity of the proposed method are demonstrated using both synthetic and real datasets.

In Chapter 6, the fifth contributory chapter also deals with feature selection as in Chapter 2. Here, a special type of features, called synergistic pairs of features, is selected, and the focus is placed on a specific but important problem related to prostate cancer. Typically, fuzzy rules are extracted from data by clustering the data point into a fixed number of clusters and then translating each cluster into a rule representing a specific class. The present problem involves 12558 genes and thus there is a need to cluster data corresponding to $\binom{12558}{2}$, i.e, about 80 million pairs of genes to extract rules. A novel scheme is proposed to address this issue. Here, the objective is to identify a special type of network called "synergy network" using synergistic pairs of genes/features. The goal is to identify synergistic gene pairs that interact through collaboration with respect to a disease and form a network of such synergistic genes. A fuzzy-rule-based approach is proposed for the discovery of synergy networks. It is justified that a fuzzy rule base is a natural choice to realize all the desired attributes of synergistic relations. This is the first attempt to exploit fuzzy modelling for finding synergy networks. The system uses a set of human understandable rules that is generated at a low cost for every pair of genes. The proposed method is applied to prostate cancer datasets, demonstrating its capability to discover gene pairs that collaborate with each other with respect to prostate cancer. The results are shown to be statistically significant, and the relevance of the identified genes to cancer biology is discussed.

Finally, the thesis concludes in Chapter 7, where the limitations of the proposed methods are discussed, along with possible directions for future work.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction and Scope of the Thesis

1.1 Introduction

Pattern recognition is a scientific field focused on categorizing objects into distinct classes/clusters. While pattern recognition has a historical foundation in statistical theory, the rise of computers has propelled its practical applications. In the post-industrial era, automation in production and the demand for efficient information handling have elevated pattern recognition's significance in engineering applications and research. It now plays a vital role in decision-making within machine intelligence systems. It is inherent for us to aspire to create machines proficient in pattern recognition. Applications like automated speech recognition, fingerprint and optical character identification, and DNA sequence analysis underscore the immense utility of accurate machine pattern recognition. Delving into the complexities of building such systems deepens the understanding of natural pattern recognition, particularly in humans. In areas like speech and visual recognition, our design endeavors may draw inspiration from nature's solutions, impacting both algorithmic choices and the crafting of specialized hardware.

1.2 Categorization of pattern recognition

Let's consider a simplified simulation resembling a medical image classification task . In Figure 1.1 [*], two distinct images display regions, each visually unique. The region in Figure 1.1a is attributed to a benign lesion (class A), while the region in Figure 1.1b is identified as a malignant lesion (cancer, class B). Assuming access to an image database with various patterns from both classes, the goal is to identify measurable quantities distinguishing these regions. Let us consider mean (μ) and standard deviation (σ) are the two measurable quantities here.

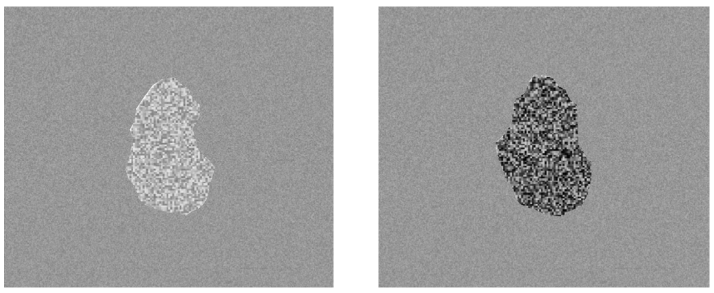

(a) Class A (b) Class B

Figure 1.1: An illustrative example of two different cell.

Figure 1.2 depicts a plot of the mean intensity versus the corresponding standard deviation for each region of interest. Class A and class B patterns exhibit different spreads, suggesting a straight line as a potential separator.

This artificial classification task illustrates the foundation of many pattern recognition problems. The measurements used, such as mean value and standard deviation in this case, are referred to as features.

In the scenario illustrated in Figure 1.2, the assumption was centered around the existence of a designated set of training data, with the classifier being formulated by leveraging this pre-existing knowledge. This particular approach falls under the umbrella of supervised pattern recognition, and in the broader context of machine learning, it is recognized as

[*]Source: "Pattern Recognition" by S. Theodoridis and K. Koutroumbas.

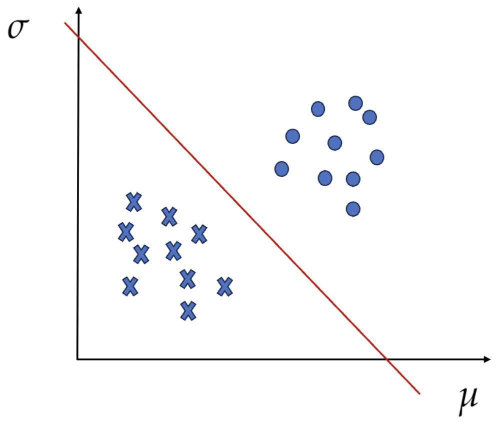


Figure 1.2: Graph depicting mean value against standard deviation for various images from class A (circle) and class B (cross).

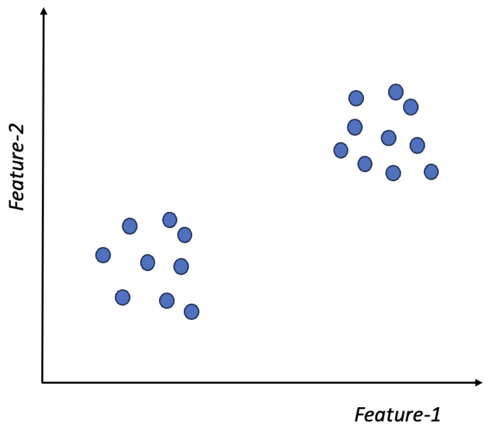


Figure 1.3: Clustering of data represented by $Feature - 1$ and $Feature - 2$.

supervised learning. However, it is crucial to acknowledge that not all pattern recognition tasks align with this paradigm. There exists another category of problems where training data, complete with known class labels, is not at our disposal.

Consider the Figure 1.3, where a collection of feature vectors denoted as $Feature - 1$ and $Feature - 2$, and the primary objective becomes the discernment of underlying similarities, facilitating the grouping or clustering of vectors that share common traits. This variant is referred to as unsupervised pattern recognition, synonymous with unsupervised learning or clustering. The essence of this approach lies in the exploration of data patterns and structures without explicit guidance from labeled examples, allowing for the discovery of inherent relationships among the feature vectors. Obviously, one can easily find two clusters in the Figure 1.3.

So, it is clear that the field of pattern recognition consists of three major categories, i.e, Feature selection/extraction, Supervised learning and Unsupervised learning. In the subsequent sections, it is discussed further.

1.2.1 Categorization of dimensionality reduction

Reducing the number of features to a necessary minimum is motivated by various factors. The most apparent one is computational complexity. Another associated reason is that, while two features may individually provide valuable classification information, combining them into a feature vector may not yield significant benefits due to high mutual correlation. Consequently, complexity increases without substantial gains. A significant additional reason arises from the necessary generalization properties imposed by the classifier.

Based on the reduction approach, dimensionality reduction schemes can be divided into two broad categories: feature extraction and feature selection. Dimensionality reduction is now formally defined. Let us denote the input space by $X \subseteq \mathbb{R}^p$, where p is the original dimension of the data. The feature set corresponding to the input space is denoted as $\{x_1, x_2, \ldots, x_p\}$. Let us denote the output space, i.e., the space where the data is projected in lower dimension, by $Y \subseteq \mathbb{R}^q$, where q is the output dimension or reduced dimension of the data, and $q < p$. The feature set corresponding to the output space is $\{y_1, y_2, \ldots, y_q\}$. Dimensionality reduction is a mapping $\phi : X \to Y$. For a point $x \in X$, its projection in lower dimension, $y \in Y$, is given by $y = \phi(x)$.

1.2.1.1 Feature extraction

In the realm of data preprocessing and analysis, managing the ever expanding dimensions of datasets has become a critical concern. Feature manipulation, encompassing both selection and extraction, emerges as a pivotal strategy to enhance the efficiency and efficacy of subsequent analytical tasks. Feature selection involves identifying and retaining a subset of the most relevant features from the original dataset, aiming to eliminate noise, reduce computational complexity, and improve model interpretability.

However, the challenges associated with increasingly intricate datasets extend beyond the mere reduction of feature dimensions. Even after selecting pertinent features, issues such as inter-feature correlation and information redundancy may persist, hindering the overall performance of analytical models. This is where feature extraction steps in, transcending the limits of selection by transforming the data into a more condensed and meaningful representation. By capturing essential patterns and characteristics, feature extraction not only addresses issues of dimensionality but also optimizes the data for subsequent modeling, ensuring a refined input for machine learning algorithms. Examples of linear methods encompass principal component analysis (PCA) [1], canonical correlation analysis [2], linear discriminant analysis (LDA) [3], factor analysis [4], and locality preserving projections [5]. It's important to note that these linear techniques may not be optimal for capturing information in datasets with non-linear structures. Several non-linear feature extraction methods include Sammon's projection [6], ISOMAP [7], Locally Linear Embedding (LLE) [8], autoencoder-based methods [9], t-distributed stochastic neighbor embedding (t-SNE) [10].

1.2.1.2 Feature selection

Feature selection is a process that involves the identification and choice of a subset of original features to accurately represent the data for a specific task. The chosen subset, denoted as $\{y_1, y_2, \cdots, y_q\} \subseteq \{x_1, x_2, \cdots, x_p\}$, constitutes a condensed representation of the initial feature set. Notably, in the context of feature selection, the mapping ϕ is linear. The representation of the selected features can be expressed as follows.

$$\begin{aligned}
y_1 &= \theta_{1,1}x_1 + \theta_{1,2}x_2 + \cdots + \theta_{1,p}x_p \qquad (1.1)\\
y_2 &= \theta_{2,1}x_1 + \theta_{2,2}x_2 + \cdots + \theta_{2,p}x_p\\
&\vdots\\
y_q &= \theta_{q,1}x_1 + \theta_{q,2}x_2 + \cdots + \theta_{q,p}x_p
\end{aligned}$$

given, $\sum_k \theta_{k,j} = 1, \sum_j \theta_{k,j} = 1, \theta_{k,j} \in \{0,1\},\ \forall k,j.$

In any feature selection method the goal is to learn $\{y_1, y_2, \cdots, y_q\}$. Mathematically, learning means to estimate the coefficients $\theta_{k,j}$s. By default, feature selection methods involve explicit mapping techniques. When a dataset undergoes projection through feature selection, the reduced space maintains the identity of the original features. If the original features carry inherent meaning, the selected features retain that meaningful essence. This contributes to the interpretability of the underlying system utilizing the chosen features. Feature selection methods necessitate a criterion to gauge the relevance of each feature. Once a feature selection criterion is established, a systematic procedure is followed to identify a subset of the most pertinent features. These procedures can be broadly categorized into three groups: filter, wrapper, and embedded methods. Filter methods rank the features and choose those that rank high. Notably, these methods do not incorporate any feedback from the predictor system where the output is employed. While filter methods, which involve selecting a subset without guidance from the underlying predictor system, may not always yield optimal subsets concerning the predictor system [11, 12], examples of filtering criteria for filter-based feature selection methods include the Fisher score or F-score [13], mutual information [14, 15], and normalized mutual information [15–18]. On the other hand, wrapper methods determine the feature selection criterion based on the performance of the predictor system. The subset of features that achieves the highest score from the predictor is chosen. Discovering the optimal feature subset in wrapper methods entails presenting all possible feature subsets to the predictor system. For a set of p input features, determining the optimal subset requires evaluating $(2p - 1)$ subsets. However, when p is moderately high, the wrapper method becomes impractical [17,19]. To address this issue, various search strategies, such as best-first search, branch and bound, sequential search, hill climbing, simulated annealing, and genetic algorithms, have been employed [11, 20]. In contrast, embedded methods conduct feature selection concurrently with the primary learning task, learning the predictor system while identifying the best subset for it. Generally, embedded methods demonstrate higher efficiency compared to filter and wrapper methods [11,12,20]. Notable approaches within the embedded category include pruning-based, decision tree-based, and regularization-based methods. Recursive Feature Elimination with Support Vector Machines (SVM) is an example of a pruning-based method, while CART, C4.5, and ID3 exemplify decision tree-based methods. Additionally, Least Absolute Shrinkage and Selection Operator (LASSO) and Elastic Net are examples of regularization-based embedded methods [11, 21]. More details on different features are discussed in Chapter 2. A novel feature selection technique for unsupervised learning has been proposed in Chapter 2 where the features are selected based on a target partition.

1.2.2 Categorization of learning method

As outlined in the initial sections of this chapter, the act of reducing the dimensionality of data can be driven by different objectives, depending on the presence or absence of a specified target problem. When no particular target problem is designated, dimensionality reduction is typically performed to eliminate redundant or noisy information, primarily aiming to alleviate computational burdens and storage requirements. However, in numerous scenarios, dimensionality reduction serves as a preprocessing step preceding a pattern recognition task.

In such cases, the primary goal of dimensionality reduction is to enhance the performance of the underlying pattern recognition task. This task may take the form of either supervised or unsupervised pattern recognition. This thesis primarily addresses unsupervised pattern recognition. However, supervised learning using a fuzzy rule-based system is also considered to highlight certain limitations of unsupervised approaches. The following subsections discuss these two methods in detail.

1.2.2.1 Supervised learning

Classification stands out as a crucial and frequently encountered challenge in the realms of data mining and machine learning. Numerous real-world issues across diverse domains can be reformulated as classification problems. These encompass tasks such as diagnosing conditions from microarray data, text categorization, medical diagnoses, software quality assurance, and more. Formally, classification can be defined as follows: consider a given training dataset $D = \{(x_i, y_i) : i = 1, \ldots, n\}$, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T \in F_S \subseteq \mathbb{R}^p$ represents the ith training pattern, and $y_i \in Y = \{Y_1, Y_2, \ldots, Y_c\}$ is the class label for x_i. In this context, the vector $f_j = (x_{1j}, x_{2j}, \ldots, x_{nj})^T \in \mathbb{R}^n$ is termed the jth feature vector, and F_S is referred to as the feature space. Here, d, c and n respectively denote the dimension of the feature space, the number of classes, and number of samples in the training data. The classification task involves formulating the learning of a mapping or model, denoted as a classifier $C : F_S \to Y$. The goal is to establish, for an unknown pattern $x \in F_S$, $C(x) = y_p \in Y$ as the desired class label associated with x. Furthermore, challenges emerge from the presence of redundant and/or noisy features. In certain instances, the adoption of an ensemble of classifiers proves to be a viable strategy to improve classification accuracy. Speech recognition [22], handwriting recognition [23], biometric identification [24], document classification [25], image classification [26], and sentiment analysis [27] exemplify practical applications of classification in various domains.

1.2.2.2 Unsupervised learning

Unsupervised learning is a pivotal paradigm in machine learning where the algorithm is tasked with extracting patterns, relationships, or structures from unlabeled data without explicit guidance from labeled outcomes. Unlike supervised learning, where the algorithm is provided with labeled examples to learn from, unsupervised learning involves exploring the inherent structures and patterns within the data itself.

One of the key advantages of unsupervised learning lies in its ability to uncover insights and patterns from unannotated data, making it particularly valuable in scenarios where labeled data is scarce or expensive to obtain. Common unsupervised learning algorithms include k-means clustering [28], hierarchical clustering [29], and Fuzzy C-means clustering [30,31].

The next section discusses two popular techniques associated with pattern recognition using Fuzzy sets.

1.3 Pattern recognition tools using Fuzzy

Fuzzy set theory, introduced by Lotfi A. Zadeh in the mid-1960s, revolutionized the way we handle uncertainty and imprecision in mathematical modeling and decision-making. Traditional set theory, developed by mathematicians like Georg Cantor, primarily deals with clear, well-defined distinctions: an element either belongs to a set or does not. However, in many real-world scenarios, distinctions are not always so black and white; uncertainty and vagueness often characterize the boundaries between different categories.

Fuzzy set theory provides a mathematical framework to represent and manage this uncertainty by allowing for degrees of membership in a set. Unlike classical sets where membership is binary (either 0 or 1), fuzzy sets enable the assignment of membership degrees between 0 and 1, capturing the gradations of belongingness. This flexibility makes fuzzy set theory a valuable tool for modeling and reasoning in situations where information is imprecise, incomplete, or subjective.

Key Concepts

(a) **Membership Function:** The cornerstone of fuzzy set theory is the membership function, denoted as μ. It quantifies the degree to which an element belongs to a particular fuzzy set. For a given element x and a fuzzy set A, the membership degree $\mu_A(x)$ represents the degree of membership of x in A.

(b) **Degrees of Membership:** Unlike classical set theory, which recognizes membership as a binary concept (either in or out), fuzzy sets allow for partial membership. An element can belong to a fuzzy set to a certain degree between 0 (not a member) and 1 (full membership).

(c) **Fuzzy Set Operations:** Fuzzy set operations, such as union, intersection, and complement, extend classical set operations to handle membership degrees. The union of two fuzzy sets A and B, denoted as $A \cup B$, is determined by taking the maximum membership degree for each element from the two sets.

(d) **Fuzzification and Defuzzification:** Fuzzification involves converting crisp, well-defined inputs into fuzzy sets to capture the uncertainty in the input data. Defuzzification is the process of converting fuzzy output into a crisp result, aiding decision-making.

Applications

Fuzzy set theory finds applications across various fields:

(a) **Control Systems:** Fuzzy logic controllers are employed in systems where precise control is challenging due to uncertainty or variability.

(b) **Pattern Recognition:** In pattern recognition, fuzzy sets help model and interpret ambiguous or imprecise patterns, enhancing the accuracy of recognition systems.

(c) **Artificial Intelligence:** Fuzzy logic is integrated into AI systems to handle uncertain information, enabling more human-like reasoning.

(d) **Decision Support Systems:** Fuzzy sets facilitate decision-making in situations where information is subjective or incomplete, providing a robust framework for decision support systems.

(e) **Risk Assessment:** Fuzzy set theory is applied in risk assessment models to account for uncertainty and vagueness in risk factors.

Fuzzy set theory has become a cornerstone in dealing with uncertainty, imprecision, and vagueness in diverse fields. Its ability to represent and manipulate partial truth has led to its widespread adoption in various applications, allowing for more realistic and nuanced modeling in the face of real-world complexity. The flexibility of fuzzy sets in capturing and managing uncertainty continues to make them an invaluable tool in contemporary mathematical modeling and decision science.

1.3.1 Fuzzy C-Means (FCM)

Fuzzy C-means (FCM) clustering is a versatile and powerful unsupervised learning algorithm that extends the classical K-Means algorithm to accommodate a more nuanced representation of cluster assignments. Unlike K-Means, where each data point is definitively assigned to a single cluster, FCM introduces a fuzzy membership concept, allowing a data point to belong to multiple clusters simultaneously with varying degrees of membership. [31]

The fundamental idea behind FCM is to capture the inherent fuzziness or ambiguity present in real-world data. By incorporating membership values, FCM reflects the likelihood or degree to which a data point belongs to each cluster. This provides a richer and more realistic representation of complex relationships within the data, making FCM particularly suitable for scenarios where boundaries between clusters are not well-defined.

A clustering algorithm partitions a set of n input vectors $X = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$, $\mathbf{x}_i \subset \mathbb{R}^p$ into c clusters so that members of the same cluster are "similar" to each other and members of different clusters are dissimilar. The Fuzzy objective function is defined in Equation 1.2.

$$J_m = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m ||\mathbf{x}_k - \mathbf{v}_i||^2 \tag{1.2}$$

where n is the number of data points, c is the number of clusters, $\mathbf{x}_k$ is the kth data point, $\mathbf{v}_i$ is the centroid of the ith cluster, u_{ik} is the membership degree of $\mathbf{x}_k$ in the ith cluster, and m is a fuzzifier parameter where $m > 1$.

The membership matrix is defined in Equation 1.3.

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\mathbf{x}_k - \mathbf{v}_i\|}{\|\mathbf{x}_i - \mathbf{v}_j\|} \right)^{\frac{2}{m-1}}} \tag{1.3}$$

The centroids are updated using the Equation 1.4

$$\mathbf{v}_i = \frac{\sum_{k=1}^{n} u_{ik}^m \cdot \mathbf{x}_i}{\sum_{i=1}^{n} u_{ik}^m} \tag{1.4}$$

The corresponding algorithm of FCM is shown in Algorithm 1.1

Chapters 3 and 4 examine the behavior of the kernel version of the C-means algorithm. An enhanced version of the conventional Fuzzy C-means (FCM) algorithm is proposed in Chapter 5, designed to leverage neighborhood information in non-image datasets characterized by Euclidean metrics.

1.3.1.1 Kernel function

Kernel functions are fundamental in machine learning and statistical modeling, serving as a mathematical tool to transform input data into a higher-

Algorithm 1.1 Fuzzy C-means

```
(a) Fix $c, maxItr, m$ and $\epsilon > 0$, a tolerance level for termination.
(b) Initialize the set of prototypes, $V^0 = [V_i^0]_{c \times p}$
(c) Initialize the membership matrix $U^0 = [U_{ik}^0]_{c \times n}$
(d) For $t = 1, 2, \cdots, maxItr$ do
        Update $U^{t+1} = [u_{ik}^{t+1}]_{c \times n}$ according to Equation (1.3)
        Update $\mathbf{v}^{t+1} = [v_{ik}^{t+1}]_{c \times p}$ according to Equation (1.4)
    until
        $\sum_{i=1}^{c} \sum_{k=1}^{n} |u_{ik}^{(t+1)} - u_{ik}^{(t)}| < \epsilon$    (1.5)
    is satisfied.
(e) Return $U^{t+1}, \mathbf{v}^{t+1}$.
```

dimensional space. This concept is particularly crucial in the context of support vector machines (SVM), where kernel functions play a key role in implicitly mapping data points into a higher-dimensional space without explicitly computing the transformed feature vectors.

At its essence, a kernel function measures the similarity between pairs of data points in the input space. The choice of a kernel function determines the shape and complexity of the decision boundary in the transformed space, enabling the handling of non-linear relationships within the data while maintaining computational efficiency.

Commonly used kernel functions include:

(a) **Linear Kernel:**

$$K(x, y) = x^T y \tag{1.6}$$

The linear kernel represents a simple inner product of the input vectors, preserving the original feature space.

(b) **Polynomial Kernel:**

$$K(x, y) = (x^T y + c)^d \tag{1.7}$$

The polynomial kernel introduces non-linearity by raising the inner product to a power d, where c is a user-defined constant.

(c) **Radial Basis Function (RBF) or Gaussian Kernel:**

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \tag{1.8}$$

The RBF kernel measures similarity based on the Euclidean distance between data points, offering flexibility to capture intricate relationships.

(d) **Sigmoid Kernel:**

$$K(x, y) = \tanh(\alpha x^T y + c) \tag{1.9}$$

The sigmoid kernel, often used in neural network architectures, introduces non-linearities through hyperbolic tangent functions.

Kernel functions are not restricted to SVMs and find widespread use in various algorithms, including kernelized versions of principal component analysis (PCA) and Gaussian processes. Their versatility in capturing complex relationships in the data makes kernel functions a crucial element in the machine learning toolbox. The selection of an appropriate kernel function depends on the nature of the data and the underlying patterns one aims to capture.

In Chapter 3, a fundamental question regarding the use of kernel functions is raised, and this issue is further amplified and explored in greater depth in Chapter 4.

1.3.2 Fuzzy Rule Based Systems (FRBS)

Dimensionality reduction can be done through feature selection using a fuzzy rule-based framework. In Chapter 6, this concept is exploited to identify a special type of network, referred to as a ”synergy network,” using synergistic pairs of genes/features. Fuzzy Rule-Based Systems

(FRBSs) are extensively applied in developing interpretable solutions for complex problems across various fields, including control engineering, robotics, and bioinformatics. These systems excel in addressing uncertainty, imprecision, and non-linearity, which are common aspects of real-world challenges. An FRBS extends the traditional rule-based system, comprising rules in the form "If FA Then FC", where both the antecedent FA and the consequent FC are fuzzy sets [32, 33].

To understand fuzzy sets, it is essential to contrast them with classical sets based on bivalent logic. Classical sets operate on the principle of an instance being either a member or not. In this binary system, the membership of elements is strictly 1or 0. In contrast, fuzzy sets introduce the concept of degrees or grades of membership. In a fuzzy set, an element's membership value lies within the range $[0, 1]$. A fuzzy set employs a membership function to define the gradation of memberships for its elements. Formally, a membership function on X is any function $\mu(x) : X \to [0, 1]$, where $x \in X$. Consequently, a fuzzy set A can be defined formally as $A = \{\langle x, \mu(x)\rangle | x \in X\}$ [34]. The fuzzy set theory facilitates the modeling of linguistic variables and rules using natural language.

In terms of rule structures, FRBSs come in three primary types: Mamdani Assilian (MA) type, Takagi Sugeno Kang (TSK) type, and classifier type [32, 33]. These distinctions provide flexibility in adapting FRBSs to different problem domains and varying requirements.

In the Mamdani-Assilian (MA) model, both the antecedent and consequent propositions of a rule are defined using linguistic variables. A rule in this model is expressed in the following form:

$$\text{If } x_1 \text{ is } FA_1 \text{ and } x_2 \text{ is } FA_2 \text{ and } \dots \text{ and } x_P \text{ is } FA_P \text{ Then } y \text{ is } FC,$$

where x_i and y are linguistic variables, and FA_i and FC are linguistic values. The key components of an MA model are illustrated in Figure 1.1.

The process in the MA model involves fuzzifying a crisp input into linguistic values through a fuzzification process. Subsequently, the fuzzified input undergoes processing in the inference engine based on a set of 'If-Then' fuzzy rules.

This approach allows for the representation of knowledge and decision-making in a linguistic form, making it particularly suitable for systems dealing with uncertainty and imprecision. The MA model provides a framework for translating crisp inputs into fuzzy linguistic terms, facilitating the utilization of expert knowledge in the form of fuzzy rules for effective decision support. Certainly, here is a rephrased version of the provided text to avoid plagiarism:

In the fuzzy logic framework, the knowledge base consists of two integral components: the rulebase (RB) and a database (DB). The fuzzy rules are stored in the rulebase, defining relationships between linguistic variables. Simultaneously, the corresponding definitions of fuzzy sets and parameters for membership functions are stored in a database. Together, the rulebase and database form the knowledge base, providing the system with a structured repository of information.

Outputs generated by the inference engine are linguistic values, which undergo defuzzification to convert them into crisp values. The defuzzification process transforms fuzzy linguistic outputs into precise and actionable information.

The Takagi-Sugeno-Kang (TSK) model [35] shares similarities with the Mamdani-Assilian (MA) model. However, in the TSK model, the consequent part does not involve fuzzy sets; instead, it is a function of the input variables. A rule in the TSK model takes the form:

$$\text{If } x_1 \text{ is } FA_1 \text{ and } x_2 \text{ is } FA_2 \text{ and } \dots \text{ and } x_P \text{ is } FAP \text{ Then } y = h(x_1, x_2, \dots, x_P).$$

In the TSK model, the consequent function yields a crisp output, eliminating the need for a separate defuzzification step. Nevertheless, different rules may produce distinct outputs for a given input, requiring aggregation to obtain the final rulebase output.

For classification rules in the TSK model, the antecedent remains the same, but the consequents are class labels, taking the form:

If x_1 is FA_1 and x_2 is FA_2 and ... and x_P is FAP Then class is k.

It is noteworthy that the TSK model is versatile and can also be employed for classification tasks.

1.4 Literature review

Dimensionality reduction serves as a fundamental technique for eliminating noisy or redundant features from datasets. It generally falls into two categories: feature extraction &selection. Extraction of feature extraction involves transforming original features space into a lower dimension feature space. The new features are typically combinations of the originals. Common feature extraction methods include Principal Component Analysis (PCA), Canonical Correlation Analysis (CCA), Linear Discriminant Analysis (LDA). Conversely, the goal of feature selection is to identify a smaller subset of features that minimize redundancy while maximizing their relevance to the target, such as class labels in classification tasks. Both feature extraction and feature selection offer benefits like enhancing learning performance, reducing computational complexity, building more generalizable models, and decreasing storage requirements. Feature extraction involves reducing the dimensionality of the original feature space by creating new features that are combinations of the originals. However, interpreting these new features can be challenging as they lack a direct physical interpretation related to the original features.

Feature selection involves selecting a subset of features from the original feature set without altering their representation, thereby preserving the geometrical structure of the original features. This aspect makes feature selection advantageous in terms of readability and interpretability, which is crucial in various practical applications such as identifying relevant genes for a specific disease or constructing sentiment lexicons for sentiment analysis. While feature extraction and feature selection are often treated as distinct techniques, sparse learning methods like l_1 regularization can convert feature extraction (transformation) methods into feature selection methods, bridging the gap between the two approaches [36]. High-dimensional data poses significant challenges to existing learning methods due to the curse of dimensionality [37, 38]. This phenomenon, well-recognized in the field, occurs when the presence of numerous features leads to overfitting, resulting in decreased performance of learning models. Dimensionality reduction techniques have emerged as a crucial area of study within machine learning and data mining to tackle this challenge effectively. Among these techniques, feature selection stands out as a popular approach for reducing dimensionality. It involves selecting a subset of relevant features from the actual set based on specific relevance evaluation criteria. This selection process typically improves learning performance, such as higher accuracy in classification tasks, while also reducing computational costs and enhancing model interpretability.

Researchers have also devoted considerable attention to the development of unsupervised feature selection methods. Unlike supervised methods, unsupervised feature selection does not rely on class labels, making it a less constrained search problem. Instead, it often relies on clustering quality measures and other unsupervised techniques to identify relevant feature subsets. However, with high-dimensional data, it becomes challenging to uncover meaningful features without imposing additional constraints. Additionally, a key challenge in unsupervised feature selection is determining objective measures to evaluate the effectiveness of the selected features [39]. Reference [40] provides a comprehensive review of unsupervised feature selection methods.

Feature selection algorithms can be classified based on the availability of labels in the training set, resulting in three main categories: supervised [41, 42], unsupervised [43, 44], and semi-supervised feature selection [45,46]. Supervised feature selection methods are further subdivided into filter models, wrapper models, and embedded models [47]. In filter models, feature selection operates independently of classifier learning to prevent the biases of one algorithm from interfering with the other. These methods rely on various measures of the training data's general characteristics, such as distance, consistency, dependency, information, and correlation. Representative algorithms in this category include Relief [48], Fisher score [1], and Information Gain-based methods [15].

Filter model methods in feature selection do not rely on clustering algorithms to assess feature quality; instead, they evaluate each feature's score based on specific criteria. Features with the highest scores are then selected, hence the term "filter" as it sifts out irrelevant features based on predefined criteria. In filter model methods, feature evaluation can take either univariate or multivariate approaches. Univariate evaluation analyzes each feature independently of the feature space, which enhances speed and efficiency. However, it may struggle with redundant features. In contrast, multivariate evaluation considers features in relation to each other, allowing for the handling of redundant features. An example of a univariate filter model is SPEC, although it has been extended to a multivariate approach [49]. Other criteria used in filter model methods for feature selection in clustering include feature dependency, entropy-based distance [50], and Laplacian score [51].

The wrapper model utilizes a clustering algorithm to evaluate the efficacy of selected features. It follows a process where it

(a) identifies a subset of features,

(b) evaluates clustering quality using this subset, and

(c) iterates these steps until achieving the required quality.

However, examining all potential feature subsets becomes impractical for datasets with high dimensions, prompting the adoption of heuristic search techniques to narrow down the search space. While the wrapper model incurs higher computational costs compared to the filter model, it often produces superior clustering outcomes by prioritizing feature selection that optimizes quality. Nonetheless, it tends to favor the chosen clustering method. Various wrapper feature selection methods for clustering have been proposed, employing different combinations of search strategies and clustering algorithms. For example, the approach described in reference [43] integrates maximum likelihood criteria for feature selection and utilizes a mixture of Gaussians as the clustering algorithm. Other methods leverage traditional clustering techniques such as k-means in conjunction with diverse search strategies for feature selection [52].

Real-world datasets often contain imperfections, such as irrelevant and redundant features. Feature selection techniques offer a way to address this issue by identifying and removing unnecessary elements, thereby reducing storage and computational costs without significantly impacting information or learning performance. In the context of a specific application, features can be classified into four groups: necessary features, derogatory features, redundant features, and indifferent features [19].

- Necessary features are essential for solving the target problem.
- Derogatory features impede learning and should be removed.
- Redundant features provide useful information but are not all necessary for problem-solving.
- Indifferent features, such as those with constant or nearly constant values, neither contribute nor cause problems.

By distinguishing between these categories, feature selection methods can effectively streamline datasets for improved performance and efficiency.

This study focuses on clustering within the Fuzzy framework. To enhance the clustering performance of Fuzzy C-means (FCM), various methods have been proposed in the literature. In a recent study, a novel approach was introduced, focusing on feature selection within the framework of adaptive weighted fuzzy clustering with intra-cluster data divergence [53]. Evolutionary strategy to optimize the weight matrix, as suggested in [54], which proved effective in enhancing the clustering process. Additionally, in [55], they proposed a technique for feature selection during unsupervised clustering. They emphasized two key advantages of their approach: firstly, it leads to more interpretable and meaningful clusters, and secondly, it contributes to improving the learning behaviour of the clustering algorithm. A methodology termed feature-reduction FCM (FRFCM) was introduced in [56]. In this approach, feature-weighted entropy was incorporated into the FCM objective function, alongside the establishment of a learning schema to optimize parameters., followed by the elimination of irrelevant feature components. A novel automatic local feature weighting scheme has been introduced to accurately assign weights to the features within each cluster [57]. Additionally, a cluster weighting process has been implemented to overcome the initialization sensitivity inherent in FCM. The feature weighting and cluster weighting procedures are carried out concurrently and autonomously throughout the clustering process, leading to the formation of high-quality clusters irrespective of the initial center selections. Authors in [58] proposed the integration of attribute weight entropy regularization into the fuzzy C-means algorithm for feature selection.

Conventional clustering techniques often rely on the Euclidean distance measure, treating each feature equally. While sufficient in scenarios where all features are equally relevant to every cluster, this approach may be inadequate when features vary in importance. To address this limitation, feature weighting algorithms have emerged, aiming to assign weights to each feature based on their importance in distance computations, thereby enhancing clustering algorithm performance [59]. The development of feature weighting algorithms has gained significant attention in cluster analysis. These algorithms assign weights to features when calculating the distance between data points. Two notable representatives include WFCM and W-k-means [60, 61]. In WFCM, feature weighting occurs before the clustering process, while in W-k-means, it is integrated into the clustering procedure itself. Generally, feature weighting algorithms learn weights for features, assigning larger values to more important ones. As a result, they typically outperform conventional clustering algorithms that rely solely on the Euclidean distance measure. An improved soft subspace clustering algorithm for brain MRI segmentation has been used in [62]. Taking within cluster and between cluster information, an adaptive soft subspace clustering has been proposed in [63].

Fuzzy clustering has gained prominence as a valuable technique for uncovering the underlying structure within datasets. Kernel methods are used with fFCM, leading to the development of kernel-based fuzzy clustering. In recent years, there has been a notable surge in interest surrounding kernel-oriented clustering methods [64, 65]. Several kernel-based fuzzy clustering techniques derived from Fuzzy C-Means (FCM) have been developed [66–70]. These techniques fall into two main categories. The first category, referred to as KFCM-F (Kernel FCM where prototypes in Feature space), is primarily utilized to find the pattern in incomplete data [69, 71]. The prototypes are in feature space in KFCM-F during clustering.

For instance, in [71], authors utilized kernel version of fuzzy clustering for imputing missing data in the iris dataset, selectively discarding the attributes in the process. They observed slight improvements in the misclassification rate when compared to alternative imputation methods. Likewise, in [69], researchers introduced a weighted version of kernel fuzzy clustering approach designed for handling both complete & incomplete datasets. Another class of kernel FCM algorithms, referred to as KFCM-K (Kernel-version FCM where prototypes in Kernel space), has been identified in the literature [67, 72, 73]. In KFCM-K, prototypes are implicitly preserved within the kernel space throughout the clustering process. As a result, an inverse mapping step becomes essential for retrieving prototypes within the original feature space.

In the realm of clustering methodologies, Fuzzy C-means (FCM) has gained wide acceptance for its versatility in data partitioning [31]. Researchers frequently employ FCM in various applications, showcasing its effectiveness. Particularly in image segmentation, FCM has been extensively utilized, with studies often exploiting neighborhood pixel information to improve segmentation accuracy [74,75]. However, a notable research gap exists – while FCM with spatial information is commonly used in image-based studies, its application to non-image datasets remains largely unexplored. This chapter strategically incorporates local/contextual information, demonstrating enhanced clustering performance compared to conventional FCM. On the other hand, the K-means clustering method, known for its simplicity, strictly assigns each image pixel to a single group. In contrast, Fuzzy C-means (FCM) introduces membership degrees, allowing pixels to belong to multiple clusters simultaneously based on their membership degrees [76]. Despite its significance in various applications, FCM has shortcomings such as the lack of consideration for spatial context in images, making it vulnerable to noise and imaging artifacts like intensity inhomogeneity. Additionally, FCM may converge to local optima due to poor initialization [77]. To address these limitations, modifications have been proposed to enhance its robustness for image segmentation.

For instance, Robust FCM (RFCM) introduced in [78] incorporates a spatial penalty term into the objective function, yet its objective function exhibits complex variations in the membership function. Spatial FCM (SFCM) [79] adjusts the membership function by integrating spatial information, showing improved performance but remaining sensitive to serious noise. Fast Generalized Fuzzy C-means (FGFCM) proposed in [80] uses spatial information for brain MRI segmentation, but its performance degrades with significant noise. In [81], maximizing fuzzy partition entropy with 2D histogram for MRI segmentation is explored, but it suffers from high time complexity. Modified versions of FCM, such as non-local FCM (NL/FCM) [82] and FCM with non-local spatial information [83], address robustness against noise and inhomogeneity but have limitations under high noise levels. In [84], an updated FCM approach is developed by modifying the objective function to include a gray-difference coefficient, aiming to enhance performance against noise. These adaptations illustrate ongoing efforts to address the limitations of FCM in image segmentation, with each method presenting its advantages and trade-offs.

1.5 Scope of the Thesis

Through our survey, several compelling areas of inquiry have emerged within the realm of pattern recognition, sparking our interest in the following key topics:

- An intriguing question arises regarding the preservation of original data geometry amidst high-dimensional feature selection. Can we devise methodologies that not only select pertinent features but also maintain the intrinsic structure of the data? The study also focuses on a special type of features, known as synergistic pairs, related to prostate cancer.
- Kernel clustering offers a promising approach by transforming data into higher dimensions, ostensibly enhancing clustering efficacy. Yet, what unfolds in these higher dimensions? Is the utilization of kernel space a secure and reliable strategy? Our investigation seeks to delve deeper into the mechanisms underlying kernel clustering and assess its robustness and applicability in practical scenarios.
- Fuzzy C-Means (FCM) has proven effective in image data by capitalizing on spatial information. The focus is now on exploring the applicability of Fuzzy C-Means (FCM) techniques to non-image data that exists in regular Euclidean space.

The objective of this thesis is to explore novel approaches in pattern recognition, focusing on preserving data geometry in high dimensions for feature selection, understanding the efficacy and safety of kernel clustering, and extending Fuzzy C-Means to non-image datasets in Euclidean space to enhance pattern recognition capabilities across diverse domains.

In this thesis, five research problems within the field of pattern recognition are addressed using Fuzzy set theory. In Chapter 2, a novel method for feature selection during clustering is proposed to achieve a target partition. To enhance clustering results, the application of Kernel space is explored in Chapter 3, with a particular focus on high-dimensional contexts. The thesis critically examines the use of the Kernel trick for clustering datasets in high dimensions, highlighting potential misguidance for researchers. In chapter 4, a thorough analysis emphasizes the sensitivity of Kernel parameters and provides guidelines for their optimal utilization, along with a method for visualizing data structure in high-dimensional space. While Fuzzy C-Means (FCM) is commonly applied to image data with spatial constraints, an innovative approach is presented in Chapter 5, extending FCM's application to non-image datasets by exploiting neighboring information. In Chapter 5, the focus is on feature selection, with particular emphasis on the utility and effectiveness of identifying synergistic pairs of features to address a significant problem related to prostate cancer. An overview on the technical contributions of this thesis is briefly mentioned below.

1.5.1 Weighted Fuzzy C-Means: Unsupervised Feature Selection to Realize a Target Partition

In Chapter 2, a novel unsupervised feature selection approach is presented, rooted in regularized weighted fuzzy C-means (WRFCM) clustering [85]. When the primary objective is clustering, our aim is to identify a subset of features that can yield a similar partition matrix to that obtained from the original high-dimensional data using a clustering algorithm. To address this, a unique objective function tailored to the FCM clustering algorithm is introduced. This method seamlessly integrates feature selection within the WRFCM framework, prioritizing features to uphold the FCM-based target partition. Our method's performance is assessed using metrics such as Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and KM-index. NMI and ARI gauge the agreement between clusters, comparing the partition in the lower dimension with that of the original data. In contrast, the KM-index measures the disagreement between the two partitions. Experimental results on both synthetic and real datasets demonstrate the effectiveness of our approach in selecting informative features. By addressing a critical gap in unsupervised feature selection, our method proves valuable for real-world applications.

The ability to maintain the integrity of clustering structures while reducing dimensionality enhances the utility of our approach across diverse domains.

1.5.2 Is it Rational to Partition a Data Set using Kernel Clustering?

In Chapter 3, the rationale behind certain clustering approaches, particularly those conducted in the kernel space, is critically questioned [86]. Using simple datasets, it is argued and demonstrated that seeking clusters in the kernel space may not align with the objective of identifying clusters in the original data. This is due to the potential difference in geometry between the data in the kernel space and that in the original feature space. Specifically, our findings highlight two key issues: (1) an inappropriate choice of the number of clusters may lead to counterintuitive results, such as merging distant clusters instead of nearby ones, and (2) selecting improper kernel parameters can significantly impact the extracted clusters, even introducing arbitrary cluster structures absent in the original data. It is essential to note that our intention is not to dismiss the potential of kernel clustering to yield desirable results. Indeed, kernel clustering can be valuable provided appropriate kernel parameters are chosen.

1.5.3 What and When can We Gain from the Kernel Versions of C-Means Algorithm?

In Chapter 4, this argument is further amplified and explored in greater depth. The central question addressed here is whether clustering object data in the kernel space is a viable approach [87]. Our arguments are as follows:

(a) **Clustering Objective:** The primary objective of any clustering algorithm is to identify natural subgroups within X based on a measure of similarity between vectors.

(b) **Nonlinear Transformation:** If data X is transformed into Y in another space through a nonlinear transformation, clustering in Y is useful only if it helps identify the same clusters present in X.

(c) **Preservation of Structure:** If Y maintains the same structure as X, the use of Y may not provide any advantage.

(d) **Structural Changes:** However, if Y induces structural changes that make extracting desired clusters from X easier, clustering in Y becomes useful.

(e) **Risk of New Structures:** Clustering in a transformed space can yield strange and irrelevant clusters if the transformation introduces new (non-existent) structures.

(f) **Philosophical Issue:** The decision to cluster in a transformed space hinges on whether it aids in identifying clusters present in X, posing a philosophical challenge independent of the clustering algorithm or the specific transformation (kernel function) employed.

(g) **Practical Limitations:** For data beyond $2D/3D$, determining the benefit of kernel clustering is challenging, as visual assessment becomes impractical.

To substantiate these claims, the arguments are demonstrated using both synthetic and real datasets, employing visual assessments and metrics such as Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and cluster instability. Additionally, Sammon's nonlinear projection method is utilized to provide a crude visual representation of data in the kernel space. While addressing the issue of choosing appropriate kernel function parameters, a definitive solution remains elusive. Lastly, the interaction between kernel parameters and algorithmic parameters is discussed, highlighting their combined influence on clustering outcomes.

1.5.4 Fuzzy Clustering Exploiting Neighbourhood Information for Non-image Data

In Chapter 5, a limitation of the conventional Fuzzy C-Means (FCM) algorithm is addressed, specifically its tendency to overlook crucial neighborhood information, particularly in non-image datasets residing in Euclidean space [88]. An extended version of the traditional FCM algorithm is introduced, specifically tailored to leverage neighborhood information in non-image datasets within the Euclidean domain. This innovative approach not only embraces the inherent fuzziness of individual data points but also strategically leverages spatial contextual information present in the Euclidean space. Our method goes beyond the limitations of conventional FCM, introducing a nuanced consideration of neighborhood information for improved clustering outcomes. The applicability and superiority of the proposed approach over the traditional FCM algorithm are demonstrated using both synthetic and real datasets. By seamlessly incorporating neighborhood information in non-image contexts, our method exhibits enhanced clustering performance, showcasing its ability to outperform the conventional FCM algorithm. The results underscore the effectiveness of our approach in capturing and leveraging spatial relationships within diverse datasets.

1.5.5 Finding Synergy Networks from Gene Expression Data: A Fuzzy Rule Based Approach

In Chapter 6, the fifth significant contribution to this thesis, the topic of feature selection is revisited with a focus on a distinct category of features known as synergistic pairs of features [89]. This chapter is dedicated to addressing a specific yet crucial problem related to prostate cancer. Our objective is to identify a unique type of network known as a "synergy network" among genes. Genes interact directly and indirectly, regulating the expression levels of one another. The objective is to discover synergistic gene pairs that collaborate in the context of a specific disease, leading to the formation of a network of such synergistic genes. Limitations in existing information-theoretic methods for identifying synergy networks are addressed by proposing a novel approach based on fuzzy rule-based systems. This method leverages fuzzy rules, offering a natural framework for capturing the essential attributes of synergistic relations. This is the first attempt, to our knowledge, to utilize fuzzy modeling for the discovery of synergy networks. The system employs a set of human-understandable rules generated at a low cost for each pair of genes. The proposed method is applied to two prostate cancer datasets, demonstrating its ability to uncover gene pairs that collaborate in relation to prostate cancer. The results are shown to be statistically significant, and the relevance of the identified genes to cancer biology is thoroughly discussed.

1.5.6 Conclusions, Limitations, and Future Scopes

The thesis concludes in Chapter 7, where the limitations of the presented work are critically examined, and potential avenues for future research and methodological extensions are explored. This chapter provides insights into the boundaries of our current research and outlines directions for further exploration and development.

In the following five subsequent technical chapters (Chapter 2 to Chapter 6), the research contributions of this thesis are presented in detail. The next chapter (Chapter 2) presents an unsupervised feature selection methods where the reduced dimensional data will preserve the local geometry of the high dimensional data.

Chapter 2

Weighted Fuzzy C-Means: Unsupervised Feature Selection to Realize a Target Partition

2.1 Introduction

In this chapter, we deal with unsupervised feature selection for clustering, which comes under the broad umbrella of dimensionality reduction. Hence, we begin with a general overview of the dimenaionality reduction problem. Real data, in addition to useful features, may contain derogatory features, strongly correlated features as well as indifferent features with respect to the target problem at hand [90]. The selection of useful features and removal of derogatory and indifferent features are important for any data. And this problem becomes more important for high-dimensional datasets. For example, in gene expression data analysis, datasets often comprise of a large number of features but a relatively limited number of instances [91]. This scenario gives rise to the challenge commonly referred to as the "curse of dimensionality" [37]. Consequently, it becomes essential to identify a small set of discriminative genes from a pool of thousands, while developing an effective diagnostic classification systems.

Dimensionality reduction can be achieved through two main approaches: feature extraction and feature selection. Feature extraction maps the original high-dimensional feature space into a new feature space with a reduced dimension. This new space typically comprises of a linear or non-linear combination of the original features. Methods like Principle Component Analysis (PCA) [92], Linear Discriminant Analysis (LDA) [93], Singular Value Decomposition (SVD) [94] and ISOMAP [7] fall under feature extraction techniques.

Feature selection, on the other hand, aims to choose the relevant features from the dataset, effectively discarding derogatory, indifferent, or redundant ones [1,90,95,96]. Feature selection offers several advantages, such as improving computational efficiency in predictive modeling, providing deeper insights into the underlying structure of the data, and enhancing predictive performance by eliminating derogatory and indifferent features from the dataset of consideration. Since the focus of the problem is feature selection, the discussion shall be restricted to either "explicit" or "implicit" feature selection. In the context of clustering, often weights are associated with features to find better clusters without explicitly using the weights to remove/select features. This is an important class of methods, which although explicitly does not remove/select features, a few such methods will be discussed, as these methods, in some sense, implicitly select or reject features.

Feature selection techniques are generally categorized into three groups: Wrapper Models, Filter Models, [47] and Embedded Models. Wrapper methods involve a clearly defined objective function that must be optimized by selecting a subset of features. Typically it requires evaluation of different subsets of features. In contrast, filter models rely on intrinsic properties of the features themselves. Filter methods are typically computationally less demanding, whereas wrapper methods generate a feature set fine-tuned to a specific predictive model requiring more computation. The third category of methods, i.e, Embedded Methods choose features as an integral part of the model construction process. Embedded methods find the useful features simultaneously along with the model construction process. A notable instance of this is the Support Vector Machine Recursive Feature Elimination (SVM-RFE) [97] approach, where features are recursively eliminated with iterations based on the SVM model weights. The final set of retained features constitutes the reduced-dimensional space. Other examples of embedded methods include sparsity-driven feature selection algorithms [98, 99]. In the case of LASSO [95] (Least Absolute Shrinkage and Selection Operator), a linear regression model is constructed while enforcing less useful model coefficients to be zero, effectively selecting a subset of the most relevant features. There are families of neural network-based embedded methods which unlike SVM-RFE, require only a single round of the construction of the model [19, 90, 100–103]. In [100–102], a novel concept of feature attenuation gates is used to realize feature selection. On the other hand, methods in [19, 90, 103] use group LASSO regularization for feature selection.

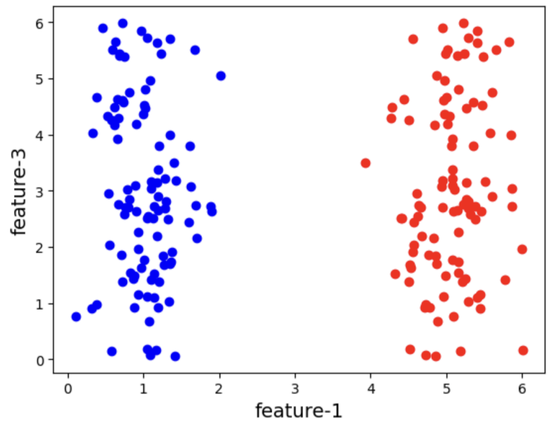
For the sake of completeness, note that dimensionality reduction techniques can also be classified primarily into two categories based on the nature of learning algorithms used: Supervised and Unsupervised. In supervised methods, each data point is associated with a target value (e.g., a class label), making feature selection straightforward. The primary objective is to identify a feature subset that can effectively accomplish the prediction or classification task.

In contrast, unsupervised methods lack target values, necessitating the use of task-independent criteria for feature selection. These methods aim to uncover underlying patterns within data without specific predictive objectives.

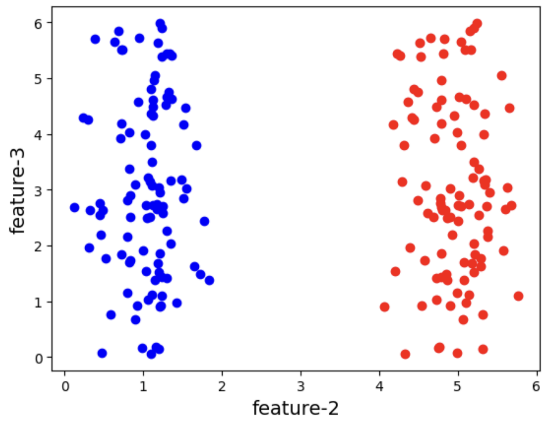
Both feature extraction and feature selection offer several benefits including enhanced learning performance, improved computational efficiency, reduced memory requirement, and the creation of more effective models. However, a drawback of the feature extraction lies in its creation of a new set of features, which complicates further analysis as the physical significance of these features in the transformed space becomes elusive. Conversely, feature selection retains some of the original features, preserving their physical meanings. This characteristic grants models greater readability and interpretability. Consequently, in numerous real-world applications like text mining and genetic analysis, feature selection often is considered the preferred approach [104–107].

Real-world data often come with imperfections, including irrelevant and redundant features. Employing feature selection helps to trim these unnecessary elements, cutting down storage and computational costs without significantly compromising information or learning performance. Given a target application, features can be categorized into four groups: necessary features, derogatory features, redundant features, and indifferent features [90]. Indifferent features are those which neither help nor cause any problem as there values remain almost constant. The necessary features are needed to solve the target problem, the derogatory features cause problem in the learning, so these should be removed, while redundant features are useful but not all are needed to solve the problem. To illustrate derogatory, necessary and redundant features a three-dimensional dataset is constructed. Two well separated clusters is taken each with 100 instances each in R^2. The corresponding features are referred to as f_1 and f_2. Each of the features f_1 and f_2 lies in [0, 6]. Then a third dimension is added as random values in [0, 6]. In Fig. 2.1, the pair-wise features are visualized using scatter plots. Fig. 2.1a depicts f_3 as a derogatory feature, which is incapable of effectively separating the two classes or clusters. Feature f_1 is a useful features as using f_1 only one can separate the clusters. In Fig. 2.1b, feature f_2 is also found to be a useful feature like $f_1$which is able to discriminate the two classes (clusters). Meanwhile, Fig. 2.1c depicts that both f_1 and f_2 are useful, but only one of f_1 and f_2 is enough, i.e, f_1, f_2 forms a redundant set as they are strongly dependent (High mutual information/Correlation). Consequently, eliminating one of f_1 and f_2, as well as f_3, will not adversely affect the learning performance.

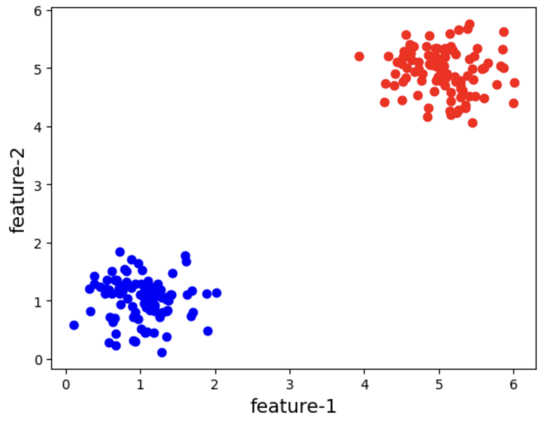
Numerous techniques have been introduced within the realm of supervised methods [19, 97, 100–102, 108, 109] showcasing a wealth of options for feature selection in supervised contexts. Conversely, there are rela-



(a) Derogatory features: f_3; Useful feature: f_1



(b) Derogatory features: f_3; Useful feature: f_2



(c) Redundant features: f_1 or f_2

Figure 2.1: An illustrative example of derogatory, useful and redundant features.

tively fewer methods available for unsupervised scenarios [110–115], as well as for the semi-supervised framework [45,116–118].

A few methods that are highly relevant to the present study are now discussed, as these methods explicitly or implicitly select features for the FCM algorithm.To enhance the clustering performance of FCM, various methods have been proposed in the literature. In a recent study, a novel approach is introduced, focusing on feature selection within the framework of adaptive weighted fuzzy clustering with intra-cluster data divergence [53]. Li et al. [54] also attempted to find feature weights for clustering using FCM. However, this approach differs from rest of the approaches in two aspects. First, it does not impose the restriction that the sum of the weights should be one. Second, unlike all other methods discussed, it uses a cluster validity index, in particular, the partition coefficient [31] as the objective function to optimize the weights using evolutionary strategy. In [55], authors have proposed a technique for feature selection during unsupervised clustering. This method is claimed to have two advantages: first, it leads to more interpretable and meaningful clusters, and second, it attempts to improve the learning behaviour of the clustering algorithm. A methodology termed feature-reduction FCM (FRFCM) has been introduced in [56]. This approach integrates feature-weighted entropy into the FCM objective function and establishes a learning schema to optimize parameters. During the iterative steps of the algorithms, it eliminates less important features using the weights. Hashemzadeh et. al [57] proposed a weighted Fuzzy-C-Means algorithm, where the weights are cluster-specific, i.e., for each cluster and each feature a separate weight is learned. Although, clusters are influenced by the cluster specific feature weights, authors do not explicitly select/reject features using the weights. Their algorithm uses the cluster weighting process to mitigate the initialization sensitivity of the FCM algorithm. Both the feature weighting and cluster weighting procedures are carried out concurrently and autonomously throughout the clustering process, that leads to the formation of good clusters irrespective of the initial choice of centers. Authors in [58] also proposed an integration of attribute weight entropy regularization into the Fuzzy C-Means objective function. Like [57], here also for each feature, cluster specific weights are learned. Here too, authors do not explicitly select features, however, their results confirm that for some clusters some specific features practically do not have any effect suggesting implicit rejection of those features for those clusters.

In the context of our method, the work of Yang et al. [119] is also worth mentioning. They have proposed feature reduction framework on feature weighted possibilistic-c-means (PCM). The model enhances the clustering performance by estimating the feature weights which are relevant for clustering. Consequently the algorithm can remove the irrelevant features.

It is important to note that the PCM is not being used here; instead, the FCM framework will be employed, which is possibly the most popular fuzzy clustering algorithm that has also been extended in many ways including deep clustering methods [120–122]. An unsupervised feature selection method is proposed, with the hope that the data in the reduced dimension will preserve the local geometry of the data in the original higher dimension. Now, a question arises what criterion would be useful to preserve the local geometry? In this context, one of the natural choices would be to preserve the cluster sub-structures present in the original data into the lower dimensional data.

Here, clusters extracted by the FCM algorithm on the original high dimensional data have been used and the corresponding partition matrix $M = [\mu_{ij}]_{c\times n}$ has been taken as the target partition; μ_{ij} is the membership of the j^{th} data point to the i^{th} cluster. Note that, the underlying philosophy of our method is very general. Thus, the partition matrix can be generated by any clustering algorithm. Even the approach can be extended to partitions induced by class label also. However, this study focuses solely on the unsupervised approach, utilizing the FCM partition.

2.2 Weighted Fuzzy-C-Means (FCM) clustering

Fuzzy-C-Means operates by partitioning a set of p-dimensional vectors $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ into c clusters, where $\mathbf{x_k} = (x_{k1}, \ldots, x_{kp})^T$ represents the kth data point for $k = 1, \ldots, n$. The clusters are defined by a fuzzy partition matrix U. The vector $\mathbf{v}_i = (v_{i1}, \ldots, v_{ip})^T$ represents the centroid of the ith cluster for $i = 1, \ldots, c$. FCM is an extension of HCM (Hard c-means), differing in that, FCM generates a fuzzy partition while HCM produces a crisp partition.

In FCM, for the kth instance $\mathbf{x}_k$ $(1 \leq k \leq n)$ and the ith cluster centroid $\mathbf{v}_i$ $(1 \leq i \leq c)$, a membership degree u_{ik} $(0 \leq u_{ik} \leq 1)$ denotes the degree to which $\mathbf{x}_k$ belongs to the ith cluster which intuitively defines how close $\mathbf{x}_k$ is to the centroid $\mathbf{v}_i$. This membership degree results in a fuzzy partition matrix $U = (u_{ik})_{c\times n}$. The objective of FCM is to determine the set of cluster centroids $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c\}$ and the fuzzy partition matrix U by minimizing the objective function J.

The objective function J is defined as follows [31]:

$$J = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} \|\mathbf{x}_k - \mathbf{v}_i\|^2 \tag{2.1}$$

such that

(a) $0 \leq u_{ik} \leq 1; \quad \forall i, k$

(b) $\sum_{i=1}^{c} u_{ik} = 1; \quad \forall k$ and

(c) $0 < \sum_{k=1}^{n} u_{ik} < n. \quad \forall i$

Here, $m > 1$ is a fuzzifier (typically $m = 2$), $\|\mathbf{x}_k - \mathbf{v}_i\|^2$ represents the squared Euclidean distance between $\mathbf{x}_k$ and cluster center $\mathbf{v}_i$. Minimizing J involves iteratively updating U and V to find the optimal V and U leading to a fuzzy partition of the dataset X into c clusters. A detailed overview on objective function based fuzzy clustering can be found in [123].

Weighted Fuzzy C-Means (WFCM) [124] is a robust clustering algorithm that extends the classical Fuzzy C-Means (FCM) method by integrating feature weighting into the clustering process. FCM is well known for its ability to handle fuzziness in data by allowing data points to belong to multiple clusters with varying degrees of membership. WFCM enhances this approach by incorporating the significance of individual features through weighting, leading to more adaptable and refined cluster assignments.

The essence of FCM lies in assigning cluster memberships to data points based on their proximity to cluster centers. However, in real-world scenarios, not all features hold equal importance or relevance in determining the similarity between data points, i.e, the similarity between the associated objects represented by data points. WFCM addresses this limitation by introducing weights that emphasize the significance of certain features over others during the clustering process. This adaptation enables the algorithm to consider the varying impact of different features, thereby enhancing its ability to handle complex datasets with heterogeneous feature contributions.

The core principle behind WFCM involves iteratively updating cluster centroids and membership degrees of data points while incorporating feature weights. These weights encapsulate the relative importance of features in defining similarities between data instances. By assigning higher weights to more influential features and lower weights to less significant ones, WFCM refines the clustering process, resulting in more “accurate and tailored” cluster assignments.

The WFCM framework finds extensive use in diverse fields such as pattern recognition, image segmentation, bioinformatics, and data mining. Its adaptability to handle data with varying feature makes it a valuable tool in scenarios where traditional clustering methods might fall short in capturing the intricacies of the dataset.

As already mentioned, WFCM, uses a non-negative weight for each feature. These weights determine the relative importance of features in constructing the clustering solution. Considering a collection of objects

$O = \{o_1, o_2, \cdots, o_n\}$, associated with observations $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ in R^p, where $\mathbf{x}_i$ represents the characteristics/features describing the object o_i. The primary goal of WFCM [124] involves minimizing the objective function:

$$J_m(U, V, W) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m \sum_{j=1}^{p} w_j (x_{kj} - v_{ij})^2 + H \tag{2.2}$$

subject to the constraints:

(a) $u_{ik} \in [0, 1] \forall i, k,$

(b) $w_j \in [0, 1] \; \forall j;$

(c) $0 < \sum_{k=1}^{n} u_{ik} < n \; \forall i;$

(d) $\sum_{i=1}^{c} u_{ik} = 1 \; \forall k,$

(e) $\sum_{j=1}^{s} w_j = 1.$

Note that, to get a soft subspace partition the penalty H is used. Chan et al. [125] proposed a penalty term $H = \sigma \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik} \sum_{j=1}^{p} w_{ij}^{\alpha}$ where $\alpha > 1$. In [126] they proposed the penalty $H = \epsilon \sum_{i=1}^{c} \sum_{j=1}^{p} w_{ij}^{\alpha}$. An entropy based penalty was proposed in [127] where $H = \gamma \sum_{i=1}^{c} \sum_{j=1}^{p} w_{ij} ln(w_{ij})$. In all these cases $w_{ij} \in [0, 1], \sum_{j=1}^{p} w_{ij} = 1; i = 1, 2, \cdots, c$. Various forms of H are used by different authors [124, 128]. The negative Shannon entropy has been chosen to control the weights, as used in [127].

The weight w_j is associated with the jth features; $j = 1, \cdots, p$; p represents the number of features. Here features with higher weights wield greater influence on the clustering process compared to those with lower weights. With $m > 1$ as the fuzziness parameter, it regulates the impact of members' ratings. In the proposed method, another regularizing term is used, which primarily drives the feature selection process.

In the next section, the proposed method involves further regularizing the weighted FCM.

2.3 Proposed method

The FCM algorithm uncovers patterns in data by minimizing an objective function. It represents data structure through a fuzzy partition matrix and a set of "c" prototypes. When utilizing the Euclidean distance, FCM tends to prioritize spherical cluster shapes. However, the weighted Euclidean distance provides greater flexibility, enabling the exploration of ellipsoidal shapes aligned with the axes of individual coordinates. The Fuzzy-c-Means, either uses the Euclidean distance or some other inner product-induced distance measure and consequently it tries to find all clusters having similar hyper-spherical structure. The Gustafson-Kessel (GK) [129] algorithm, on the other hand, adapts the norms of each cluster and thereby can find different clusters with different shapes. However, the GK algorithm does not do any feature selection. Similarly, there are other clustering algorithms which adapt the matrices that induce the inner product to define the distance measures for different clusters. For example, the semi-supervised Metric Pairwise Constrained K-Means (MPC-K Means) clustering algorithm [130] integrates the constraints imposed by the data points with known cluster labels as well as the distance metric learning. However, this method also does not do any feature selection.

In this context, it is worth mentioning that the GK-FCM [129] can find non spherical clusters, but does not do feature selection.

In our proposed method, the following objective function is minimized

$$J_m(U,V,W) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m} \sum_{j=1}^{p} w_j (x_{kj} - v_{ij})^2 + \alpha \sum_{i=1}^{c}\sum_{k=1}^{n} (\mu_{ik} - u_{ik})^2$$
$$+\beta \sum_{j=1}^{p} w_j \ln(w_j) \tag{2.3}$$

subject to the constraints

(a) $u_{ik} \in [0,1]\ \forall i,k,$

(b) $w_j \in [0,1]\ \forall j$;

(c) $0 < \sum_{k=1}^{n} u_{ik} < n\ \forall i$;

(d) $\sum_{i=1}^{c} u_{ik} = 1\ \forall k$,

(e) $\sum_{j=1}^{p} w_j = 1$.

To account for constraints (4) and (5) using the Lagrange multiplier, the objective function is rewritten as follows:

$$J_m(U,V,W) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m} \sum_{j=1}^{p} w_j (x_{kj} - v_{ij})^2 + \alpha \sum_{i=1}^{c}\sum_{k=1}^{n} (\mu_{ik} - u_{ik})^2$$
$$+\beta \sum_{j=1}^{p} w_j \ln(w_j) - \sum_{k=1}^{n} \lambda_k (\sum_{i=1}^{c} u_{ik} - 1) + \nu(\sum_{j=1}^{p} w_j - 1) \tag{2.4}$$

For the FCM framework, the parameter $m > 1$ is called the fuzzifier because it controls the level of fuzziness in the terminal partition. For FCM, the most popular choice of m is $m = 2$. Note that for all our subsequent derivation and analysis, $m = 2$ is taken, which makes the mathematical derivation of the algorithm simpler.

In Equation (2.4), the 1^{st} term is the weighted FCM (WFCM) objective function. In the 2^{nd} term, a regularizer has been introduced, where $M = [\mu_{ik}]_{c\times n}$ is the target partition matrix generated by FCM using Equation (2.1) on the high-dimensional data. Our aim is to select features while maintaining the partition matrix as $M = [\mu_{ik}]_{c\times n}$ in reduced dimension. The membership matrix $U = [u_{ik}]_{c\times n}$ is updated, aiming to make it close to M. The second term of the objective function can be viewed as a similarity measure between M and U. So, our model looks for a set of features which can provide a partition of set of objects o similar to the target partition M obtained by FCM on the original high dimensional data. There could be many ways to generate the target partition leading to feature selection problems with different purposes. For example, the target partition can be generated using the Spherical k-Means [131] clustering algorithm which works well both for high and low-dimensional datasets. Moreover, as explained earlier, the scope of our study is primarily restricted to the cases where fuzzy clustering algorithms, like FCM works although it can work even beyond that.

To control the effect of this similarity measure, a hyper-parameter α is introduced. By increasing the value of α, more emphasis is placed on the similarity measure. Therefore, for a large α, the difference between M and U is expected to be small. This results in U becoming more similar to M, but it may also require using more features.

The 3^{rd} term imposes penalty for using more features. The 4^{th} and 5^{th} terms are the constraints imposed on U and $\mathbf{w}$ where λ and ν are the Lagrange multipliers.

The partition matrix U is updated using the Equation (2.5).

$$u_{ik} = \frac{\frac{1-\sum_{l=1}^{c} \frac{\alpha\mu_{lk}}{\sum_{j=1}^{p} w_j(x_{kj}-v_{lj})^2+\alpha}}{\sum_{l=1}^{c} \frac{1}{2\sum_{j=1}^{p} w_j(x_{kj}-v_{lj})^2+2\alpha}} + 2\alpha\mu_{ik}}{2\sum_{j=1}^{p} w_j(x_{kj}-v_{ij})^2 + 2\alpha} \tag{2.5}$$

Equation (2.5) is derived using the first order necessary condition of optimality with respect to U; i.e.,by setting the first derivative of Equation (2.4) with respect to u_{ik} to zero.

The prototype is updated using Equation (2.6)

$$v_{ij} = \frac{\sum_{k=1}^{n} u_{ik}^2 w_j x_{kj}}{\sum_{k=1}^{n} u_{ik}^2 w_j} \tag{2.6}$$

Equation (2.6) is obtained from the first order necessary condition of optimality of J with respect to $\mathbf{v}$.

Note that there is another hyper-parameter β involved here. For a large β, the model will assign similar weights to all the features. To get sparsity in feature selection, β should be low. The sensitivity of α and β has been discussed in the results section. To nullify the effect of data dimension, the coefficient α is normalized as $\frac{\alpha}{n.c}$, and β is normalized as $\frac{\beta}{\log(p)}$.

The weights are updated using Equation (2.7) which is obtained by setting the first derivative of Equation (2.4) with respect to w_j to zero.

$$w_j = \frac{exp[\frac{-2\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 (x_{kj} - v_{ij})^2 - \beta}{\beta}]}{exp[\frac{\beta \ln[\sum_{l=1}^{p} exp \frac{-2\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^2 (x_{jl} - v_{il})^2 - \beta}{\beta}]}{\beta}]}, \forall k \tag{2.7}$$

From Equations (2.6) and (2.7), it is easy to verify that required condition on U and $\mathbf{w}$ are satisfied.

The algorithm of our proposed model is described in Algorithm-2.1.

Algorithm 2.1 Proposed algorithm

```
(a) Fix $\alpha, \beta, c, maxItr, m = 2$ and $\epsilon > 0$, a tolerance level for termination.
(b) Set the target partition $M$ from FCM
(c) Initialize the set of prototypes, $V^0 = [V_i^0]_{c \times p}$
(d) Initialize the set of weights, $w^0 = (w_1^0, \cdots, w_p^0)^T$
(e) Initialize the membership matrix $U^0 = [U_{ik}^0]_{c \times n}$
(f) For $t = 1, 2, \cdots, maxItr$ do

        Update $U^{t+1} = [u_{ik}^{t+1}]_{c \times n}$ according to Equation (2.5)
        Update $\mathbf{v}^{t+1} = [v_{ij}^{t+1}]_{c \times s}$ according to Equation (2.6)
        Update $\mathbf{w}^{t+1} = (w_1^{t+1}, \cdots, w_p^{t+1})$ according to Equation (2.7)
    until
        $$\sum_{i=1}^{c}\sum_{k=1}^{n} |u_{ik}^{(t+1)} - u_{ik}^{(t)}| < \epsilon \tag{2.8}$$
    is satisfied.
(g) Return $U^{t+1}, \mathbf{v}^{t+1}, \mathbf{w}^{t+1}$.
```

In the next section, synthetic and real datasets are used to study the performance of the proposed model. It is again emphasized that the algorithm looks for features that can find partitions in the lower dimension that are similar to the FCM-induced partitions in the higher dimension.

2.4 Results

Experiments are conducted on 10 datasets, comprising seven real datasets (Table 2.1) and three synthetic dataset. Out of three synthetic datasets two are taken from [55]. First, the datasets used in [55] are considered. Next, a synthetic dataset is created, where two distinct clusters are generated in $\mathbb{R}^3$ and augmented with four random features. Although redundancy in the selected features is not considered in this thesis, there is a curiosity to evaluate the performance of the model on data with redundant features. Therefore, a few redundant features are also added.

Table 2.1: Real datasets used in the experiment [132].

Dataset	Instances	Features	Classes
Iris	150	4	3
Vertebral	310	6	3
Ecoli	336	7	8
Wine	178	13	3
WBC(D)	569	30	2
Sonar	208	60	2
Musk	476	166	2

2.4.1 Studies on the synthetic dataset

Frigui et al. [55] have used two synthetic datasets. The first dataset is generated in $\mathbb{R}^2$ having two Gaussian clusters as shown in Figure 2.2. Number of samples in each cluster is 20. They proposed two methods, *Simultaneous clustering and attribute discrimination-version 1* (SCAD1) and *Simultaneous clustering and attribute discrimination-version 2* (SCAD2). The aim of these two methods are to learn a different set of feature weights for each identified cluster.

Note that, the goal of the method is to learn one weight for each feature by selecting one set of features for all clusters. The hyper-parameters are set as $\alpha = 25000$ and $\beta = 100$. The feature weights vector is generated by

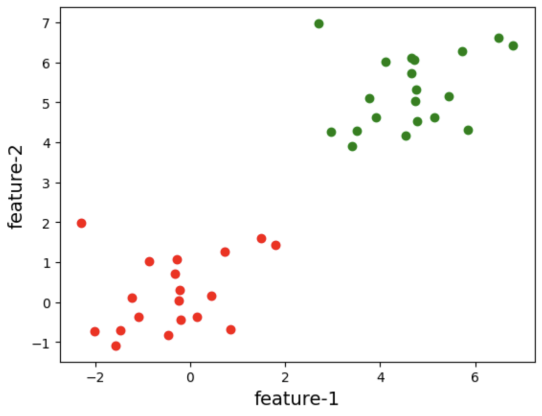
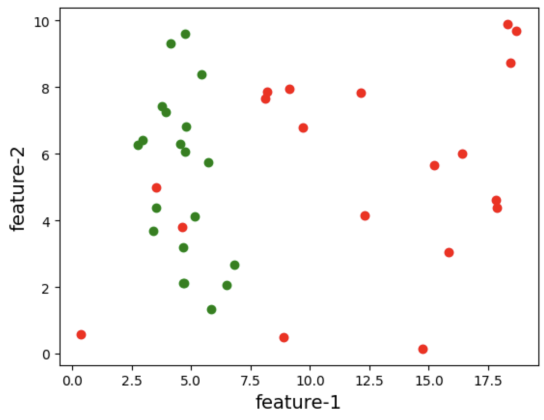


Figure 2.2: 2D data from [55].

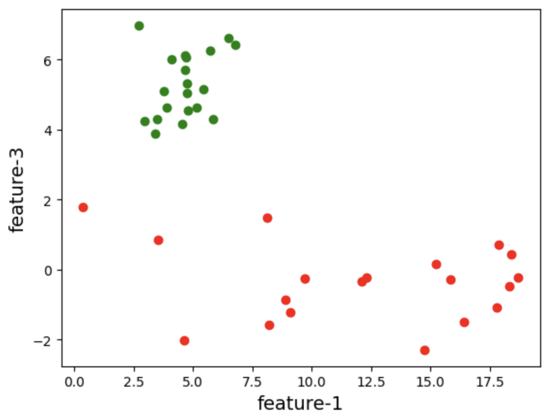
the method are $[0.497, 0.503]$ which is evidently in accordance with the findings of Frigui et al. In their second dataset, they increased the number of features to four by adding two irrelevant features to each cluster. The first two features of the first cluster are uniformly distributed in the intervals $[0, 20]$ and $[0, 10]$, respectively. Features two and four of the second cluster are uniformly distributed in the intervals $[0, 10]$ and $[0, 5]$, respectively. The pairwise plot of the features are shown in Figure 2.3.

All the figures that are associated with feature-2, it is evident from Figure 2.3 that feature-2 is not a good feature. The weight vector has been generated using the model for $\alpha = 25000$ and $\beta = 50$. The weights from the proposed method are $0.285, 0.145, 0.356$ and 0.214. So, our model assigned the least weight to the poor feature.

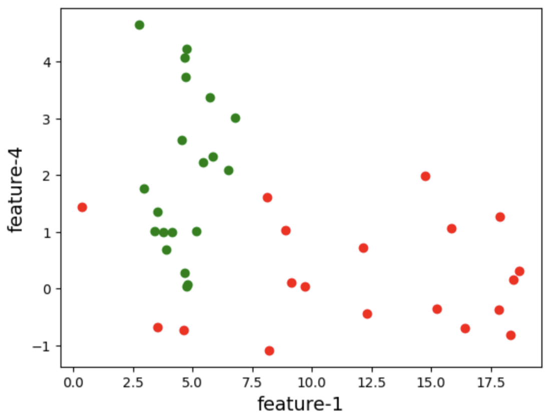
Next, the third synthetic data is used. It consists of two clusters in 3D by generating 200 points (100 for each cluster) from two Gaussian distribution ($\sigma = 1$) with center $(2, 2, 2)$ and $(7, 7, 7)$. Initially the clusters are formed in 3D space, each point in these clusters is augmented with four additional features. Two clusters are formed with centroids around the coordinates $(2, 2, 2)$ and $(2, 7, 7)$. Notably, the first feature does not contribute to discriminating between clusters. Two more features, the fourth and fifth, are introduced, with the fourth highly correlated with the second and the fifth with the third features, respectively. These additional features are generated by slightly modifying the values of features 2 and



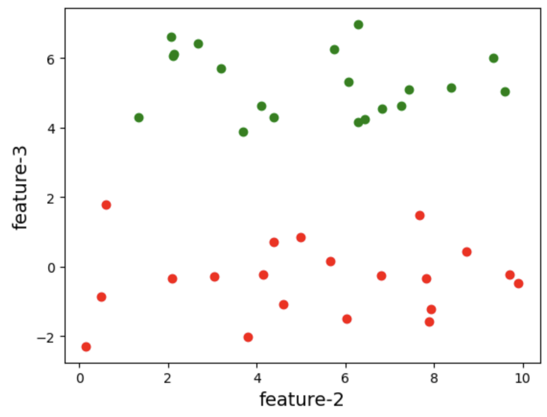
(a)



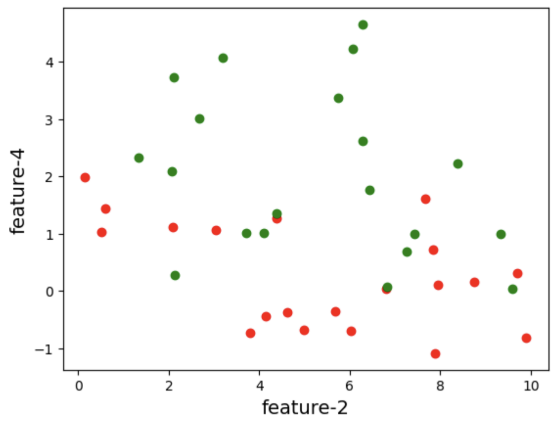
(b)



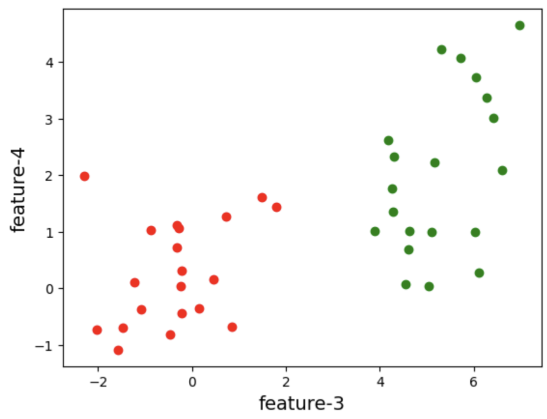
(c)



(d)



(e)



(f)

Figure 2.3: Pairwise plot of 4D data from [55]

3 respectively, incorporating small random noise. Two random features, ranging from -5 to 5, are added as the sixth and seventh features. Consequently, the crucial useful features are either the second or fourth and the third or fifth. The first feature is neutral, while the sixth and seventh features have no significant impact on the cluster structure. Since no penalty for redundancy was applied, the best features should be any subset of features $2, 3, 4$, and 5. However, to reduce the use of redundant features, a penalty or regularizer can be applied to prevent the use of strongly dependent features, ensuring that the selected features are less redundant [114].

To check the model performance, the confusion matrix is used. The best two selected features are chosen by the model, and subsequently, other features are considered. In each case, the misalignment between the original cluster labels and the generated cluster labels is checked. For this experiment, $\alpha = 25000$ and $\beta = 100$ are set. First, FCM is applied to the original dataset in seven dimensions, and the corresponding confusion matrix is shown in Table 2.2. The alignment between the actual cluster labels and the clusters generated by FCM is checked. The Kuhn-Munkres [133] algorithm is used to find the best relabeling of clusters.

Table 2.2: Confusion matrix of synthetic dataset for FCM.

		Predicted by FCM		
		Group-1	Group-2	Total
Actual	Group-1	100	0	100
	Group-2	0	100	100
	Total	100	100	200

The weights of the seven features obtained by our algorithm are 0.011, 0.236, 0.276, 0.22, 0.238, 0.01 & 0.01. It is clear that our model discarded the 1^{st}, 6^{th} and 7^{th} features straightway having weights $0.011, 0.01$, and 0.01 respectively. The top two features are feature-3 (weight=0.276) and feature-5 (weight=0.238). Since redundancy in the selected features is not being controlled, dependent features are also selected. The alignment between the natural groups in the dataset and the clusters obtained by FCM using only the two selected features is checked once again. Specifically, the alignment between the cluster labels in $2D$ and the clusters generated by FCM in $7D$ is assessed. The target partition matrix M in the model is set to the same partition matrix as generated by FCM on the high dimensional data. In this case, a confusion matrix is considered where the rows represent the FCM-generated results on the 7-dimensional data, and the columns represent the FCM-generated results using the selected two features. The confusion matrix is shown in Table 2.3 and it shows that the proposed model rightly find similar groups in the dataset using only two features.

Table 2.3: Confusion matrix of synthetic dataset taking feature 3 and 5.

		Predicted by Model		
		Group-1	Group-2	Total
Predicted by FCM	Group-1	0	100	100
	Group-2	100	0	100
	Total	100	100	200

To demonstrate the effect of the selected features on the model accuracy it is better to use some index in place of the confusion matrix. To validate the results further, two popular cluster validity indexes are used: Normalized Mutual Information (NMI) [134] and Adjusted Rand Index (ARI) [135]. Note that to compute ARI and NMI, the target partition generated by FCM on the original high-dimensional data is compared with the partition generated by FCM in the lower dimension. Here, the top features are subsequently added, and both NMI and ARI are checked. For every set of features, the targeted partition matrix, M, is set to the same partition matrix generated by FCM on the high-dimensional data. The results are shown in Table 2.4.

Table 2.4 reveals that our model selects the GOOD features and discards the BAD features. There may be some doubt regarding the confusion matrices shown in Tables 2.2 and 2.3 because, in the first matrix the data are

Table 2.4: NMI and ARI for different set of features for synthetic dataset.

features	NMI	ARI
3,5	1.0	1.0
2,5,3	1.0	1.0
4,2,5,3	1.0	1.0
1,4,2,5,3	1.0	1.0
6,1,4,2,5,3	1.0	1.0

aligned with the diagonal whereas in the other it is anti-diagonal. This is possible because the actual cluster labels are not being compared with those generated by the $2D$ data. Rather, the intention is to check whether the groups predicted by the proposed model in the lower dimension are properly aligned with the clusters found in the original dataset. To overcome this confusion, in the subsequent section, in place of confusion matrix, Kuhn-Munkres index (KM-index) is used which is computed as the number of mis-labeled points after the best relabelling of the clusters found in the lower dimension using the Kuhn-Munkres algorithm [133]. Kuhn-Munkres algorithm solves an assignment problem in polynomial time. Lets take an example to illustrate it. There are two partions of a given dataset using all features and using a subset of features. For ease of understanding, let us call the labels of the clusters found using all features as the "Actual" cluster labels and the labels of the clustes found in a reduced dimension as the "Predicted" labels. Suppose the confusion matrix between these two partitions is as shown in Table 2.5.

Table 2.5: Confusion Matrix.

		Predicted		
		Group-1	Group-2	Total
Actual	Group-1	10	90	100
	Group-2	80	20	100
	Total	100	100	200

It suggests that the best assignment of actual to predicted is Actual-Group-

1 to Predicted-Group-2, i.e, 90 and the Actual-Group-2 to Predicted-Group-1, i.e, 80. So out of 200 points, $90 + 80 = 170$ points are rightly aligned and 30 points are wrongly aligned. After the Kuhn-Munkres algorithm aligns the two partitions, the sum of the off diagonal entries in the confusion matrix is computed as $SUMOFF$. Then the KM-index is computed as

$$KM - index = \frac{SUMOFF}{n} * 100\%. \tag{2.9}$$

Thus for Table 2.5, the KM-index is 15%.

2.4.2 Effect of α and β

This section examines the effect of α and β on the results. Observe that the value of α and β cannot be chosen randomly because higher values of α will make the two partitions closer, while β controls crispness of the weights. If we want to select either a fixed % of features or a fixed number of features given a data set, then there may exist an optimal pair of α and β, which can be found using training-validation paradigm. In the future studies, such possibilities of finding the most desirable pair(s) of α and β will be explored.

For this, two datasets have been considered. One is a new synthetic dataset and the other is the Iris dataset. The synthetic dataset contains 3 clusters with 50 points each having centres at (1,1,1), (5,5,5) and (7,7,7) respectively. The 4^{th} and the 5^{th} features are random values from -5 to 5.

First, a fixed value of β is chosen while α is varied. In each run, the difference between M and U is analyzed, i.e., how similar the updated partition matrix is with the FCM generated partition matrix, say we call it "Similarity" and it is defined as in Equation 2.10.

$$similarity = \frac{1}{n} \sum_{i} \sum_{j} |u_{ij} - \mu_{ij}| \tag{2.10}$$

Table 2.6: Effect of α for a fixed value of $\beta = 100$.

	Similarity	
α	Synthetic data	Iris data
0	0.34	0.14
100	0.20	0.079
500	0.09	0.034
1000	0.06	0.021
5000	0.02	0.868
10000	1.2	0.0057
15000	0.008	0.002
20000	0.611	0.0015
25000	0.0004	0.0012
30000	0.003	0.0006

In Equation (2.10), the division by n ensures that the index remains independent of n.

An ablation study is conducted first. An ablation study removing the regularizer ($\alpha = 0$) on the partition matrix will reduce it to a problem of entropy-regularized feature selection. It may help us to figure out how similar are the generated partition to the target partition. Note that depending on the dataset, the set of selected features may be the same with or without the regularizer involving the partition matrix. It is noted that the similarity at $\alpha = 0$ is 0.34 and 0.14 for synthetic and Iris data respectively. From Table 2.6, it is seen that as α increases the model emphasizes more on the 2^{nd} term in Equation (2.3). Thus, more similar partition matrices are obtained, as generated by FCM, during the feature selection process. It is useful when we have a targeted structure in the dataset and maintaining the same structure if someone wants to select the features. It is worth mentioning that this technique can be extended to a classification problem as well.

Now, the effect of β is considered while keeping $\alpha = 25000$ fixed. For this, the entropy of the weight distribution is computed. The results are shown in Table 2.7.

Table 2.7: Effect of β for a fixed value of $\alpha = 25000$.

	Entropy	
β	Synthetic data	Iris data
10	0.92	0.62
20	1.06	0.86
30	1.14	1.02
50	1.30	1.19
70	1.40	1.27
100	1.48	1.90
200	1.57	1.37
500	1.60	1.38
1000	1.60	1.38
3000	1.60	1.38

In the proposed objective function as shown in Equation (2.3), the parameter β controls the distribution of weights among the features. A more uniform distribution of features (higher entropy) is obtained as β increases. For each set of feature w_j where $j = 1, \cdots, p$, the entropy is measured as $-\sum_{j=1}^{p} w_j \ln(w_j)$. For uniform distribution the entropy is high and for sparse weights the entropy is low. In Table 2.7, it is observed that for lower values of β, the weights are more sparse compared to those for higher β. So, a higher α and a lower β are suggested. Here, $\alpha = 25000$ and $\beta = 100$ are chosen based on the results in Table 2.6 and Table 2.7.

2.4.3 Studies on Real datasets

In this section, seven real datasets (Table 2.1) [132] are used to evaluate the performance of the proposed model. Before running the algorithm all the real datasets are z-score normalized. As stated earlier, the FCM-generated partition matrix is taken as the target, referred to as M in the algorithm. The model assigns weights to each feature. First, the top two features with the highest weights are selected, and the remaining features are subsequently added in order of their weights to evaluate NMI, ARI, and misalignment in terms of the KM index. For a large number of features, the process begins with 10% of the features and subsequently increases the percentage.

The Iris dataset contains four features. The proposed model selects feature-4 and feature-3 as top two features. The NMI and ARI are 0.71 and 0.69 respectively. The mis-alignment is 12.67. It is evident that the model selected the best two features and the mis-alignment is reasonably good. It is well known that in Iris data classes two and three have some overlap and class one is well separated. As the number of features increases, better results are observed in terms of NMI and ARI. It is worth mentioning here that the focus is not on classifying the Iris data; rather, the aim is to identify the natural groups as predicted by FCM. The results with the Iris dataset are shown in Table 2.8.

Table 2.8: Mean of NMI, ARI and KM-index (Standard Deviation of NMI, ARI & KM-index) for Iris dataset. The reported values are for 5 runs.

Features	NMI(SD)	ARI(SD)	KM-index(SD)
4,3	0.71 (0.0)	0.69 (0.0)	12.67 (0.0)
1,4,3	0.80 (0.0)	0.82 (0.0)	6.67 (0.0)
2,1,4,3	1.0 (0.0)	1.0 (0.0)	0.0 (0.0)

Table 2.9: Mean of NMI, ARI and KM-index (Standard Deviation of NMI, ARI & KM-index) for Ecoli dataset. The reported values are for 5 runs.

Features	NMI(SD)	ARI(SD)	KM-index(SD)
7,6	0.40(0.0)	0.23(0.01)	61.67(1.75)
1,7,6	0.56(0.01)	0.38(0.04)	46.43(2.38)
2,1,7,6	0.67(0.01)	0.53(0.05)	32.92(2.33)
4,2,1,7,6	0.67(0.01)	0.52(0.05)	32.56(2.8)
3,4,2,1,7,6	0.66(0.01)	0.51(0.04)	33.69(2.26)
5,3,4,2,1,7,6	0.83(0.09)	0.77(0.11)	16.19(8.10)

Next, the Ecoli dataset is considered and the results are shown in Table 2.9. The data are represented by seven features and there are eight classes. The analysis started with the top two features. As the dataset is not well separable, the NMI and ARI are low. Approximately 62% of data are mis-aligned when the top two features are selected. As the number of selected features increases, over all indexes improve. In Vertebral data (Table 2.10), the mis-alignment for the top two features is 14.52% and subsequently reduced except for top four features. In case of Wine data (Table 2.11), we observe that the mis-alignmet for top two features is 16.29% and the NMI and ARI for are 0.6 and 0.58, respectively.

Similar experiments are conducted on the WBC(D) (Table 2.12), Sonar (Table 2.13), and Musk (Table 2.14) datasets. Note that, taking % of features, the rounded value is considered. The mis-alignment for top 10% features for WBC(D), Sonar and Musk are 3.51%, 25.48% and 5.04% respectively. For WBC(D) the NMI and ARI for top 10% features are 0.78 and 0.86 respectively. For Musk, it is 0.75 and 0.81 respectively which is reasonably good. In case of Sonar, for top 10% features, the NMI and ARI are 0.19 and 0.24 respectively which improves further with the features.

Table 2.10: Mean of NMI, ARI and KM-index (Standard Deviation of NMI, ARI & KM-index) for Vertebral dataset. The reported values are for 5 runs.

Features	NMI(SD)	ARI(SD)	KM-index(SD)
3,1	0.60(0.0)	0.63(0.0)	14.52(0.0)
6,3,1	0.70(0.0)	0.76(0.0)	9.03(0.0)
4,6,3,1	0.67(0.0)	0.72(0.0)	10.32(0.0)
2,4,6,3,1	0.81(0.0)	0.85(0.0)	4.84(0.0)
5,2,4,6,3,1	1.0(0.0)	1.0(0.0)	0.0(0.0)

It is re-emphasized that FCM identifies the natural groupings in the data. One can use any such partition matrix to find the corresponding selected features. We have shown the same idea using seven real and three synthetic datasets. It is shown that if the data contain clusters in a subspace and that is captured by FCM then our model can find that with less (the appropriate) number of features.

Table 2.11: Mean of NMI, ARI and KM-index (Standard Deviation of NMI, ARI & KM-index) for Wine dataset. The reported values are for 5 runs.

Features	NMI(SD)	ARI(SD)	KM-index(SD)
12,7	0.6(0.0)	0.58(0.0)	16.29(0.0)
6,12,7	0.5(0.0)	0.48(0.0)	21.91(0.0)
13,6,12,7	0.71(0.0)	0.71(0.0)	11.24(0.0)
11,13,6,12,7	0.82(0.0)	0.85(0.0)	5.06(0.0)
1,11,13,6,12,7	0.79(0.0)	0.82(0.0)	6.18(0.0)
9,1,11,13,6,12,7	0.78(0.0)	0.81(0.0)	6.74(0.0)
10,9,1,11,13,6,12,7	0.82(0.0)	0.85(0.0)	5.06(0.0)
2,10,9,1,11,13,6,12,7	0.94(0.0)	0.95(0.0)	1.69(0.0)
8,2,10,9,1,11,13,6,12,7	0.89(0.0)	0.91(0.0)	2.81(0.0)
4,8,2,10,9,1,11,13,6,12,7	0.89(0.0)	0.91(0.0)	2.81(0.0)
5,4,8,2,10,9,1,11,13,6,12,7	0.88(0.0)	0.90(0.0)	3.37(0.0)
3,5,4,8,2,10,9,1,11,13,6,12,7	1.0(0.0)	1.0(0.0)	0.00(0.0)

2.5 Conclusion

As per our knowledge, all the unsupervised feature selection models, specially, using extended version of the FCM framework looks for a feature subset based on some penalty/regularizer [124, 128]. We could not find any study where FCM provided partition matrix or any other target partition matrix is used to select the features. Our idea is to select features based on some targeted structure generated by any clustering algorithm. As a natural choice, we have formulated a model to select feature that can maintain the cluster structure found by FCM in the original data space to the cluster structure in the lower dimensional feature space. In place of FCM partition, any other partition matrix generated by any other clustering algorithm also can be used for feature selection. In our study, we have seen that if the data has good separability between clusters/groups, and the actual clusters lie in a lower dimensional subspace our model can find this.

In support of our claim, we have studied the performance on three synthetic datasets and seven real datasets. In the synthetic datasets, we have

Table 2.12: Mean of NMI, ARI and KM-index (Standard Deviation of NMI, ARI & KM-index) for WBC(D) dataset. The reported values are for 5 runs.

Features	NMI(SD)	ARI(SD)	KM-index(SD)
Top 10%	0.78(0.0)	0.86(0.0)	3.51(0.0)
Top 20%	0.78(0.0)	0.87(0.0)	3.34(0.0)
Top 30%	0.76(0.0)	0.85(0.0)	3.87(0.0)
Top 40%	0.69(0.0)	0.78(0.0)	5.62(0.0)
Top 50%	0.66(0.0)	0.75(0.0)	6.5(0.0)
Top 60%	0.75(0.0)	0.83(0.0)	4.39(0.0)
Top 70%	0.84(0.0)	0.90(0.0)	2.46(0.0)
Top 80%	0.91(0.0)	0.95(0.0)	1.23(0.0)
Top 90%	0.96(0.0)	0.98(0.0)	0.53(0.0)
100%	1.0(0.0)	1.0(0.0)	0.00(0.0)

used well separated clusters in a subspace and augmented the datasets with several random features, and as expected, our model selects the right features and discarded the random ones. For further study, we have taken seven real datasets where the number of features ranging from 4 to 166. We have demonstrated the model performance in terms of NMI, ARI and KM-index. We emphasize that our focus on mis-alignment was more than that on NMI and ARI. Except for the Ecoli and Sonar data, the mis-alignments are reasonably good. In our study, we put emphasis on the data structure while finding the features and the partition matrix plays a pivotal role to do that.

As discussed earlier, there are many choices for the regularizer for feature selection under a unsupervised framework. However, none of those pays any attention to the partition present in the original high-dimensional data. In absence of any additional information (such as availability of cluster labels of some data points) about the clusters to be extracted, the most natural goal of feature selection would be to find a subset of features that can find the same (or almost the same) partition matrix that are present in the original high-dimensional data (we call this partition as the target partition matrix). We also want the regularizer to be flex-

Table 2.13: Mean of NMI, ARI and KM-index (Standard Deviation of NMI, ARI & KM-index) for SONAR dataset. The reported values are for 5 runs.

Features	NMI(SD)	ARI(SD)	KM-index(SD)
Top 10%	0.19(0.0)	0.24(0.0)	25.48(0.0)
Top 20%	0.31(0.0)	0.39(0.0)	18.75(0.0)
Top 30%	0.35(0.0)	0.44(0.0)	16.83(0.0)
Top 40%	0.43(0.0)	0.53(0.0)	13.46(0.0)
Top 50%	0.51(0.0)	0.62(0.0)	10.58(0.0)
Top 60%	0.57(0.0)	0.68(0.0)	8.65(0.0)
Top 70%	0.63(0.0)	0.73(0.0)	7.21(0.0)
Top 80%	0.73(0.0)	0.82(0.0)	4.81(0.0)
Top 90%	0.8(0.0)	0.87(0.0)	3.37(0.0)
100%	1.0(0.0)	1.0(0.0)	0.00(0.0)

ible enough to accommodate the situation when cluster labels of a few points are available (semi-supervised clustering) or even when the entire target partition matrix is generated by a different clustering algorithm, or the target partition is obtained from a classification dataset (class label information). In the last case the problem of interest is to find a subset of features that can maintain the original class structure, i.e., effectively it becomes a supervised feature selection problem. In summary, we want a regularizing framework that can deal with unsupervised feature selection to preserve the same cluster structure as that of the original high-dimensional data, can handle cluster labels of a few data points when available (semi-supervised feature selection) as well as supervised feature selection. To the best of our knowledge, none of the methods in the literature is general enough to deal with such a variety of situations, but the proposed method can.

The unique characteristic of the proposed method is that it tries to select features that can maintain the cluster structure present in the original high dimensional data. This is significantly different from other unsupervised feature selection using the FCM framework. In this study, we have selected only one set of features for all clusters. However, there may

Table 2.14: Mean of NMI, ARI and KM-index (Standard Deviation of NMI, ARI & KM-index) for MUSK dataset. The reported values are for 5 runs.

Features	NMI(SD)	ARI(SD)	KM-index(SD)
Top 10%	0.75(0.0)	0.81(0.0)	5.04(0.0)
Top 20%	0.76(0.0)	0.81(0.0)	4.83(0.0)
Top 30%	0.76(0.0)	0.81(0.0)	4.83(0.0)
Top 40%	0.78(0.0)	0.84(0.0)	4.2(0.0)
Top 50%	0.86(0.0)	0.91(0.0)	2.31(0.0)
Top 60%	0.88(0.0)	0.93(0.0)	1.89(0.0)
Top 70%	0.9(0.0)	0.94(0.0)	1.47(0.0)
Top 80%	0.91(0.0)	0.95(0.0)	1.26(0.0)
Top 90%	0.94(0.0)	0.97(0.0)	0.84(0.0)
100%	1(0.0)	1(0.0)	0.00(0.0)

be applications where different clusters may lie in different subspaces. In our future study, we would extend our model to deal with such cases. we also plan to study the effectiveness of the proposed framework using the class labels, i.e., using the supervised mode.

Chapter 3

Is it Rational to Partition a Data Set using Kernel-Clustering?

3.1 Introduction

In Chapter 2, a feature selection technique has been explored, aiming for a regularizing framework that can handle unsupervised feature selection while preserving the cluster structure of the original high-dimensional data. The regularizer is designed to be flexible enough to accommodate scenarios where cluster labels for a few points are available (semi-supervised clustering), the entire target partition matrix is generated by a different clustering algorithm, or the target partition is derived from a classification dataset (class label information). After feature selection, in unsupervised domain, next task is to find the natural groups in the data, i.e to find the clusters. In this chapter and in the next chapter (Chapter 4), the issue of kernel clustering has been addressed. The rationality behind such clustering approaches is questioned. Using simple datasets, it is argued and demonstrated that finding clusters in the kernel space is not a good idea when the objective is to identify clusters in the original data. A cluster seeking algorithm partitions a given dataset $X = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$, $\mathbf{x}_i \in \mathbb{R}^p$ into c number of clusters so that members of the same cluster are similar to one another and members of different cluster are dissimilar.

Clustering has many applications covering almost all areas of science and technology [136, 137] and hence there are many algorithms [31, 138–140] and the research efforts are still going on to develop more effective algorithms. Some algorithms use probabilistic concepts, while others use fuzzy and/or possibilistic modeling [31, 139, 141].

Crisp clustering methods assume that each data point belongs to one and only one cluster, which may not be true in some real life applications such as land-cover classification. In such cases, fuzzy clustering is considered a better alternative for extracting sub-structures in the data. In a fuzzy cluster, a data point may partially belong to more than one cluster and the membership degrees/grades lie in [0,1]. The most popular fuzzy clustering algorithm is the Fuzzy C-Means (FCM) [31] which works fine when the dataset has hyperspherical clusters. However, if a pair of clusters is not linearly separable (eg., two nested spherical-shells), such algorithms usually fail. To identify such clusters there have been many extentions/variants of the FCM algorithm [129, 142–144]. In the recent past there have been many attempts to kernelize clustering algorithms. These algorithms are demonstrated to be useful in clustering data such as a disk surrounded by a shell, which are otherwise difficult to cluster [67, 72, 73, 140, 145–147]. These algorithms project the data implicitly into a high dimensional space with a hope that the "clusters" in the original space (we shall call it feature space) will be well separated in the kernel space. These algorithms then find the clusters in the kernel space.

There are different types of kernelized version of FCM. We shall consider a version which assumes that the cluster centers/prototypes are in the Kernel space [148–151]. This family of algorithms requires an inverse mapping of prototypes from the kernel space to feature space. This type of algorithms will be called as KFCM-K (Kernel FCM with prototypes in the kernel space).

The objective of any clustering algorithm is to find the clusters (i.e., natural groups) in the *original* data space. Does a kernel clustering algorithm do so? Sometimes 2-Dimensional datasets are used to demonstrate the success of a kernel clustering algorithm. Usually some "suitable" kernel parameter values are chosen and results are demonstrated. Do we know if the algorithm will work for a wide range of choices of parameters? It will be shown that, even for some simple datasets, the answer to the previous question is "No". When the datasets are in 2-D we can visualize easily the performance. But when the data are in high dimension visual assessment is difficult. For such cases, usually the cluster centroids or the partition are used in conjunction with a labeled dataset (the dataset is for a classification problem) to assess the resubstitution or classification error. This approach makes an assumption that number of natural clusters is equal to the number of classes. But this may not necessarily be true as clustering and classification are philosophically two different problems. For example, a two class problem may have four distinct clusters and we may not know it (A high dimensional generalization of the Ex-OR problem). If the resubstitution or classification error is low, we should be happy, but if not, we cannot make any inference because the number of clusters may be equal to the number of classes but the clustering could be poor or the number of clusters could be more than the number of classes and that may lead to poor classification performance. So assuming number of classes equal to the number of clusters and then checking the resubstitution error or classification error is not a good way of justifying kernel clustering.

Here, the investigation focuses on whether clustering in the kernel space is justified. The findings suggest that kernel clustering can lead to strange (counterintuitive) results if the number of clusters and/or the kernel parameters are not appropriately chosen. This issue is discussed and demonstrated using some simple datasets.

3.2 Kernelized Version of FCM

A dataset may have clusters in it, but it may be difficult for an algorithm like FCM to extract those clusters. One reasons for this may be that a pair of clusters is not linearly separable. In such a case, the original data can be mapped to a higher dimensional space (kernel space) with a hope that clusters will be easily identifiable [149–151]. For this a non linear mapping function f is used such that $\phi : \mathbb{R}^p \longrightarrow H$, where H is the kernel space. Typically kernel-clustering algorithms use the kernel trick that takes the advantage of the fact that the dot products in the kernel space can be represented by a Mercer kernel K where $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y})$, where, $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$. Thus the distance/similarity in the kernel space can be replaced by the Mercer kernel. Some of the popular kernel functions are:

(a) RBF kernel : $e^{-||\mathbf{x}-\mathbf{y}||^2/\sigma^2}, \sigma^2 > 0$

(b) Polynomial kernel : $(\mathbf{x^T}.\mathbf{y} + a)^b, a \geq 0, b \in N$

(c) Hyper-tangent : $tanh(\mathbf{x^T}.\mathbf{y} + a), a \geq 0$

In this study, only one kernel-based FCM algorithm, KFCM-K, is considered, using the Polynomial kernel (PK). Other versions, such as KFCM-F [140,148,152], where cluster centers are considered in the feature space itself, are not included.

3.2.1 Kernel Fuzzy C-means with prototypes in kernel space (KFCM-K)

Here the prototypes are located in the kernel space, so the pre-images of prototypes in the feature space have to be computed by taking the inverse mapping from kernel space to feature space [67,72,73,147,148]. KFCM-K algorithm minimizes the following objective function at (3.1).

$$Q = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} ||\phi(\mathbf{x_k}) - \mathbf{v_i}||^2 \tag{3.1}$$

subject to the constraints

(a) $u_{ik} \in [0, 1] \forall i, k$

(b) $0 < \sum_{k=1}^{n} u_{ik} < n \; \forall i$

(c) $\sum_{i=1}^{c} u_{ik} = 1 \; \forall k$

In (3.1) $\mathbf{v}_i$ is ith prototype in the kernel space. Using the Lagrange Multiplier technique, optimization of Q with respect to $\mathbf{v}_i$, results in

$$\mathbf{v_i} = \frac{\sum_{k=1}^{n} u_{ik}^{m} \phi(\mathbf{x_k})}{\sum_{k=1}^{n} u_{ik}^{m}} \tag{3.2}$$

Similarly, the expression for the membership values are derived as

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{||\phi(\mathbf{x_k}) - \mathbf{v_i}||}{||\phi(\mathbf{x_k}) - \mathbf{v_j}||}\right)^{\frac{2}{m-1}}} ; i = 1, \cdots, c; k = 1, \cdots, n \tag{3.3}$$

where

$$\begin{aligned} ||\phi(\mathbf{x_k}) - \mathbf{v_i}||^2 &= (\phi(\mathbf{x_k}) - \mathbf{v_i})^T (\phi(\mathbf{x_k}) - \mathbf{v_i}) \\ &= \phi(\mathbf{x_k})^T \phi(\mathbf{x_k}) - 2\phi(\mathbf{x_k})^T \mathbf{v_i} + \mathbf{v_i}^T \mathbf{v_i}. \end{aligned} \tag{3.4}$$

Using the expression for the prototypes (3.2) into the expression (3.4) , one gets

$$\begin{aligned} ||\phi(\mathbf{x_k}) - \mathbf{v_i}||^2 &= K(\mathbf{x_k}, \mathbf{x_k}) - 2\frac{\sum_{j=1}^{n} u_{ij}^{m} K(\mathbf{x_k}, \mathbf{x_j})}{\sum_{j=1}^{n} u_{ij}^{m}} \\ &\quad + \frac{\sum_{j=1}^{n} \sum_{l=1}^{n} u_{ij}^{m} u_{il}^{m} K(\mathbf{x_k}, \mathbf{x_l})}{(\sum_{j=1}^{n} u_{ij}^{m})^2} \end{aligned} \tag{3.5}$$

Following, [147, 148], the expression for the prototypes in the feature space $V = \{\tilde{\mathbf{v}}_i; i = 1, 2, ..., c\}$ with Polynomial kernel is given in (3.6).

$$\tilde{\mathbf{v}}_i = \frac{\sum_{k=1}^{n} u_{ik}^{m} (\mathbf{x_k^T} \tilde{\mathbf{v}}_i + a)^{b-1} \mathbf{x_k}}{(\tilde{\mathbf{v}}_i{}^T \tilde{\mathbf{v}}_i + a)^{b-1} \sum_{j=1}^{n} u_{ij}^{m}} \tag{3.6}$$

The corresponding algorithm of KFCM-K is shown in Algorithm 3.1.

Note that, there are many other equivalent termination criteria that can be used in place of (3.7) and (3.8).

Algorithm 3.1 KFCM-K

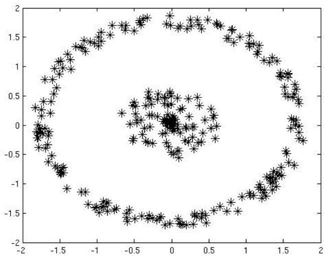
```
(a) Fix $c, t_{max}, m > 1$ and $\epsilon > 0$ , the tolerance level for termination.
(b) Initialize membership matrix $U$.
(c) Repeat
        Update $U = [u_{ik}]$ according to equation (3.3) via (3.5).
    until $\sum_{i=1}^{c}\sum_{k=1}^{n} |u_{ik}^{(t+1)} - u_{ik}^{(t)}| < \epsilon$ is satisfied.   (3.7)
(d) Return $U$.
(e) Initialize prototypes $V$
(f) Repeat
        Update $\tilde{\mathbf{v}}_i$s according to equation (4.6) for Polynomial kernel
    until $\sum_{i=1}^{c}\sum_{k=1}^{n} |\tilde{v}_{k,i}^{(t+1)} - \tilde{v}_{k,i}^{(t)}| < \epsilon$ is satisfied.   (3.8)
(g) Return $V$.
```

3.3 Results

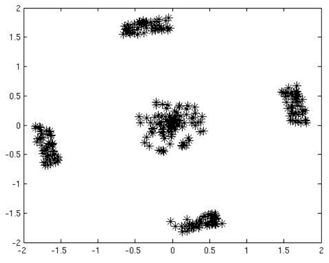
Three synthetic datasets are used in this study, all in R^2:Ring, Five-Cluster, and Uniform-Disc.

- **Ring** : The Ring Dataset is depicted in Figure 3.1a, where the inner disk contains 100 points and the outer ring contains 200 points.
- **Five-Cluster** : Figure 3.1b shows the Five-Cluster dataset where the inner disk, like Ring contains 100 data points forming a cluster. Surrounding this cluster, there are four smaller clusters, each with 50 points.
- **Uniformdisk** : Figure 3.1c depicts the Uniform dataset with 441 points that are uniformly distributed. This dataset does not have any cluster in it.

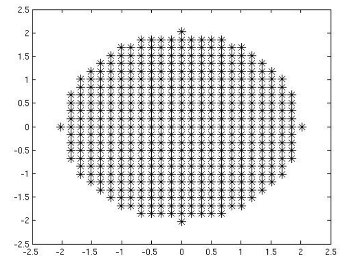
For each dataset, X, every feature x is normalized by $\frac{(x-\mu)}{\sigma}$, where μ and σ are the mean and standard deviation of x as estimated from the given dataset. This protocol is followed in [148].



(a) Ring dataset



(b) FiveCluster dataset



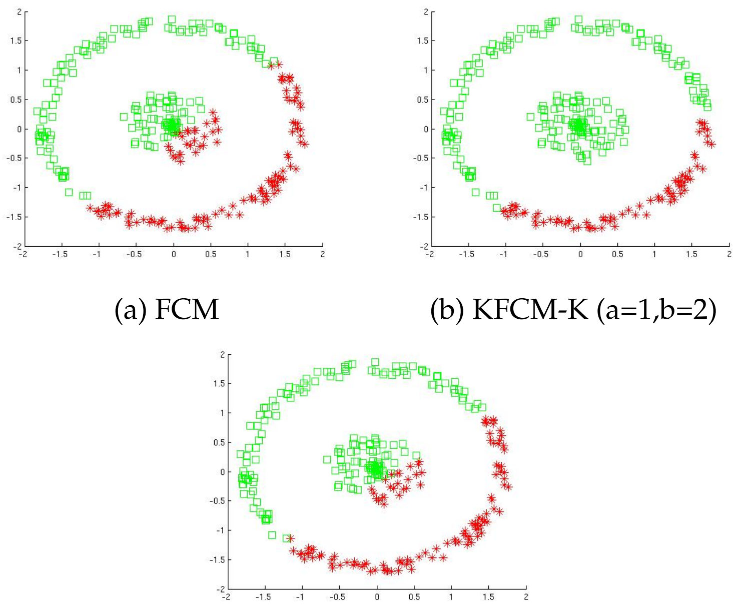
(c) Uniform dataset

Figure 3.1: The three Synthetic datasets used in this study

3.3.1 Kernel Clustering with Polynomial Kernel

The behavior of FCM and KFCM-K with the Polynomial kernel is studied on three datasets: Ring, Five-Cluster, and Uniform. The parameters used are $m = 1.2, \epsilon = 10^{-8}$, and $t_{\max} = 100$. For the Polynomial kernel, experiments are conducted with different values for $a \in \{1, 2, 3, 4, 5, 7, 10, 15\}$ and $b \in \{2, 4, 6, 8, 10, 12, 14, 16\}$, resulting in a total of 64 combinations. The initial prototypes are taken randomly from the input dataset.

First, the Ring dataset is considered, Figure 3.1a. Clearly there is a disk-shaped cluster at the center. Many authors share the view that the shell around the disk is also a cluster. The same view is shared here as well. For this dataset with $c = 2$, FCM, as expected, divides the dataset as



(a) FCM

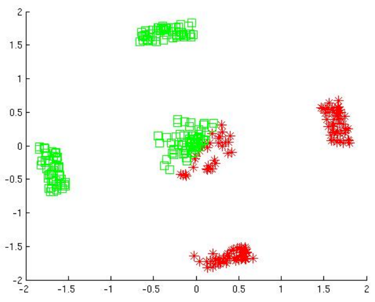
(b) KFCM-K (a=1,b=2)

(c) KFCM-K (a=15,b=2)

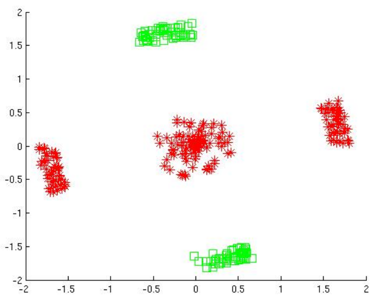
Figure 3.2: Clustering of Ring data with FCM and KFCM-K for c=2

shown in Figure 3.2a. For some choices of a and b, the KFCM-K behaves similar to FCM, but for none of the 64 choices of parameters, KFCM-K could extract the shell as a separate cluster. Sometimes, part of the shell is extracted as a cluster and the remaining data points as another cluster. One such partition is exhibited in Figure 3.2b. It is also noticed that when a is high and b is low, KFCM-K behaves similarly to FCM for this dataset. Figure 3.2c shows one such partition with $a = 15$ and $b = 2$. For this dataset we find that the choice of a and b has a significant influence on the clustering output. Moreover, in this case, KFCM-K failed to produce the desired partition. Therefore, no further analysis is conducted on this dataset.

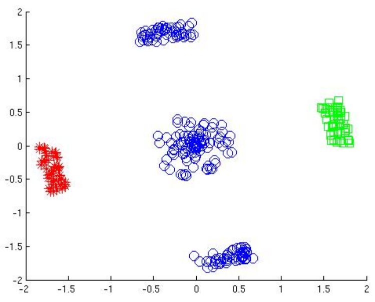
Next, the focus shifts to the Five-Cluster dataset (Figure 3.1b), which contains five distinct clusters. As expected, FCM finds all the five clusters correctly when $c = 5$ (Figure 3.3g), so does KFCM-K with $a = 1, b = 2$ (Figure 3.3h). Figure 3.3, also compares the results with $c = 2, 3,$ and 4. For example, when $c = 4$ (with $a = 1, b = 2$), the FCM result is quite



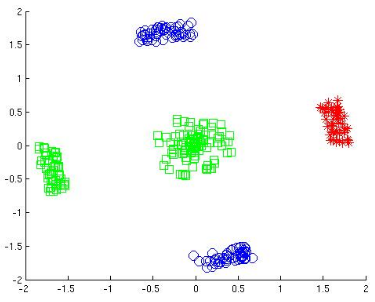
(a) FCM



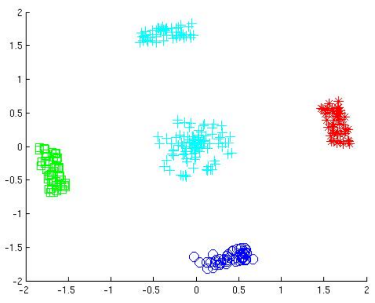
(b) KFCM-K



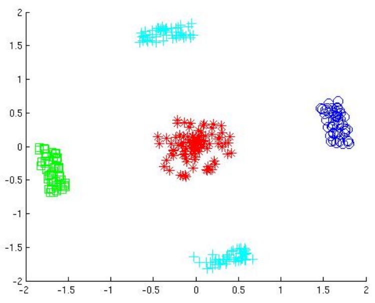
(c) FCM



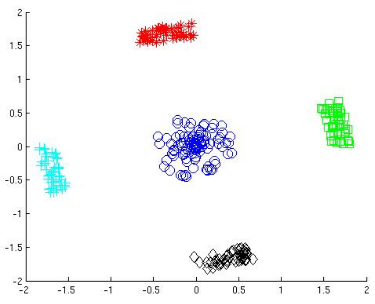
(d) KFCM-K



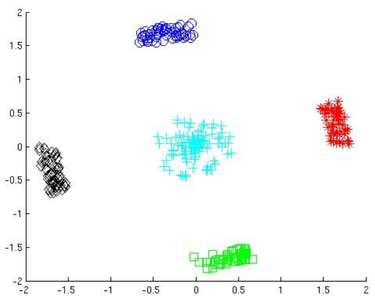
(e) FCM



(f) KFCM-K

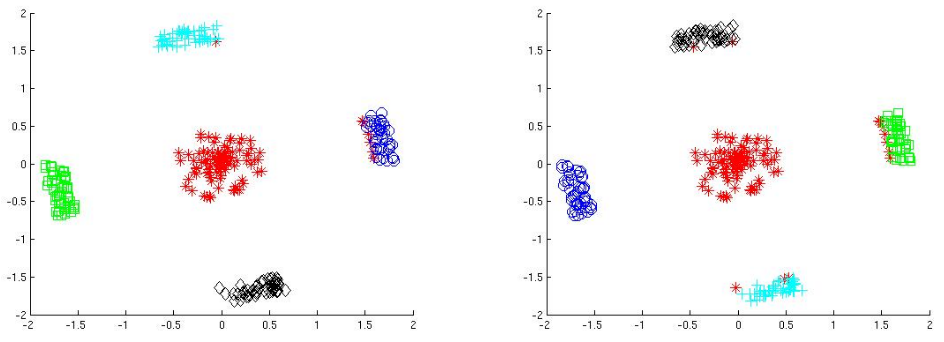


(g) FCM



(h) KFCM-K

Figure 3.3: (a) FCM Clusters with $c = 2$;
(b)KFCM-K Clusters with $c = 2$;
(c) FCM Clusters with $c = 3$; (d)KFCM-K Clusters with $c = 3$;
(e) FCM Clusters with $c = 4$; (f)KFCM-K Clusters with $c = 4$;
(g) FCM Clusters with $c = 5$; (h)KFCM-K Clusters with $c = 5$;



(a) $a = 1, b = 12$ (b) $a = 2, b = 16$

Figure 3.4: Few bad clusters on the Five-Cluster dataset using KFCM-K.

intuitive (Figure3.3e), while KFCM-K merges two clusters, which are on the *opposite sides* of the central disk, into one clusters (Figure 3.3f). Thus KFCM-K merges two small clusters when they are quite far from each other. It is noted that while running clustering algorithms with different initializations similar results are obtained across several runs. However, for $c = 2$ and $c = 3$, FCM produces intuitively acceptable results (Figure 3.3a and Figure 3.3c), while KFCM-K does not (Figure 3.3b and Figure 3.3d) as it merges two clusters which are not nearby. In this case, since the data are in 2-D, the results can be assessed visually. However, for high-dimensional data, the desired number of clusters may not be known, which can lead to undesirable results, such as the merging of two distant clusters, without even realizing it. This is one facet of the problems associated with kernel clustering.

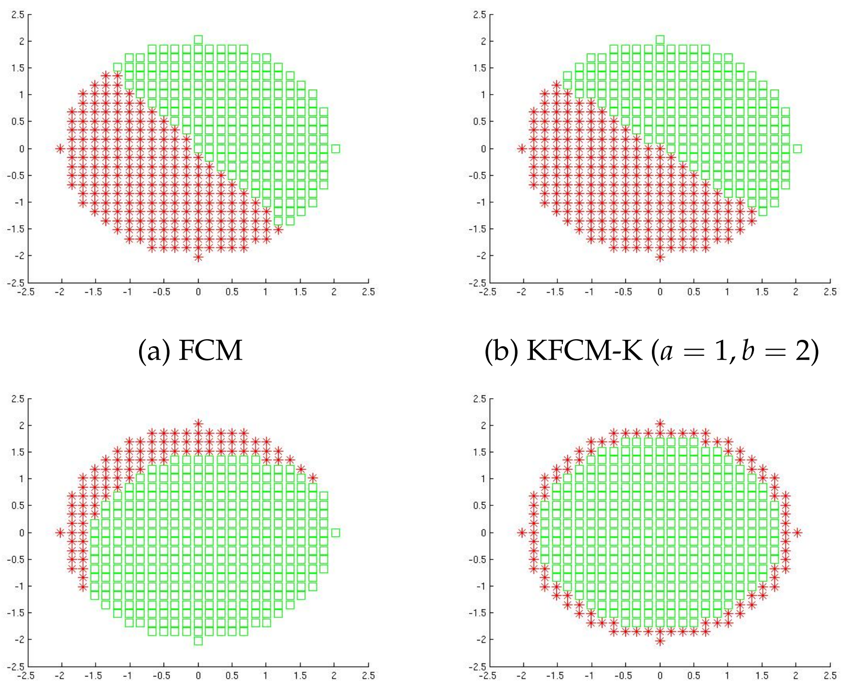
The other problems is associated with the choice of kernel parameters. To demonstrate this issue, consider $c = 5$, the best choice for number of clusters for this dataset. Next, KFCM-K is run for all 64 combinations of a and b as mentioned earlier. Table 3.1, summarizes the results. Table 3.1 reveals that when $a = 1$, and b is high (12 or higher), KFCM-K fails to extract the correct partition. Figure 3.4 (a) reveals that with $a = 1$ and $b = 12$, a few points of the cyan (plus) and blue (circle) clusters get merged with the central disk. Similarly, with $a = 2$ and $b = 14$ or higher, the partition starts departing from the desirable one (Figure 3.4 (b)). Thus, it

Table 3.1: Value of Correlations for different kernel parameters [a,b-Kernel parameters, Q-Quality].

DATA	**a,b**	**Q**	**a,b**	**Q**	**a,b**	**Q**
FiveCluster	1,2	Good	5,2	Good	10,2	Good
	1,4	Good	5,4	Good	10,4	Good
	1,6	Good	5,6	Good	10,6	Good
	1,8	Good	5,8	Good	10,8	Good
	1,10	Good	5,10	Good	10,10	Good
	1,12	Bad	5,12	Good	10,12	Good
Uniform Disk	1,2	Good	5,2	Good	10,2	Good
	1,4	Medium	5,4	Good	10,4	Good
	1,6	Bad	5,6	Good	10,6	Good
	1,8	Bad	5,8	Medium	10,8	Good
	1,10	Bad	5,10	Bad	10,10	Good
	1,12	Bad	5,12	Bad	10,12	Medium

is found that the choice of kernel parameters is crucial to the success of the algorithm, even for very simple datasets. Since the sub-structures in high-dimensional data are unknown, it would be difficult to determine whether KFCM-K has found a good solution or not. One can make the same arguments with FCM, but the differences are : (a) with improper choices of c, the number of clusters, FCM splits or merges clusters in such a manner that they agree with our intuition and FCM does not impose strange geometric structures on the data and (b) FCM does not have these two extra parameters, which may change the geometry of the data.

Now the Uniform (Figure 3.1c) dataset is considered. Clearly Uniform does not have any sub-structure or cluster. Hence any clustering algorithm, with $c = 2$, should partition the data into approximately two halves, where the approximate line partitioning the data may have one of many possible orientations. Figure 3.5a shows that FCM does the same. Similarly, KFCM-K also does the same with $a = 1, b = 2$ (Figure 3.5b); as b increases, one of the clusters starts becoming bigger (Figure 3.5c). With $a = 1$, $b = 12$ or above, KFCM-K extracts a *nice shell* (Figure 3.5d) as one of the clusters. This behavior is also repeated with other choices of a. In fact, when a is increased, KFCM-K starts extracting a shell for further high values of b. This is a very serious limitation of the KFCM-K algorithm. As an illustration, Figure 3.6 shows the membership values of different points to the cluster they belong to. Figure 3.6 corresponds to Figure 3.5d. In Figure 3.6, for each point, the maximum membership value is determined. Based on its cluster label, the point is plotted either as a green(square) or a red (circle). Figure 3.6 shows that for most of the points in the central big cluster the membership values to the cluster are very high, while for most points belonging to the imposed outer-shell cluster, the membership values are above 0.6. There are some points in the bigger cluster with memberships varying from values slightly higher than 0.5 to further high values suggesting a kind of gradual transition from one cluster to the other. In this dataset, there is no reason to assume such a strange cluster structure.



(a) FCM (b) KFCM-K ($a = 1, b = 2$)

(c) KFCM-K ($a = 1, b = 6$) (d) KFCM-K ($a = 1, b = 12$)

Figure 3.5: Results on the Uniform dataset by FCM and KFCM-K for different parameters

Table 3.1 summarizes the visual assessment of quality of the partitions generated by kernel clustering for some illustrative sets of a and b. In Table 4.3, Medium indicates minor departure from ideal partition while Bad indicates significant departure from the ideal partition.

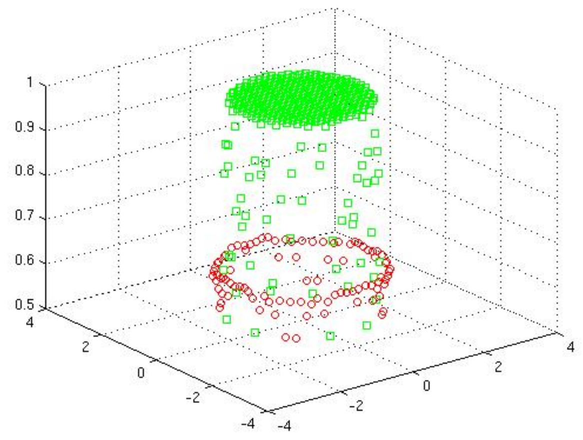


Figure 3.6: Pictorial representation of the maximum membership values to the clusters.

So what do we learn from these experiments?

- Knowledge of the number of clusters present in the data is essential for the success of KFCM-K; otherwise, some cluster may include data points which are far from each other. In particular when c is less than the actual number clusters, KFCM-K may merge two subclusters which are far from each other in the actual data (feature) space ignoring nearby clusters. Thus producing counterintuitive clusters.
- When a dataset does not have any cluster structure, depending on the choice of kernel-parameters, it can impose very artificial substructures (e.g., the KFCM-K extracts a shell from the uniform data). To summarize, KFCM-K may impose arbitrary structures on the data.
- Even when the number of clusters is known, appropriate choice of the kernel parameters is essential to get the desirable results.

These problems are quite serious because for any real life application, data are usually in a high dimensional space. But why do all these happen? This may happen because we are trying to extract clusters in a space which may alter the geometry of the original data. This does not mean that for some dataset KFCM-K type clustering cannot help. For example, if we have two touching clusters and in the projected space the two clusters become separated, then it would be easy to find the clusters in the kernel space. There could be other interesting cases where kernelized version could be useful. For example, using RBF kernel with an appropriate choice of the kernel width, it is possible to extract the shell and the central disk as two different clusters [148], but we need to use the right kernel with the right choice of parameters.

3.4 Conclusion

The performance of the kernel clustering algorithm, KFCM-K raises a very fundamental question : Should we really cluster a given data set in the kernel space? Although the data points may be more sparsely distributed in the kernel space, it cannot be assumed that the clusters identified in this space will correspond to the same groups as those in the original feature space. This has been illustrated using several synthetic datasets. In some of the studies involving kernel clustering the results are justified by matching the clusters with the classes [148]. In other words, one starts with a dataset on k classes where every point has an associated class label indicating the class that the data point comes from. Such a dataset is then clustered into $c = k$ clusters ignoring the class labels. Finally, one assesses how good the partition matches with class structure. If there is a good match we should be happy. But we need to remember that cluster structure in a dataset may be quite different from the structure imposed by class labels. The situation is actually more complicated because most kernel needs specification of some parameters. It has been demonstrated that the choice of kernel parameters significantly impacts the performance of the clustering algorithm. When the problem is of classification, we can use different ways, such as cross-validation, to choose a desirable value for the kernel parameters. But for cluster seeking problems, it is difficult to choose the optimal parameters because the problem is unsupervised.

It has been demonstrated that kernel clustering may lead to highly unexpected results. When the goal of clustering a dataset is to find the clusters / substructures in the original data (feature) space, unless one is careful, kernelization may impose undesirable structures on the data and hence the clusters obtained in the kernel space may not exhibit the structure of the original data. But, this does not mean that kernel clustering cannot be useful at all. If one wants to cluster in the kernel space, then proper choice of the parameters of the kernel function has to be used. There is no established mechanism for selecting the desirable parameters.

In the next chapter (Chapter 4), the merits and limitations of kernel-based clustering will be explained in detail.

Chapter 4

What and When can We Gain from the Kernel Versions of C-Means Algorithm?

4.1 Introduction

In Chapter 3, a fundamental question has been addressed regarding the efficacy of clustering within kernel spaces, with a particular focus on polynomial kernels. The study examined whether the clusters identified in the kernel space accurately corresponded to those in the original feature space. The chapter illuminated the intricate nature of kernel clustering, highlighting how the process of kernelization can introduce unintended structures onto the data, thereby yielding unexpected outcomes. In this chapter, the exploration of clustering within kernel space is extended, delving into philosophical inquiries regarding its usefulness and the circumstances under which it proves advantageous. This chapter emphasized the distinction between poor clustering outcomes resulting from issues such as local minimum problems or incorrect algorithmic parameters.

Cluster analysis has found many applications in different fields of science and engineering such as pattern recognition, image analysis, communication, and data mining [31, 136, 137, 139, 153, 154]. Clustering divides data into "homogeneous" groups (clusters) in order to improve our understanding about the data or to find sub-structures in the data. A clustering algorithm partitions a set of n input vectors $X = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$, $\mathbf{x}_i \subset \mathbb{R}^p$ into c clusters so that members of the same cluster are "similar" to each other and members of different clusters are dissimilar. Depending on the clustering algorithm, in addition to the partition, the c clusters may be represented by a set of c prototypes, $V = \{\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_c}\}, \mathbf{v}_i \subset R^p$.

Crisp clustering methods assume that each data vector belongs to one and only one cluster. This assumption is valid when data have well separated sub-groups. In real life applications, often it is difficult to find distinct boundary between clusters. Thus, fuzzy clustering is often considered a better alternative for extracting structures in data. In a fuzzy cluster, a data point may partially belong to more than one cluster and the membership degrees/grades lie in $[0, 1]$. The most popular fuzzy clustering algorithm is the Fuzzy C-Means (FCM) [31, 155, 156]. However, sometimes data may have "not-nice" clusters where a pair of clusters is not linearly separable. For example, if data contain a sphere surrounded by a shell, or contain two nested shells, often researchers want to call / identify the sphere and shell, or the two nested shells as two clusters. Such clusters usually cannot be found unless we specifically design algorithms to look for such structures [129, 142–144]. In the recent past there have been many attempts to kernelize clustering algorithms [67, 72, 73, 140, 145–147, 157]. These algorithms project the data into a high dimensional space with a hope that the clusters in the original space (we shall call it feature space) become well separated. These algorithms then find the clusters in the kernel space.

Two major categories of kernel based fuzzy clustering algorithms are available in the literature. The first category considers the prototypes in the feature space [140, 148, 152] and the other category assumes the prototypes in the kernel space [148–151]. Thus the later family of algorithms requires an inverse mapping of prototypes from the kernel space to feature space. These two types of algorithms will be called as KFCM-F (Kernel FCM with prototypes in the feature space) and KFCM-K (Kernel FCM with prototypes in the kernel space), respectively.

The kernel clustering algorithms raise a very fundamental question: should we really cluster a given dataset in the kernel space? The objective of any object-data clustering algorithm is to find the clusters (i.e., natural groups) in the *original* data space. Although in the kernel space the data points might be more sparsely distributed, we cannot say that the clusters that we can find in the kernel space will necessarily represent the same subgroups as those in the original feature space. In some studies involving kernel clustering the results are justified by matching the clusters with the classes [148]. In other words, we start with a dataset on k classes where every point has an associated class label indicating the class that the data point belongs to. Such a dataset is clustered into $c = k$ clusters, ignoring the class labels, and the quality of the partition is then assessed based on how well it aligns with the underlying class structure. Well, if there is a good match then we should be happy. It is important to remember that the cluster structure in a dataset may differ significantly from the structure imposed by class labels. A simple example is the exclusive-OR type two-class datasets. The situation is actually more complicated because most kernel needs specification of some parameters. For example, Gaussian/Radial Basis Function kernel needs the user to specify σ - the width of the kernel. It is well known from the literature on support vector machines that the choice of σ has a significant impact on the performance of classifiers. When the problem is of classification, we can use different ways, such as cross-validation, to choose a desirable value for σ. But for cluster seeking problems, it is difficult to choose the optimal parameters because the problem is unsupervised. It is emphasized that the parameter σ should not be viewed as a parameter of the clustering algorithm, like c and the initial cluster centers.

There are many cluster validity indices such as NMI (Normalized Mutual Information) [134], ARI (Adjusted Rand Index) [135,158], cluster instability [159], [160], NID (Normalized Information Distance) [161]. Use of cluster validity indices may not be useful to choose the optimal kernel parameters because if the transformation imposes well separated substructures (that were not present in the original data), a validity index may vote for such a partition. It will be demonstrated that indices such as NMI, ARI, and cluster instability are not particularly effective. In this context, Kleinberg's impossibility result is worth mentioning [162]. Kleinberg proposed a set of three axioms on scale invariance, consistency and richness for a clustering function and proved that there is no clustering algorithm that satisfies those three axioms. Ackerman [163] questioned Kleinberg's axioms and reformulated those axioms for cluster quality measures (CQM), which are consistent. Ackerman also added one more axiom (axiom of CQM).

The investigation focuses on the justification for using kernel clustering, demonstrating that it can lead to unusual results and may impose artificial cluster structures on the data, depending on the choice of kernel parameters. Sammon's nonlinear projection is then used to make a visual assessment of the data structure in the kernel space. This further demonstrates that kernelization may impose strange structure. The effect of kernel parameters on clustering a dataset in the kernel space has been discussed. Here the prime focus is on the RBF kernel. Some illustrative results on Hyper-tangent and polynomial kernels are also included. Detailed results of some early investigation using the polynomial kernel are reported in [86]. Our findings with polynomial and Hyper-tangent kernels are consistent with those of RBF kernel.

To reduce the effect of the choice of kernels, some authors have used multiple kernels [164,165]. For example, in [164], authors use a non-negative weighted combination of multiple kernels. Since different kernels use different nonlinear transformations, the resultant structure imposed by multiple kernels could still be strange and in fact, sometimes, it may even worsen the situation. However, for supervised learning, use of multiple kernels could be useful because one can choose the weights of different kernels as well as their parameters using the target values (or class label information) that are available.

4.2 Kernel based Fuzzy c-means

It is emphasized that the focus will be on clustering of object data, where each object is represented by a set of numerical features. If the clusters are not easily identifiable, (for example, if a pair of clusters is not linearly separable in the original p-dimensional feature space), then the data from the feature space may be mapped to a higher dimensional kernel space with a hope that it would be an easy job to find the clusters [149–151]. A non linear mapping function ϕ is used such that $\phi : \mathbb{R}^p \longrightarrow H$, where H is the kernel space. Typically for finding clusters, the kernel trick is employed by taking advantage of the fact that the dot product in the kernel space can be represented by a Mercer kernel K where $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y})$, where, $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$. Hence the distance/similarity in the kernel space can be replaced by the Mercer kernel. The Kernel matrix $K_{n \times n}$ can be viewed as a relation and relational clustering algorithms can also be developed on this [120].

Some of the popular kernel functions are :

(a) Gaussian (RBF) kernel : $e^{-||\mathbf{x}-\mathbf{y}||^2/\sigma^2}, \sigma^2 > 0$

(b) Polynomial kernel : $(\mathbf{x}^{\mathbf{T}}.\mathbf{y} + a)^b, a \geq 0, b \in N$

(c) Hyper-tangent : $tanh(\mathbf{x}^{\mathbf{T}}.\mathbf{y} + a), a \geq 0$

In this study, primarily we shall consider the RBF (Gaussian) kernel. As mentioned earlier, kernel based FCM algorithms are classified in two categories, Kernel Fuzzy C-means with prototype in feature space (KFCM-F) and Kernel Fuzzy C-means with prototype in kernel space (KFCM-K), depending on the position of the prototypes. Next, two such algorithms are described, one from each category.

4.2.1 Kernel FCM with prototype in the Feature Space

In this method [140, 148, 152] the following objective function is minimized :

$$Q = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} ||\phi(\mathbf{x_k}) - \phi(\mathbf{v_i})||^2, \tag{4.1}$$

subject to the constraints

(a) $u_{ik} \in [0,1] \forall i, k,$

(b) $0 < \sum_{k=1}^{n} u_{ik} < n \; \forall i,$

(c) $\sum_{i=1}^{c} u_{ik} = 1 \; \forall k.$

Here the prototypes are placed in the original feature space and are implicitly mapped in the kernel space through a kernel function.

Thus the square distance in the kernel space can be computed as

$$\begin{aligned} ||\phi(\mathbf{x_k}) - \phi(\mathbf{v_i})||^2 &= \phi(\mathbf{x_k})^T \phi(\mathbf{x_k}) + \phi(\mathbf{v_i})^T \phi(\mathbf{v_i}) \\ &\quad - 2\phi(\mathbf{x_k})^T \phi(\mathbf{v_i}) \\ &= K(\mathbf{x_k}, \mathbf{x_k}) + K(\mathbf{v_i}, \mathbf{v_i}) \\ &\quad - 2K(\mathbf{x_k}, \mathbf{v_i}). \end{aligned} \tag{4.2}$$

For the Gaussian kernel

$$||\phi(\mathbf{x_k}) - \phi(\mathbf{v_i})||^2 = 2(1 - K(\mathbf{x_k}, \mathbf{v_i})). \tag{4.3}$$

Using the Lagrange multiplier technique, the membership value, u_{ik} of the k-th data point to the i-th cluster can be obtained as:

$$u_{ik}^{(t+1)} = \frac{1}{\sum_{j=1}^{c} \left(\frac{1-K(\mathbf{x_k}, \mathbf{v_i}^{(t)})}{1-K(\mathbf{x_k}, \mathbf{v_j}^{(t)})} \right)^{\frac{1}{m-1}}}. \tag{4.4}$$

$i = 1, \cdots, c; k = 1, \cdots, n,$

where c is the number of clusters, $m > 1$ is the fuzzifier that controls the fuzziness in the partition, and t is the iteration number.

The expression of the prototypes for Gaussian kernel can be shown as

$$\mathbf{v_i^{(t+1)}} = \frac{\sum_{k=1}^{n} u_{ik}^m K(\mathbf{x_k}, \mathbf{v_i^{(t)}})\mathbf{x_k}}{\sum_{k=1}^{n} u_{ik}^m K(\mathbf{x_k}, \mathbf{v_i^{(t)}})}. \tag{4.5}$$

Algorithm 4.1 KFCM-F

```
(a) Fix $c, t_{max}, m > 1$ and $\epsilon > 0$, a tolerance level for termination.
(b) Initialize the set of prototypes, $V$
(c) For $t = 1, 2, ..., t_{max}$ do
        Update $U = [u_{ik}]_{c \times n}$ according to equation (4)
        Update $\mathbf{v}_i; i = 1, \cdots, c$ according to equation (5)
    until
        $$\sum_{i=1}^{c}\sum_{k=1}^{n} |u_{ik}^{(t+1)} - u_{ik}^{(t)}| < \epsilon \quad (4.6)$$
    is satisfied.
(d) Return U and V.
```

4.2.2 Kernel FCM with prototypes in the Kernel space

As discussed in chapte 3, in KFCM-K the prototypes are located in the kernel space.

KFCM-K algorithm [67,72,73,147,148] minimizes the following objective function:

$$Q = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^m ||\phi(\mathbf{x_k}) - \mathbf{v_i}||^2, \tag{4.7}$$

subject to the same set of constraints on the membership matrix as with the KFCM-F. Again using the Lagrange Multiplier technique, optimization of Q with respect to $\mathbf{v}_i$, results in

$$\mathbf{v_i} = \frac{\sum_{k=1}^{n} u_{ik}^m \phi(\mathbf{x_k})}{\sum_{k=1}^{n} u_{ik}^m}. \tag{4.8}$$

.

Similarly, the expression for the membership values are derived as

$$u_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{||\phi(\mathbf{x_k})-\mathbf{v_i}||}{||\phi(\mathbf{x_k})-\mathbf{v_j}||}\right)^{\frac{2}{m-1}}}. \tag{4.9}$$

$i = 1, \cdots, c; k = 1, \cdots, n$ where

$$\begin{aligned} ||\phi(\mathbf{x_k}) - \mathbf{v_i}||^2 &= (\phi(\mathbf{x_k}) - \mathbf{v_i})^T(\phi(\mathbf{x_k}) - \mathbf{v_i}) \\ &= \phi(\mathbf{x_k})^T\phi(\mathbf{x_k}) - 2\phi(\mathbf{x_k})^T\mathbf{v_i} \\ &\quad + \mathbf{v_i}^T\mathbf{v_i}. \end{aligned} \tag{4.10}$$

By putting the expression for the prototypes (4.8) into the expression (4.10) ,one gets

$$\begin{aligned} ||\phi(\mathbf{x_k}) - \mathbf{v_i}||^2 = \;& K(\mathbf{x_k}, \mathbf{x_k}) - 2\frac{\sum_{j=1}^{n} u_{ij}^m K(\mathbf{x_k}, \mathbf{x_j})}{\sum_{j=1}^{n} u_{ij}^m} \\ & + \frac{\sum_{j=1}^{n}\sum_{l=1}^{n} u_{ij}^m u_{il}^m K(\mathbf{x_k}, \mathbf{x_l})}{(\sum_{j=1}^{n} u_{ij}^m)^2}. \end{aligned} \tag{4.11}$$

Since the prototypes are in the kernel space, if we want to get their pre-images in the original feature space, then those have to be computed by taking an inverse mapping from kernel space to feature space. The expression for the prototypes, $V = \{\tilde{\mathbf{v}}_i; i = 1, 2, ..., c\}$ (in the input space), with Gaussian kernel is given as [147, 148]

$$\tilde{\mathbf{v}}_i^{t+1} = \frac{\sum_{k=1}^{n} u_{ik}^m K(\mathbf{x_k}, \tilde{\mathbf{v}}_i^t)\mathbf{x_k}}{\sum_{k=1}^{n} u_{ik}^m K(\mathbf{x_k}, \tilde{\mathbf{v}}_i^t)}. \tag{4.12}$$

Using the final membership matrix, these prototypes in the input (feature) space are computed using an iterative algorithm as explained below. Note that, to generate the partition matrix, we do not need to compute these prototypes. The algorithm of KFCM-K is discussed in Chapter 3.

Before presenting the results, some philosophical differences between kernel c-means (fuzzy/hard) clustering and other prototype-based clustering algorithms are highlighted. Typically, for a conventional (not a kernel version) clustering algorithm, the procedure is as follows:

(a) We are given a dataset (a set of objects, each of which is represented by a set of numerical features).

(b) The goal is to find clusters in the *given* data, i.e., to find sub-groups of “similar” objects in the given data.

(c) We have a conceptual understanding (model) of what we want to find as clusters. For example, hyper-spheres, shells, and lines, to name a few.

(d) The conceptual model of clusters usually dictates the similarity measure between objects as well as mathematical model(s) via optimization of some objective function(s). Some examples are : objective functions of FCM, Fuzzy c-Shell, Fuzzy c-varities [120]. The objective function typically involves a measure of (dis)similarity between objects or between prototypes and objects.

(e) The next step is to define an *algorithm* to optimize the objective function to extract the clusters.

    [Notes:] The algorithm as well as the objective function may have some parameters whose choice may influence the clustering results.

However in a kernel-c means type clustering, we project the data into a different space and the dot product of two projected data vectors defines the similarity between the two data points involved. The commonly used kernel functions have some parameters, whose choice may influence the similarity between points in the kernel space and hence may impose “artificial” cluster structure on the data. Thus, parameters of kernel functions and parameters of algorithms are of completely different nature. Improper choice of kernel parameter may impose structures that are non-existent in the original data, while improper choice of algorithm parameters may make the algorithm fail to find the desired clusters. But if the data structure is changed by some improper choice of kernel parameters, no choice of algorithmic parameters will be able to extract the clusters present in the original data.

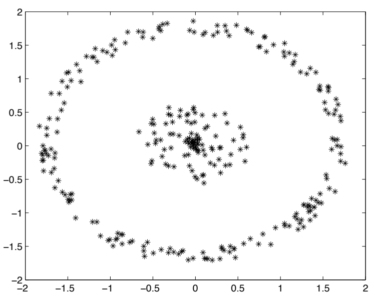
4.3 Results

To elaborate on the issues raised, four synthetic datasets are first used: Five-Cluster, Ring, Bar and Disc, all in R^2 because for such datasets ground truths are known and we can visually asses the results. Later, some real datasets will also be used.

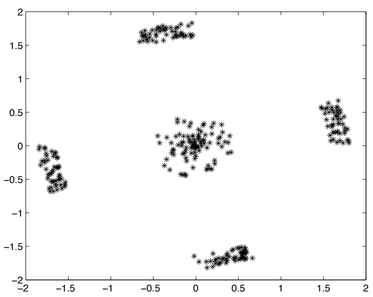
- **Ring** dataset: Figure 4.1a represents the Ring dataset, where the inner circle of radius 3 contains 100 data points and the outer ring contains 200 data points between radii 6 and 8.
- **Five-Cluster** dataset : Figure 4.1b displays the Five-Cluster dataset in which the inner disk contains 100 data points representing a cluster. These dataset also has four smaller clusters, each with 50 points, on four sides of the central cluster.
- **Bar** dataset: Figure 4.1c shows Bar dataset containing 300 data points. This dataset also does not have any cluster in it.
- **Disc** dataset: Figure 4.1d depicts the Disc dataset. It has 500 randomly generated data points on a disc of radius 5. This dataset does not have any cluster.

For each dataset, X, every feature x is normalized by $\frac{(x-\mu)}{s}$, where μ and s are the mean and standard deviation of x as estimated from the given dataset. This protocol is followed in [148]. These normalized data are used to generate Figure 4.1. In addition to these four datasets, a dataset called Uniform will also be used and discussed in the appropriate sections.

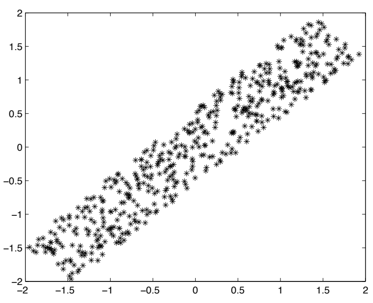
In all our clustering results, different clusters will be depicted using different colors as well as by different symbols.



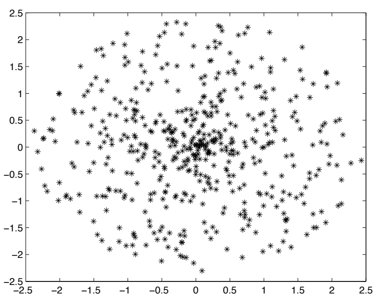
(a) Ring dataset



(b) FiveCluster dataset



(c) Bar dataset



(d) Disc dataset

Figure 4.1: The Four Synthetic datasets used in this study.

4.3.1 Strange Behavior of Kernel Clustering with Gaussian Kernel

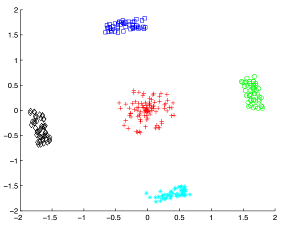
The behavior of FCM, KFCM-F, and KFCM-K with the Gaussian kernel is studied on the Five-Cluster, Ring, Disc, and Bar datasets. For all examples, the parameters used are $m = 1.2$, $\epsilon = 10^{-8}$, and $t_{\max} = 100$. The kernel width, σ is varied from 0.1 to 3 with a step size of 0.1. For KFCM-F the initial prototypes are taken randomly from the input dataset, while for KFCM-K the membership matrix is initialized with random values.

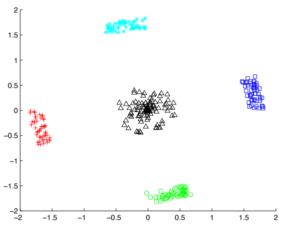
First we consider the Five-Cluster dataset (Figure 4.1b). How many clusters are there in this dataset? Undoubtedly, everybody would agree that there are five clusters. We examine the behavior of the three algorithms on Five-Cluster dataset with $c = 5, 4, 3$, and 2 clusters. The results are shown in Figure 4.2.

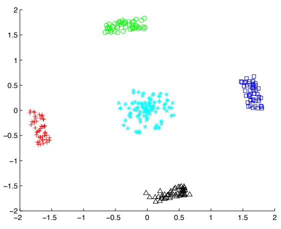
In case of FCM (Figures 4.2a, 4.2d, 4.2g, and 4.2j) we find that for $c = 5, 4$, and 3, the clustering results are quite intuitive; the clusters are as per our expectation. Similarly, for KFCM-F and KFCM-K (with $\sigma = 2.5$) all results are quite intuitive for $c = 3, 4$ and 5; all three algorithms can be said to produce good clustering of the dataset. But for $c = 2$, we would be in favor of the partition produced by FCM although it splits the central cluster, while the partition produced by KFCM-F is the worst one.

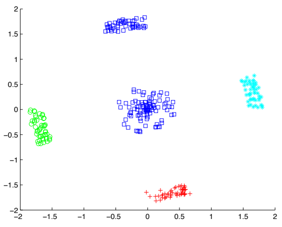
In case of KFCM-F (Figure 4.3a), there is nothing new compared to the results with higher kernel width, but in KFCM-K (Figure 4.3b) we get a shell type cluster, where all four outer clusters are combined to form single cluster! Then the question is: out of the Figures 4.2j, 4.3a and 4.3b which one represents a more logical clustering of the data for $c = 2$? We think, Figure 4.3b is the least intuitive partition for $c = 2$. But since the number of actual clusters does not match with c, one can argue that Figure 4.3b is also equally good. Subsequently, we use only KFCM-K algorithm except for the Ring data as it also has a shell.

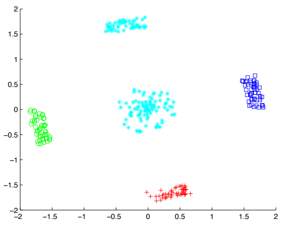
To further illustrate the underlying issue, the Bar dataset (Figure 4.1c) is used. The KFCM-K algorithm is applied with $\sigma = 0.2$ and $\sigma = 2$. The results are shown in Figures 4.4a and 4.4b.

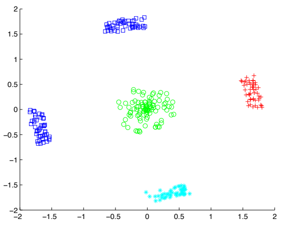

(a) FCM

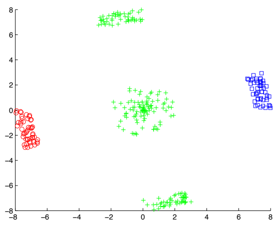

(b) KFCM-F

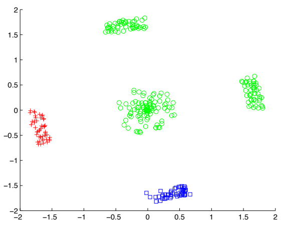

(c) KFCM-K

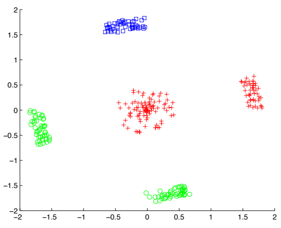

(d) FCM

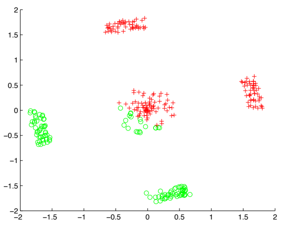

(e) KFCM-F

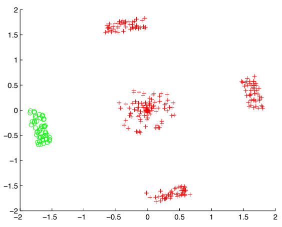

(f) KFCM-K


(g) FCM


(h) KFCM-F


(i) KFCM-K


(j) FCM


(k) KFCM-F

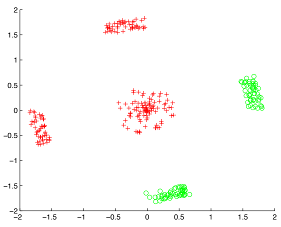
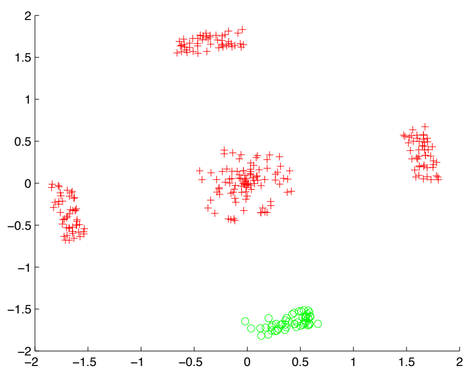
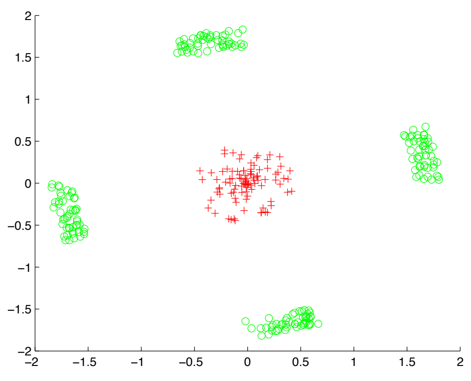

(l) KFCM-K

Figure 4.2: (a)–(c) show the five clusters found by FCM, KFCM-F, and KFCM-K, respectively. (d)–(f) show the four clusters found by FCM, KFCM-F, and KFCM-K, respectively. (g)–(i) show the three clusters found by FCM, KFCM-F, and KFCM-K, respectively. (j)–(l) show the two clusters found by FCM, KFCM-F, and KFCM-K, respectively [for KFCM-F and KFCM-K, $\sigma = 2.5$].



(a) KFCM-F



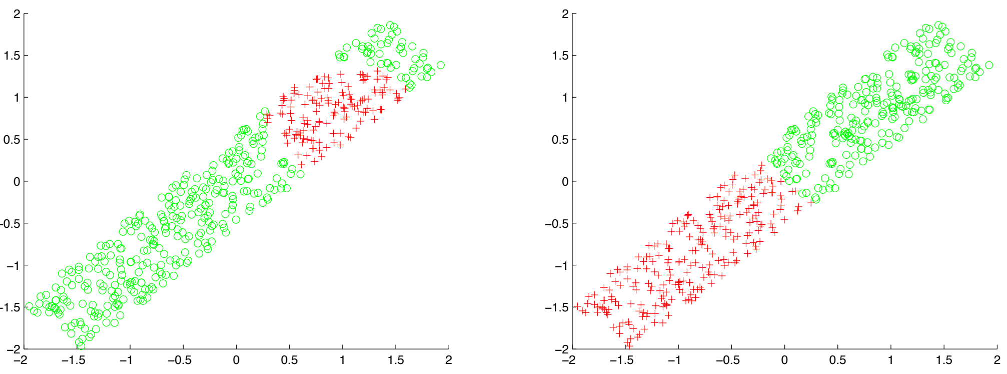
(b) KFCM-K

Figure 4.3: (a) & (b) show 2 clusters for $\sigma = 1.1$.

The same question is posed again: which partition is more acceptable? In this view, the answer should be Figure 4.4b, as there is no justification for accepting a partition like the one shown in Figure 4.4a. This might be an indicator to suggest that moderately large σ is better. One can also argue that the dataset has no cluster. Yes, that is true. But for any high-dimensional data we would not know if the dataset has clusters and if yes, how many? Yet, if we want to find c clusters, the algorithm should produce a partition that reflects the actual structure of the data.

The KFCM-K and KFCM-F algorithms are now applied to the Ring dataset (Figure 4.1a). Both algorithms were run for different values of σ, ranging from 0.1 to 3 with an increment of 0.1. The observations are summarized here. The KFCM-F for most of the runs (with different σs) could not extract the shell and disk as separate clusters. However, for $\sigma = 0.2$ for some runs, KFCM-F could find the shell and disk as separate clusters (see Figure 4.5). On the other hand, KFCM-K for σ from about 0.4 to about 1.5 could extract the shell and disk as separate clusters (Figure 4.6a). The KFCM-K produces a very good partition (a partition that we want to have).

So both KFCM-F and KFCM-K can find the shell structure when it is present and the choice of σ that produces the desirable partition for the two algorithms are different.



(a) $\sigma = 0.2$ (b) $\sigma = 2$

Figure 4.4: KFCM-K using Bar dataset.

Now let us see what happens with the Disc data (Figure 4.1), which do not have any cluster structure. With $c = 2$ we expect any clustering algorithm to divide the data roughly into two half-disks, where the line dividing the disk may have different orientation depending on the initialization. But it is strange to note that for $\sigma = 1$, KFCM-K imposes a nice shell structure on the Disc data (Figure 4.6b), but such a shell is *not* present in the data. This experiment is repeated with $\sigma = 1$ ten times, and in all cases, a shell-like structure is obtained. However, for the KFCM-F algorithm, the results are similar to those of the standard FCM.

Thus, it is observed that, depending on the choice of σ, KFCM-K identifies a shell structure if such a structure is present in the data.It can also find a shell structure even if it is not present in the data. In the former case, KFCM-K does a good job, while in the later case, it imposes a structure which is NOT there in the data. Hence, such clusters cannot serve any purpose. The success of KFCM-K to identify the Shell as a separate cluster is consistent with results in Figure 4.3b for Five-Clusters with $c = 2$ because the Five-Clusters dataset is generated by deleting some points from a Shell as in the Ring data!

Taking a closure look into the results shown in Figures 4.3b, 4.4a, 4.6a and 4.6b, we *may* say that the Gaussian kernel tries to find/impose a shell type structure at low σ, irrespective of whether the dataset has such

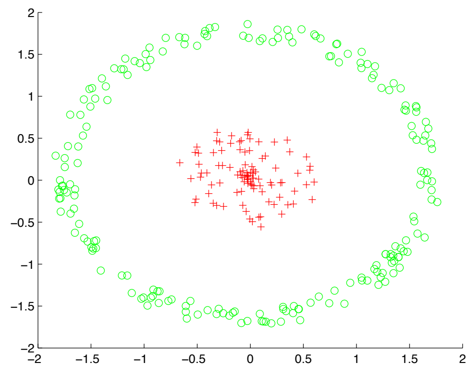


Figure 4.5: The KFCM-F partition of the Ring dataset with $\sigma = 0.2$.

a structure or not; but this is not a proof. One can argue that the dataset Disc has no cluster and hence the problem. But for any real data in more than 3-D, as already mentioned, we shall neither know if there any cluster, nor the number of clusters, if any. When attempting to find c clusters, the goal is to obtain a partition that closely aligns with the actual structure of the data.

As indicated in the introduction, we have experimented with both polynomial and Hyper-tangent kernels. In [86], results using polynomial kernels were reported. Here, a few illustrative results are included in Figure 4.7. For each dataset in Figure 4.7, two results are presented: the best one (based on visual assessment) and a poor one derived from the set of kernel parameters tried. For the Ring dataset, none of the parameters tested were able to separate the ring from the central cluster. On the other hand, for the Uniform Disk, it imposed a very nice shell. Only for Five cluster data, Polynomial kernel separates the clusters [86].

Figure 4.8 displays some results on the four datasets using the Hyper-tangent kernel. For the Uniform Disk and Bar datasets, Hyper-tangent kernel produces the desirable partition for a wide range of a. For the Bar dataset, all values of a between 0.1 and 5.0 that were tested resulted in the desirable partition shown in Figure 4.8a. For the Uniform Disk data also for a wide range of a the results are good. For the Ring and Five cluster datasets, we could not extract the desirable partitions using

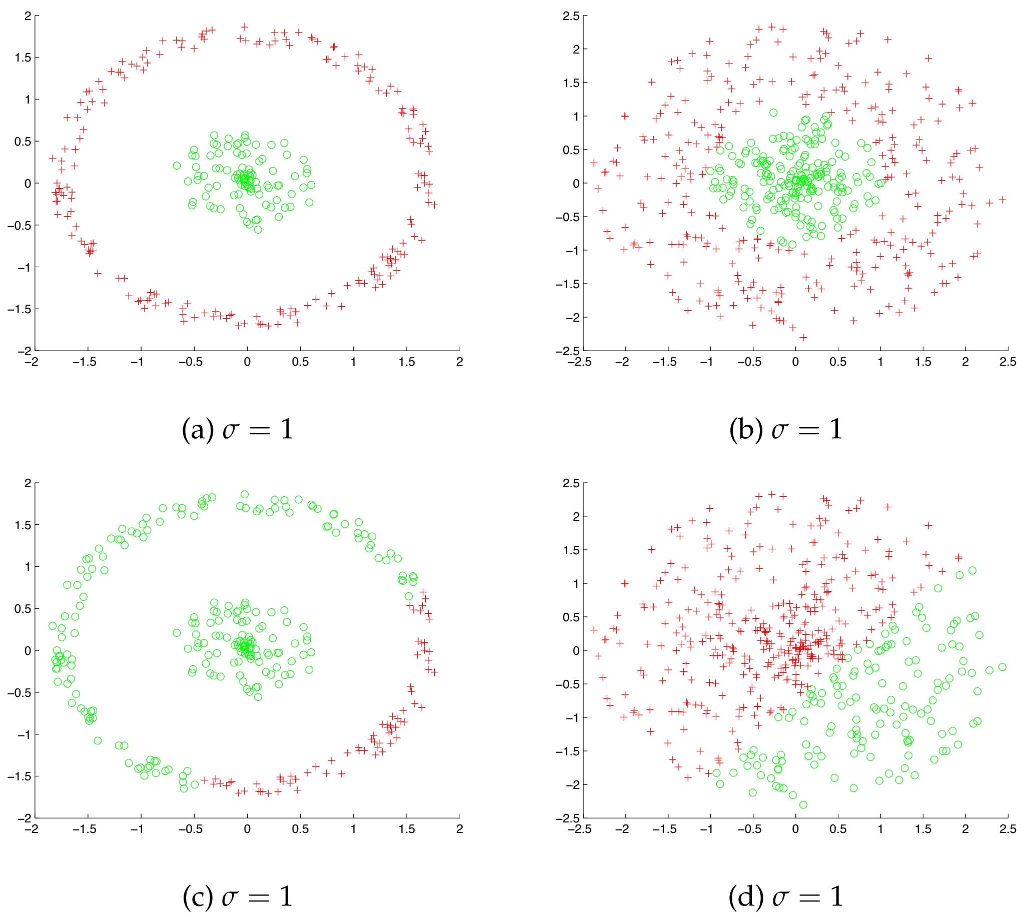

(a) $\sigma = 1$ (b) $\sigma = 1$ (c) $\sigma = 1$ (d) $\sigma = 1$

Figure 4.6: (a-b) using KFCM-K and (c-d) using KFCM-F.

the Hyper-tangent kernel; in fact for the Five Cluster, all partitions are equally bad and hence we include one result for this dataset in Figure 4.8b. The results displayed in Figures 4.7 and 4.8, along with those presented in Chapter 3, indicate that the issues identified for the RBF kernel are equally applicable to the other two kernels.

4.3.2 Are these reported results typical?

The results reported above for different choices of the kernel parameters are very typical. Since the results of kernel clustering also depends on the initialization, for each choice of the kernel parameter, it is necessary to check the robustness of the results with respect to initialization. To do this, we have repeated the experiments 30 times and the quality of the clusters is assessed visually. Since these datasets are in 2-D and the desired partition is known, visual assessment was easy (for the Disk Data any 2-partition along any diagonal is a desirable partition). Moreover, visual assessment was more appropriate because we are more concerned with the identity of the points which are mis-clustered than the number of points which are wrongly clustered. The visual assessments are summarized in Table 4.1. For each of Five-Cluster and Ring, there is a unique desirable partition. For Bar data also the FCM partition which divides it into two almost equal halves can be taken as the most desired partition. As already explained, for Disk data there is no unique desirable partition. To provide a quantitative assessment of the clustering results in Table 4.1, the average and standard deviation of the Normalized Mutual Information (NMI) [134] and Adjusted Rand Index (ARI) [135] are also included. The extremely low values (practically zero) of the standard deviation suggest that partitions produced practically did not depend on the initialization, but on the value of σ. So the strange results produced by the clustering algorithm are the effect of the imposition of improper structure on the data.

4.4 Cluster (in)Stability

Some researchers have proposed measures of cluster instability [159] as cluster quality index. To assess the instability of clustering results, the clustering algorithm is repeatedly run on a perturbed version of the original dataset. Then the consistency of the partitions over different perturbed datasets is assessed based on a measure of instability [159]. The algorithm below is used to compute the cluster instability index.

(a) Given: a set, S, of N data points.

(b) For $b = 1$ to 20

Table 4.1: Visual assessment and mean of NMI & ARI (SD of NMI & ARI) of the clustering results produced by KFCM-K for different choices of the kernel parameter σ for all datasets used. Number of clusters = 2 for all datasets.The reported values are for 30 runs.

DATA	σ	**Visual assessment**	**NMI(SD)**	**ARI(SD)**
Ring	0.1	Worst	0.27(0.0)	0.24(0.0)
	0.2	Bad	0.40(0.0)	0.40(0.0)
	0.3	Medium	0.69(0.0)	0.76(0.0)
	0.4	Good	0.97(0.0)	0.99(0.0)
	0.5	Good	1.0(0.0)	1.0(0.0)
	0.6	Good	1.0(0.0)	1.0(0.0)
	up to 1.4	Good	1.0(0.0)	1.0(0.0)
	1.5	Bad	0.28(0.0)	0.11(0.0)
	Beyond 1.5	Bad		
Five Cluster	0.1	Bad	0.42(0.0)	0.42(0.0)
	0.2	Bad	0.69(0.0)	0.76(0.0)
	0.3	Good	0.89(0.0)	0.92(0.0)
	0.4 to 0.9	Good	1.0(0.0)	1.0(0.0)
	1.0	Good	0.98(0.13)	0.97(0.16)
	1.3	Bad	0.71(0.31)	0.67(0.36)
	1.5	Bad	0.44(0.21)	0.35(0.28)
Bar	0.1	Bad	0.07(0.06)	0.01(0.02)
	0.2	Bad	0.30(0.09)	0.20(0.07)
	0.3	Medium	0.61(0.0)	0.62(0.0)
	0.4	Good	0.70(0.0)	0.74(0.0)
	0.5	Good	0.72(0.0)	0.77(0.0)
	0.6 and above	Good	0.75(0.0)	0.82(0.0)

i. Generate a perturbed versions S_b of the original dataset (by subsampling with repeatation).

ii. Apply the clustering algorithm on S_b. Let C_b be the corresponding cluster output.

iii. Apply the clustering algorithm again on S_b with a different initialization. Let C'_b be the corresponding cluster output.

iv. Compute the instability between C_b and C'_b using the equation (4.13).

(c) Compute the mean and standard deviation of instability.

$$d(C_b, C_b^{'}) = min_{\pi} \frac{1}{N} \sum_{i=1}^{N} C_b(x_i) \neq \pi C_b^{'}(x_i). \tag{4.13}$$

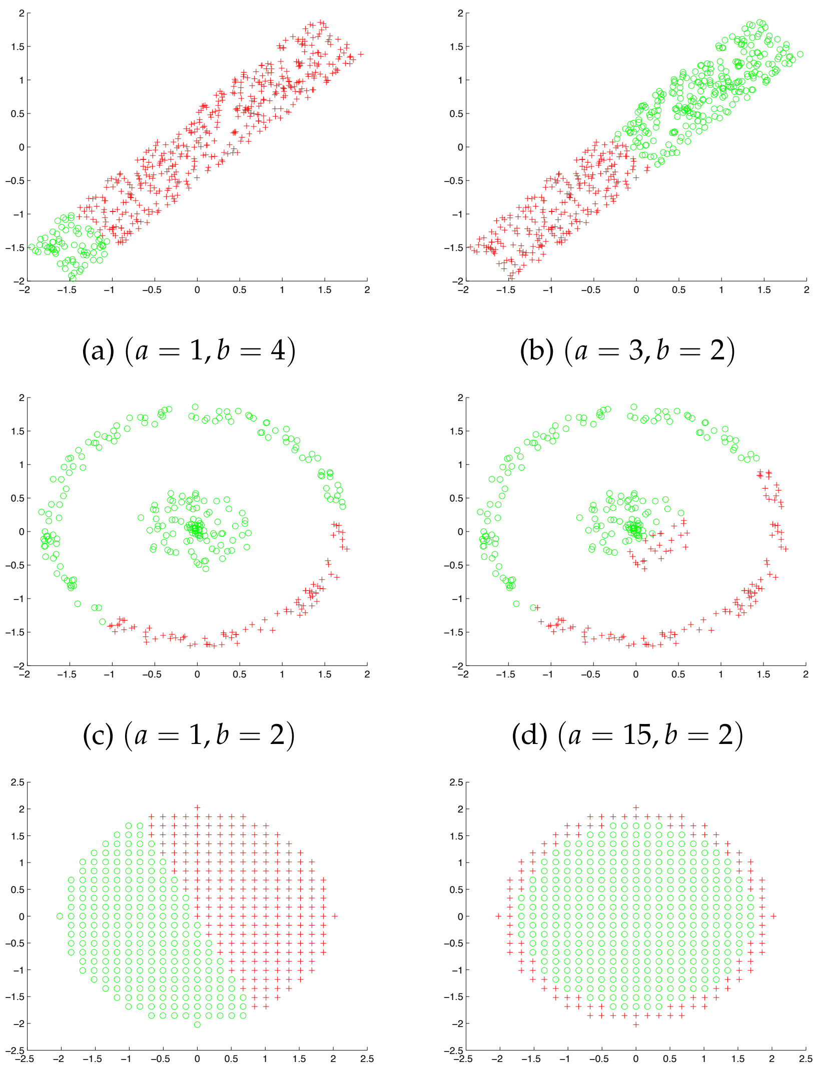
In equation (4.13), π denotes a permutation of the class labels and the minimum is taken over all permutations. Instability is a good cluster validity criterion and if the dataset has well separated cluster structure, it will perform good. But as mentioned earlier, if the kernel projection imposes non-existent but well differentiated sub-structures, like any cluster validity indices, instability also may select inappropriate partition. The usefulness of the instability measure in selecting the appropriate values of the kernel parameters is now examined. For this purpose, three datasets—Bar, Ring, and Uniform—are considered, along with three types of kernels: Gaussian, Polynomial, and Hyper-tangent. For each dataset and each kernel, wherever possible we consider two choices of kernel parameters: one that generates the most desirable partition and one which produces an undesirable partition. Table 4.2 summarizes the instability values for these three datasets. Table 4.2 reveals that in majority of the cases, instability can pick the desirable partition. However, in several cases it fails too. For example, in case of Ring with Gaussian kernel ($\sigma = 0.2$), like the case with desirable partition, instability exhibits practically a zero value. On the other hand, for Uniform data with Gaussian kernel an improper partition yields a very low (lower than that of a good partition) value of the index. For the same dataset with polynomial kernel, the complement behavior is exhibited. Here for a very poor partition

(Figure 4.7f), the instability index is zero. For the Bar using the Hyper-tangent kernel we get good clusters for a wide range of $a = 0.1 - 5.0$ (Figure 4.8a) and that is why in the Table just one value is shown. Thus even for very simple datasets, the stability index cannot address the issue of choosing the kernel parameters. The reason behind is the same as that for any other cluster validity index.

Table 4.2: Instability measure for different kernels.

		Instability		Visual
Data set	Kernel type	Mean	SD	assessment
Bar	Gaussian (σ=0.2), Figure 4.4a	0.08	0.14	Bad
	Gaussian (σ=2), Figure 4.4b	0	0	Good
Ring	Gaussian (σ=0.2), Figure 4.12g	0.0	0.0	Bad
	Gaussian (σ=1), Figure 4.12a	0	0	Good
Uniform	Gaussian (σ=0.1), Figure 4.11a	0.09	0.05	Bad
	Gaussian (σ=0.4), Figure 4.11g	0.13	0.15	Good
Bar	Polynomial (a=1, b= 4), Figure 4.7a	0.06	0.12	Bad
	Polynomial (a=3, b=2), Figure 4.7b	0	0	Good
Ring	Polynomial (a=1, b=2), Figure 4.7c	0.21	0.20	Bad
	Polynomial (a=15, b=2), Figure 4.7d	0.11	0.17	Bad
Uniform	Polynomial (a=1, b=2), Figure 4.7e	0.13	0.18	Good
	Polynomial (a=1, b=12), Figure 4.7f	0	0	Bad
Bar	Hyper-tangent (a=1), Figure 4.8a	0	0	Good
Ring	Hyper-tangent (a=0.3), Figure 4.8c	0.08	0.15	Bad
	Hyper-tangent (a=3.5), Figure 4.8d	0.18	0.17	Bad
Uniform	Hyper-tangent (a=0.3), Figure 4.8e	0.05	0.13	Good
	Hyper-tangent (a=5), Figure 4.8f	0.24	0.17	Marginally Bad

Since the utility of clusters produced by kernel clustering depends on the structure imposed by the kernel function on the dataset, next an attempt is made next to visualize the data structure in the kernel space.

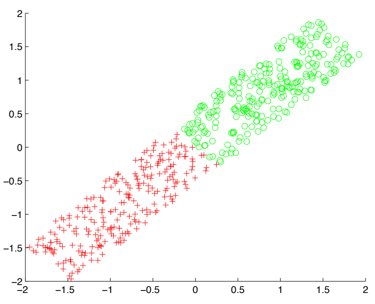


(a) $(a = 1, b = 4)$ (b) $(a = 3, b = 2)$

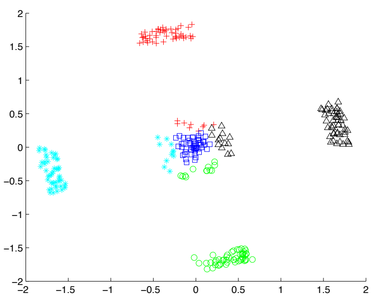
(c) $(a = 1, b = 2)$ (d) $(a = 15, b = 2)$

(e) $(a = 1, b = 2)$ (f) $(a = 3, b = 12)$

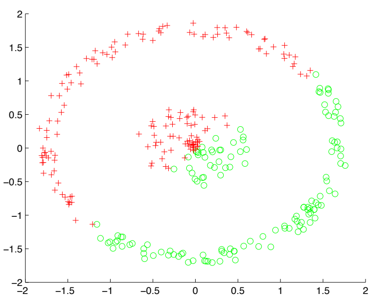
Figure 4.7: Clustering synthetic dataset by KFCM-K using Polynomial kernel



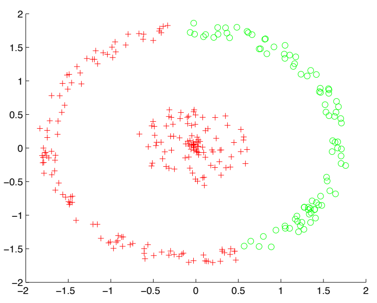
(a) $a = 1$



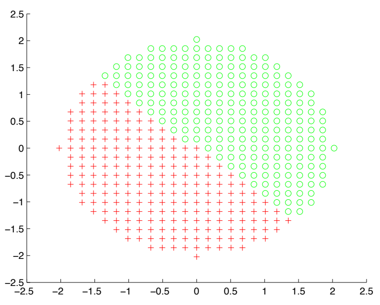
(b) $a = 1.5$



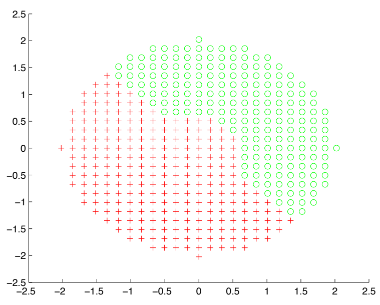
(c) $a = 0.3$



(d) $a = 3.5$



(e) $a = 0.3$



(f) $a = 5$

Figure 4.8: Clustering synthetic dataset by KFCM-K using Hyper-tangent kernel.

4.5 How do the data look like in the kernel Space?

The strange behavior of kernel clustering results above raises the following questions :

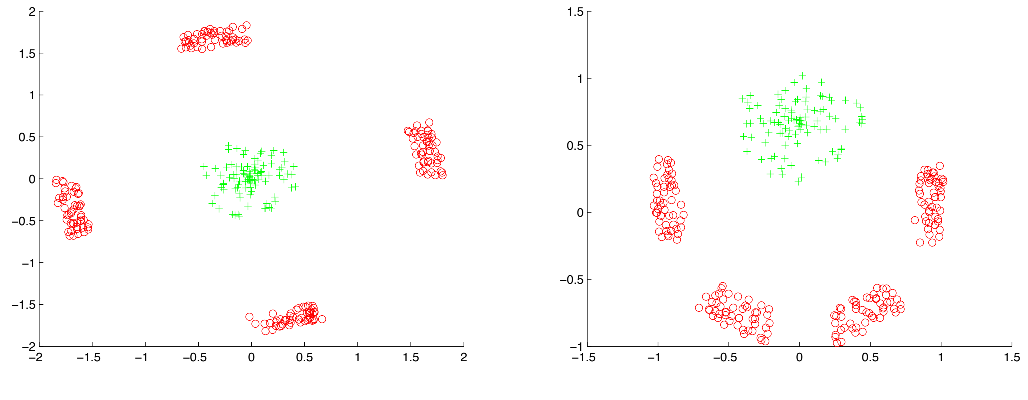
- How do the data look like in the kernel space?
- Does it maintain the structure of the original data?
- Does kernelization impose strange structures on the data?

Sammon's nonlinear projection [6] is used to address these issues. Let d_{ij} be the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ in the original p-dimensional feature space and and kd_{ij} be the "distance" between the same pair points in the kernel space. Thus, $kd_{ij} = \sqrt{2(1 - K(\mathbf{x_i}, \mathbf{x_j}))}$. So to get a visual representation of data in the kernel space, we find a set of 2-D points in the Euclidean space such that distance structure in the kernel space is preserved by the set of 2-D points. In other words, we find a set of points $Y = \{\mathbf{y}_i; i = 1, \cdots, n\}; \mathbf{y}_i \in R^2$, such that $kd_{ij} = ||\mathbf{y}_i - \mathbf{y}_j||$. Sammon's stress function gives a good measure of how the inter-point distances in two different spaces are preserved. The matrix Y is obtained by minimizing Sammon's stress function, defined as:

$$SE(K;Y) = \frac{1}{\sum\sum_{i<j} kd_{ij}} \sum_i \sum_{j,i<j} \frac{(kd_{ij} - ||\mathbf{y}_i - \mathbf{y}_j||)^2}{kd_{ij}}. \tag{4.14}$$

Details of Sammon's algorithm can be found in [6]. In order to emphasize our points, we consider the Disc data shown in Figure 4.10 where the data points are uniformly placed (equispaced) on the disc. Such a uniform data is used to minimize the effect the distribution of the data on the clustering results. Several runs of the KFCM-K are conducted for different choices of the kernel width σ. Figure 4.11 displays the results for four different choices of σ. The left panel shows the kernel clustering results labeled on the original data, while the right panels exhibits the same cluster structure on the projected data (data projected on 2-D by Sammon's algorithm). In other words, a set of 2-D points is first determined to preserve the kernel distance matrix. Then if a point in original data appears in cluster 1 (labeled as red, represented by o) then the corresponding point generated by the Sammon's projection is also labeled as belonging to cluster 1 (i.e., labeled as red, represented by o). The remaining points (cluster 2) are labeled as green, represented by $+$. From these figures we see that for lower value of σ, kernelization imposes strange structure on the data points. As the value of σ increases, the results start to become more meaningful; the KFCM-K partition becomes almost identical to the FCM partition.

Figures 4.12 and 4.13 present the 2-D representation of the kernel data for the other datasets, along with the corresponding cluster labeling. The Sammon-projected data in Figure 4.12a (for $\sigma = 1$) reveal how nicely kernalization makes the outer ring almost linearly separable from the inner disk. Figure 4.12b suggests that in the kernel space the shell is folded along a diameter and hence it was easy for the clustering algorithm to obtain the desired partition. Similarly, Figure 4.9 also explains why with $c = 2$, KFCM-K merges all four small clusters into a single cluster (the partition in Figs. 4.13e and 4.13f is, of course, for $c = 5$). It is interesting to observe the significant difference in the projected data for Bar with $\sigma = 0.1$ and 1.0 (Figs. 4.12d and 4.12f). Hence for these two values of σ, significantly different clustering results are produced. We note that Sammon's algorithm terminates at a local minimum of the stress function and hence different runs (different initial conditions) of the algorithm may yield different projections. The algorithm is quite robust and on termination if the stress function is small for two different runs, the projections are usually similar in structure although one may be an approximately rotated version of the other. To demonstrate the effect of initialization on the projection generated by Sammon's algorithm, the experiments were repeated 30 times for each choice of σ, each with a different initialization. Table 4.3 reports the mean and standard deviation of the stress function. The very low values of the standard deviation indicate that the projections are practically independent of the initialization in terms of preservation of inter-point distances. Table 4.3 reveals that at lower values of σ, the terminal value of Sammon's stress is higher. This indicates that the structure/topology of the data is changed so much that it cannot be preserved in 2-D, although the actual data were in 2-D. Note that this cannot be attributed to initialization because standard deviation of the stress value is practically zero. Since the original data are in 2-D, the projections have been restricted to 2-D as well.



(a) Cluster result (b) Projected result

Figure 4.9: Clustering and Sammon projection of Five-Cluster data for $\sigma = 1$.

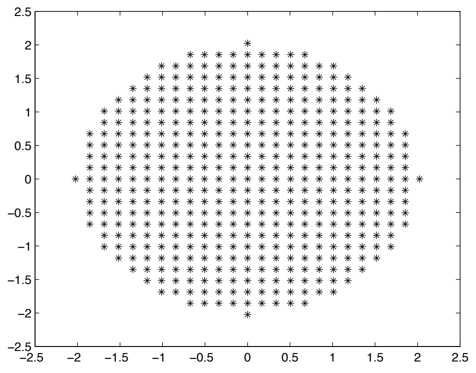
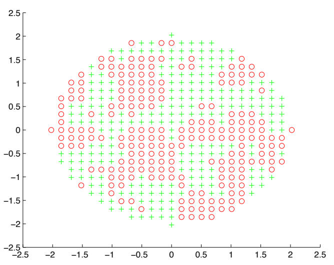
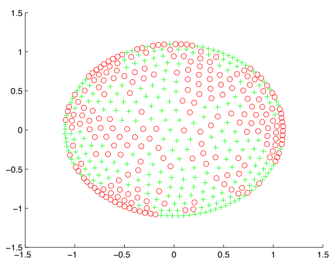


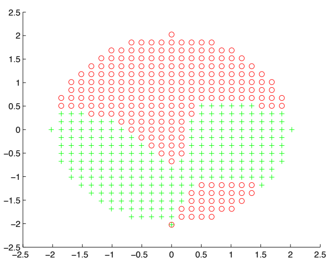
Figure 4.10: Disk with uniform data points

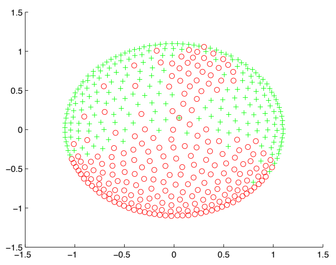
4.6 Kernel clustering on some real datasets

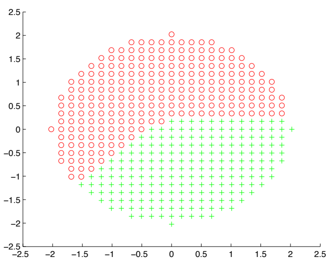
Up to this point, only 2-D synthetic data have been considered, as the known cluster structures in synthetic data allow for easier validation,

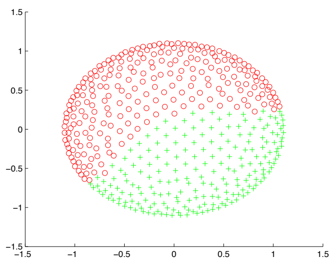

(a) Cluster result ($\sigma = 0.1$)

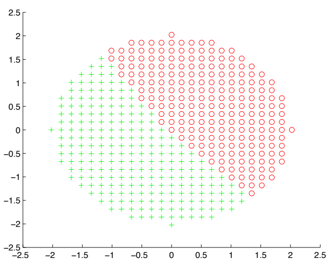

(b) Projected result ($\sigma = 0.1$)


(c) Cluster result ($\sigma = 0.2$)


(d) Projected result ($\sigma = 0.2$)


(e) Cluster result ($\sigma = 0.3$)


(f) Projected result ($\sigma = 0.3$)


(g) Cluster result ($\sigma = 0.4$)

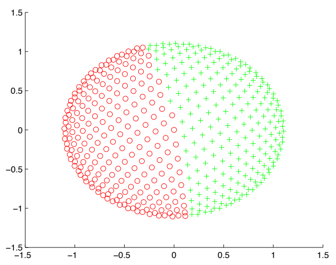
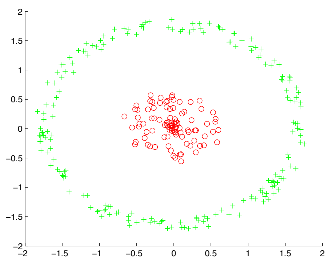

(h) Projected result ($\sigma = 0.4$)

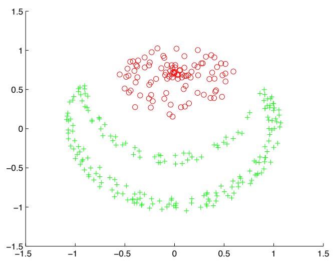
Figure 4.11: Clustered output and corresponding projected result for different σ.

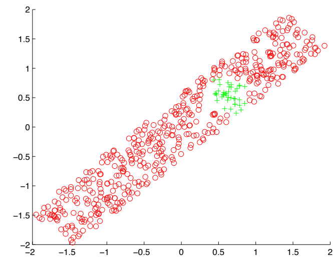
Table 4.3: The values of the Mean (Standard deviation) of the stress function on termination of Sammon's projection algorithm for different choices of σ, i.e., for different kernel data. The reported values are computed based on 30 runs.

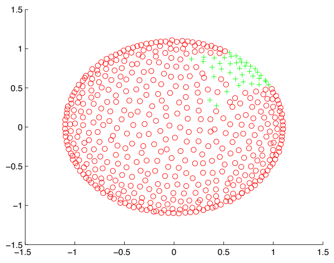
	Ring	Disk	Bar	Uniform Disk	FiveCluster
σ	Stress(SD)	Stress(SD)	Stress(SD)	Stress(SD)	Stress(SD)
0.1	0.17(0.0)	0.18(0.0)	0.17(0.0)	0.18(0.0)	0.16(0.0)
0.2	0.16(0.0)	0.17(0.0)	0.16(0.0)	0.17(0.0)	0.14(0.0)
0.3	0.15(0.0)	0.16(0.0)	0.15(0.0)	0.17(0.0)	0.12(0.0)
0.4	0.14(0.0)	0.16(0.0)	0.13(0.0)	0.16(0.0)	0.11(0.0)
0.5	0.13(0.0)	0.15(0.0)	0.12(0.0)	0.15(0.0)	0.09(0.0)
0.6	0.12(0.0)	0.14(0.0)	0.10(0.0)	0.14(0.0)	0.09(0.0)
0.7	0.12(0.0)	0.13(0.0)	0.08(0.0)	0.13(0.0)	0.08(0.0)
0.8	0.11(0.0)	0.12(0.0)	0.07(0.0)	0.12(0.0)	0.07(0.0)
0.9	0.10(0.0)	0.11(0.0)	0.06(0.0)	0.11(0.0)	0.07(0.0)
1.0	0.09(0.0)	0.10(0.0)	0.05(0.0)	0.10(0.0)	0.06(0.0)

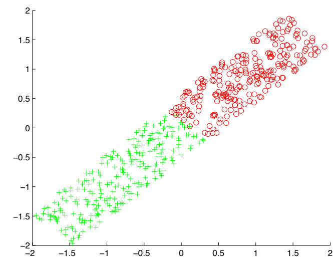
and 2-D data can be readily visualized. The focus now shifts to investigating the effect of kernel clustering on real datasets. Specifically, four datasets from the UCI Machine Learning Repository [132] are considered: Wine, Ionosphere, Iris, and Wisconsin Breast Cancer Diagnostic (WBCD). The datasets are summarized in Table 4.4. Since the dimensionality of these datasets vary between 4 to 34, we cannot visualize the datasets and do not have any idea about the number of clusters present. However, for each dataset, the number of classes it represents is known. Many authors used such datasets to evaluate performance of kernel clustering [148, 164] assuming the number of clusters equal to the number classes. Not only that, these methods implicitly assume that each class corresponds to a single cluster because they find the agreement (or misclassification) between the partition produced by a clustering algorithm with the partition imposed by the classes. It is important to caution readers that such assumptions may not be valid, and in most cases, they are not. When comparing multiple clustering algorithms, the use of Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) can be meaningful for ranking the performance of the algorithms. Neverthe-

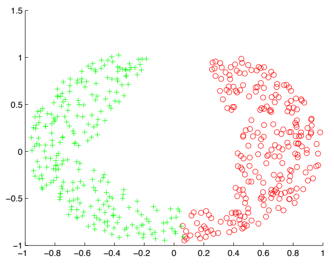

(a) Cluster result ($\sigma = 1$)

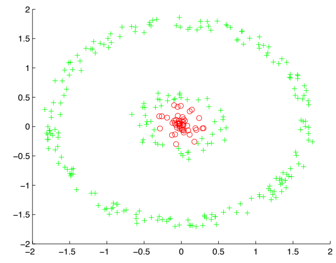

(b) Projected result ($\sigma = 1$)


(c) Cluster result ($\sigma = 0.1$)


(d) Projected result ($\sigma = 0.1$)


(e) Cluster result ($\sigma = 1$)


(f) Projected result ($\sigma = 1$)


(g) Cluster result ($\sigma = 0.2$)

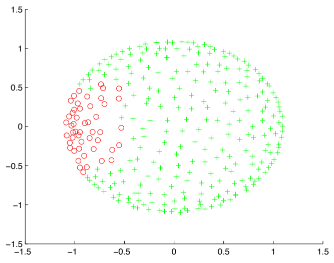
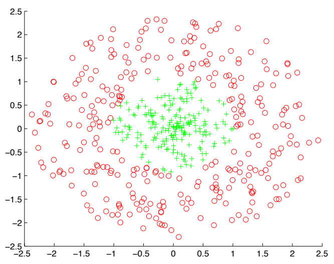
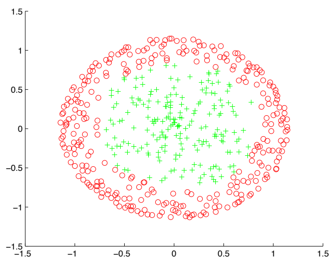
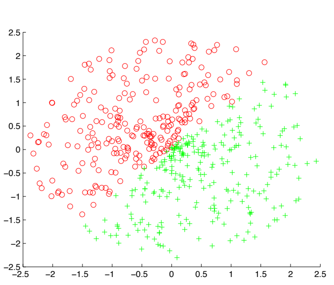
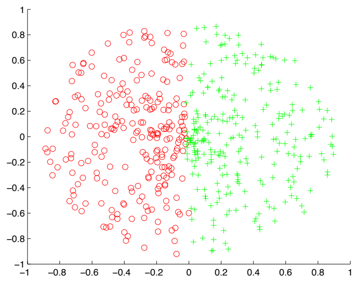

(h) Projected result ($\sigma = 0.2$)

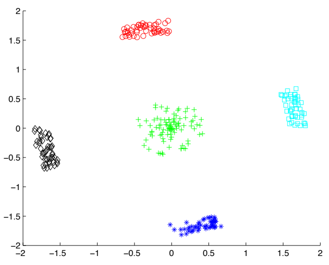
Figure 4.12: Clustering of the Synthetic datasets by KFCM-K along with the Sammon's projection of the kernelized data.


(a) Cluster result ($\sigma = 1$)


(b) Projected result ($\sigma = 1$)


(c) Cluster result ($\sigma = 3$)


(d) Projected result ($\sigma = 3$)


(e) Cluster result ($\sigma = 1$)

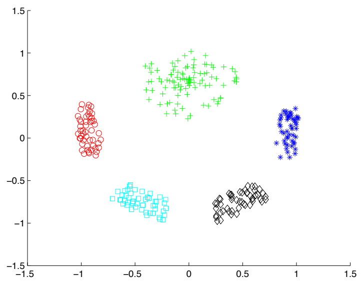

(f) Projected result ($\sigma = 1$)

Figure 4.13: Clustering of the synthetic datasets by KFCM-K along with the Sammon's projection of the kernelized data.

less, since such protocols have been followed in other studies, the same approach is adopted here. 13 different choices of σ is used over a wide range, and for each choice, the clustering algorithm is repeated 30 times. Table 4.5 reports the mean and standard deviation of NMI and ARI. Note here that for the Iris data, with number of clusters = 3, the algorithm failed to find 3 clusters in 8 and 14 cases out of 30 runs with $\sigma = 0.2$ and $\sigma = 0.4$ respectively. Note that this failure of the algorithm is dependent on the initial condition. Table 4.5 shows the statistics over the cases in which the algorithm was successful.

It is well known that for Iris data, class one is well separated from the other two classes, while class two and three have some overlap. A typical clustering algorithm makes a mislabeling of $12 - 16$ points from classes two and three. To have a deeper look, in Figure 4.14 the clustering results of the Iris dataset are plotted. We have used features three and four to generate these plots. The three classes are represented by three symbols 'star','plus' and 'square' while the clusters are represented by three colours. An interesting result occurs for $\sigma = 0.6$ (Figure 4.14a), where one of the clusters (shown in colour) merges points from class one and class three, which is again a very strange results caused by improper structure imposed by the nonlinear transformation. For Iris data the average ARI and NMI varies between $0.07 - 0.62$ and $0.23 - 0.66$ respectively. This clearly suggests that kernelization imposes widely varying non-existent structure on the data. The last row of Table 4.5 reports the NMI and ARI values for 30 runs of FCM. This row reveals that for many choices of σ, the kernel clustering produces much worse results compared to FCM and for high σ kernel clusters is similar to those obtained by FCM.

From Table 4.5 we find similar situations with the other three datasets. For example, in case of Wine data, the ARI varies between 0 to 0.90. Thus depending on the choice of σ the agreement between the class structure and cluster structure extracted by kernel clustering could be worse than the agreement due to chance and it could be even almost perfect. But the user has no way to know, which kernel transformation would yield the desired partition. More importantly, the user has no way to know which kernel transformation will impose non-existent structure leading to useless clustering. The situation is worst for the Ionosphere data as for a wide range of σ, the ARI is even worse than the agreement by chance and obviously for those cases the NMIs are also very poor. The best values of ARI and NMI achieved for this dataset are 0.18 at $\sigma = 15$ and 0.24 at $\sigma = 4$, respectively.

Comparing Table 4.5 with Table II and Table III of [164] which uses mul-

Table 4.4: Summary of the real datasets.

Name	Instances	Features	Classes
Wine	178	13	3
Ionosphere	351	34	2
Iris	150	4	3
Wisconsin Breast Cancer (D)	569	30	2

tiple kernels, we find that the KFCM-K with just a single kernel can produce ARI and NMI comparable to those by the use of multiple kernels for the three UCI datasets. In fact, for the Ionosphere data the single kernel KFCM-K can yield better ARI and NMI than those by multiple kernels. This again reemphasizes the fact that the use of multiple kernel does not address the problem that we are referring to.

It is also interesting to note that for Wine data, with σ in the neighborhood of $2.2(1.9 - 2.3)$, the clustering algorithm failed to find 3 clusters.

Since we are hardening the fuzzy partition into a hard partition and computing NMI and ARI, one can question that the results might be different if we use a hard clustering algorithm. In order to check this, we have run the KFCM-K algorithm with $m = 1.02$ which produces practically a hard partition. Table 4.6 summarizes the NMI and ARI values of over 30 runs of KFCM-K with $m = 1.02$. The last row of the table reports the NMI and ARI for FCM with $m = 1.02$ (i.e these are really the results of the hard c-means algorithm). A comparison of Table 4.6 with Table 4.5 reveals that the observations made in Table 4.5 are also applicable to Table 4.6.

Thus, the conclusions drawn based on the synthetic datasets are equally valid even for the real datasets.

4.7 Comments on choice of kernel parameters

To use any of the above kernelized clustering algorithms on a data set, we need to choose appropriate values for the kernel parameters. In par-

Table 4.5: The values of Mean (Standard deviation(SD)) of Normalized Mutual Information(NMI) and Adjusted Rand Index (ARI) for different choices of σ taking $m = 1.2$. The reported values are for 30 runs.

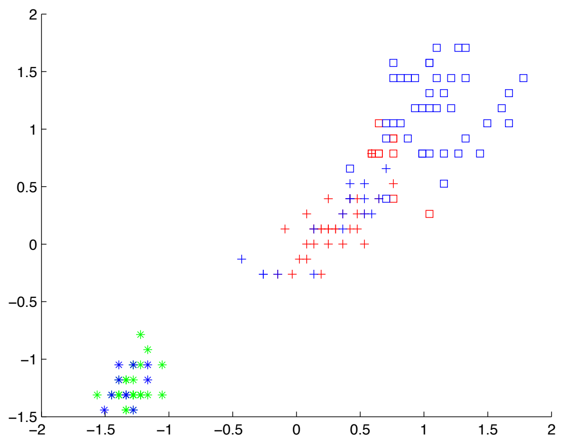
	Wine		Iris	
	NMI	ARI	NMI	ARI
σ	Mean(SD)	Mean(SD)	Mean(SD)	Mean(SD)
0.2	0.01(0.01)	-0.002(0.01)	0.23(0.05)	0.07(0.03)
0.4	0.01(0.01)	-0.00(0.01)	0.35(0.10)	0.17(0.09)
0.6	0.01(0.01)	-0.00(0.01)	0.48(0.0)	0.41(0.0)
0.8	0.02(0.01)	0.01(0.01)	0.56(0.0)	0.53(0.0)
1.0	0.16(0.09)	0.14(0.08)	0.60(0.0)	0.57(0.0)
1.2	0.44(0.12)	0.33(0.09)	0.61(0.0)	0.59(0.0)
2.2	0 (0.0)	0(0.0)	0.66(0.01)	0.61(0.03)
4.0	0.89(0.0)	0.91(0.0)	0.66(0.01)	0.61(0.03)
10.0	0.89(0.0)	0.91(0.0)	0.66(0.0)	0.62(0.0)
15.0	0.88(0.0)	0.90(0.0)	0.66(0.0)	0.62(0.0)
20.0	0.88(0.0)	0.90(0.0)	0.66(0.0)	0.62(0.0)
30.0	0.88(0.0)	0.90(0.0)	0.66(0.0)	0.62(0.0)
50.0	0.88(0.0)	0.90(0.0)	0.66(0.0)	0.62(0.0)
FCM	0.88(0.0)	0.90(0.0)	0.66(0.0)	0.62(0.0)

	Ionosphere		Wisconsin Breast Cancer(D)	
	NMI	ARI	NMI	ARI
σ	Mean(SD)	Mean(SD)	Mean(SD)	Mean(SD)
0.2	0.01(0.01)	0.003(0.01)	0.00(0.002)	0.00(0.002)
0.4	0.08(0.001)	-0.04(0.0)	0.001(0.001)	-0.00(0.002)
0.6	0.11(0.0)	-0.05(0.0)	0.002(0.003)	0.002(0.004)
0.8	0.14(0.0)	-0.04(0.0)	0.002(0.002)	0.001(0.002)
1	0.15(0.0)	-0.04(0.0)	0.11(0.06)	-0.02(0.01)
1.2	0.17(0.0)	-0.03(0.0)	0.16(0.01)	-0.01(0.0)
2.2	0.21(0.0)	0.06(0.0)	0.26(0.0)	0.16(0.0)
4	0.24(0.0)	0.18(0.0)	0.40(0.0)	0.46(0.0)
10	0.14(0.0)	0.18(0.0)	0.54(0.0)	0.67(0.0)
15	0.14(0.0)	0.18(0.0)	0.55(0.0)	0.67(0.0)
20	0.13(0.0)	0.17(0.0)	0.54(0.0)	0.66(0.0)
30	0.12(0.0)	0.16(0.0)	0.56(0.0)	0.68(0.0)
50	0.12(0.0)	0.16(0.0)	0.55(0.0)	0.67(0.0)
FCM	0.12(0.0)	0.16(0.0)	0.56(0.0)	0.66(0.0)

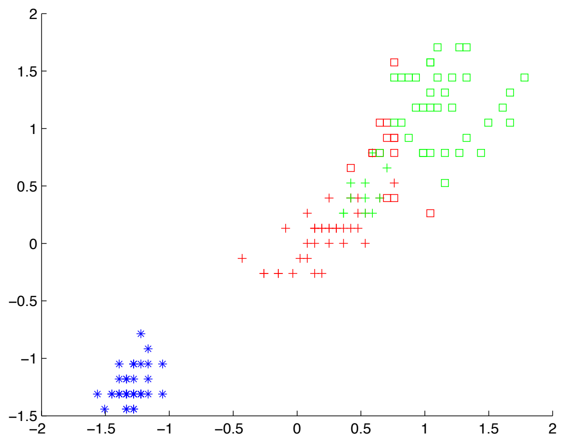
Table 4.6: The values of Mean (Standard deviation(SD)) of Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) for different choices of σ, taking $m = 1.02$. The reported values are for 30 runs.

	Wine		Iris	
	NMI	ARI	NMI	ARI
σ	Mean(SD)	Mean(SD)	Mean(SD)	Mean(SD)
0.2	0.01(0.01)	0.001(0.01)	0.21(0.09)	0.12(0.07)
0.4	0.01(0.01)	0.004(0.01)	0.37(0.05)	0.21(0.04)
0.6	0.01(0.01)	0.002(0.01)	0.41(0.07)	0.30(0.07)
0.8	0.17(0.07)	0.10(0.05)	0.51(0.08)	0.43(0.09)
1.0	0.39(0.08)	0.23(0.08)	0.59(0.04)	0.54(0.08)
1.2	0.48(0.09)	0.39(0.13)	0.64(0.01)	0.61(0.01)
2.2	0.77(0.02)	0.78(0.05)	0.66(0.01)	0.62(0.05)
4.0	0.93(0.0)	0.95(0.0)	0.65(0.01)	0.59(0.05)
10.0	0.89(0.0)	0.91(0.0)	0.66(0.01)	0.60(0.06)
15.0	0.88(0.0)	0.90(0.0)	0.65(0.03)	0.59(0.07)
20.0	0.88(0.0)	0.90(0.0)	0.66(0.02)	0.60(0.05)
30.0	0.88(0.0)	0.90(0.0)	0.65(0.03)	0.58(0.07)
50.0	0.88(0.0)	0.90(0.0)	0.65(0.02)	0.60(0.06)
FCM	0.88(0.0)	0.90(0.0)	0.65(0.03)	0.58(0.07)

	Ionosphere		Wisconsin Breast Cancer(D)	
	NMI	ARI	NMI	ARI
σ	Mean(SD)	Mean(SD)	Mean(SD)	Mean(SD)
0.2	0.04(0.0)	0.01(0.0)	0.001(0.001)	0.00(0.00)
0.4	0.08(0.0)	-0.04(0.0)	0.001(0.001)	-0.00(0.00)
0.6	0.13(0.0)	-0.05(0.0)	0.001(0.001)	-0.00(0.00)
0.8	0.08(0.0)	-0.04(0.0)	0.02(0.03)	-0.01(0.01)
1	0.12(0.0)	-0.05(0.0)	0.12(0.07)	-0.03(0.0)
1.2	0.13(0.0)	-0.04(0.0)	0.17(0.0)	-0.003(0.0)
2.2	0.21(0.0)	0.02(0.0)	0.19(0.0)	0.02(0.0)
4	0.27(0.0)	0.18(0.0)	0.41(0.0)	0.47(0.0)
10	0.15(0.0)	0.19(0.0)	0.54(0.0)	0.66(0.0)
15	0.13(0.0)	0.18(0.0)	0.55(0.0)	0.67(0.0)
20	0.13(0.0)	0.18(0.0)	0.54(0.0)	0.66(0.0)
30	0.13(0.0)	0.17(0.0)	0.54(0.0)	0.66(0.0)
50	0.13(0.0)	0.17(0.0)	0.54(0.0)	0.66(0.0)
FCM	0.13(0.0)	0.17(0.0)	0.55(0.0)	0.67(0.0)



(a) $\sigma = 0.6$



(b) $\sigma = 2.2$

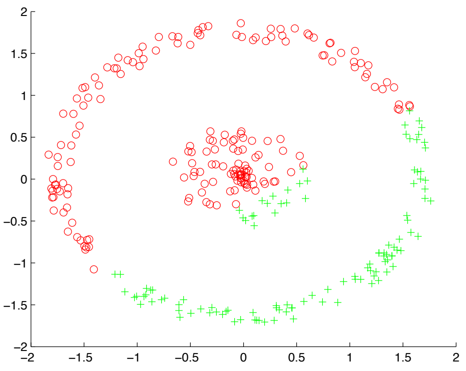
Figure 4.14: Clustering of the Iris datasets by KFCM-K for different σ, m=1.2 (Symbols represent the class label and the colours represent the cluster label; x-axis represents feature-3 and y-axis represents feature-4).

ticular, for the RBF kernel, it is necessary to choose σ, the spread of the kernel. It has also been observed that the choice of the kernel parameter significantly affects the partition produced by the clustering algorithm. In particular, for the RBF kernel, we have seen that except for Ring and Five-Cluster (with $c = 2$), a higher value of σ produces more acceptable results - results that agree with our expectation. Such observations can be made only for data in 2-D or 3-D. Based on the Sammon's projection, it appears that the topology of the projected data changes with the choice of the kernel parameter. This change of topology could sometimes be very useful to obtain the desirable partition (as in case of Ring with $\sigma = 0.6$, say) and sometimes it may impose structures definitely not existing in the actual data (for example, see Figure 4.4a and Figure 4.12c). But we do not know how the topology of the data is related to the choice of the Kernel parameter and what choice of parameter will produce the desirable partition of the data. Note that, choice of algorithmic parameters such as initialization, the value of m, may also produce an undesirable partition. The reference here is not to these cases, but rather to the unwanted partitions that arise due to the imposition of an unnatural structure on the data through kernelization. For example, it is difficult to imagine any choice of initialization or parameter for the FCM algorithm that could produce partitions like those shown in Figure 4.4a and Figure 4.12c.

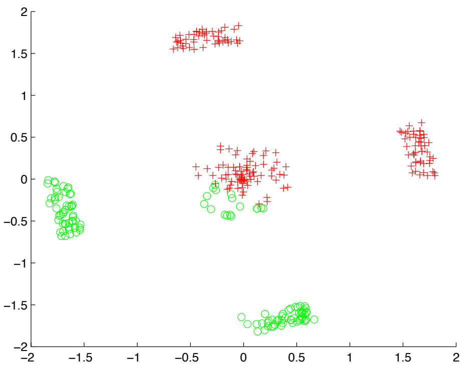
Then how can we choose the most desirable values of the kernel parameter? An answer to this question has not yet been found. However, a few points relevant specifically to the RBF kernel are discussed below. Based on the results from this limited dataset, the following remarks are offered: (a) Use of higher values of σ (say more than 1) may help to preserve the original structure of the data (to be further elaborate later). (b) Higher values of σ are likely to produce partitions similar to that by FCM; hence, there may not be any gain from Kernel clustering. (c) Lower values of σ may change the structure of the data and that may help to obtain the desired partition of the data. (d) But the desired partition may be known only for 2D data or synthetic data!

The following section attempts to justify these remarks. Based on the KFCM-K clustering results in Figure 4.12a, for the Ring dataset with $\sigma = 1$, we expect the ring and the disk to form two separate clusters in the kernel space. The Sammon's projection (Figure 4.12b) suggests that it is indeed the case although the original structure of the shell is changed. This is a case where original structure is not preserved completely but it has helped to generate the most desirable partition. On the other hand, although the KFCM-F algorithm can find the shell as a separate cluster (Figure 4.5 for $\sigma = 0.2$), the Sammon projection is not as good as that for $\sigma = 1$. However, in this case the points corresponding to the shell do appear in a compact area. This shows occasional success of KFCM-F to produce the desirable results. On the other hand, in Figure 4.12g and Figure 4.12h, It is observed that a low σ value (e.g., $\sigma = 0.2$) imposes an undesirable structure on the data, resulting in an undesirable partition.

Table 4.1 summarizes the visual assessment of the clusters generated by KFCM-K across different datasets with varying choices of σ. From this table, it is observed that, except for the Ring and Five Cluster datasets, higher values of σ yield desirable partitions. Although, for Ring and Five Cluster KFCM-K does not produce the desirable partition, it does produce FCM type partition (see Figure 4.15). So this observation is also consistent with remarks (a) and (b).



(a) $\sigma = 3$



(b) $\sigma = 3$

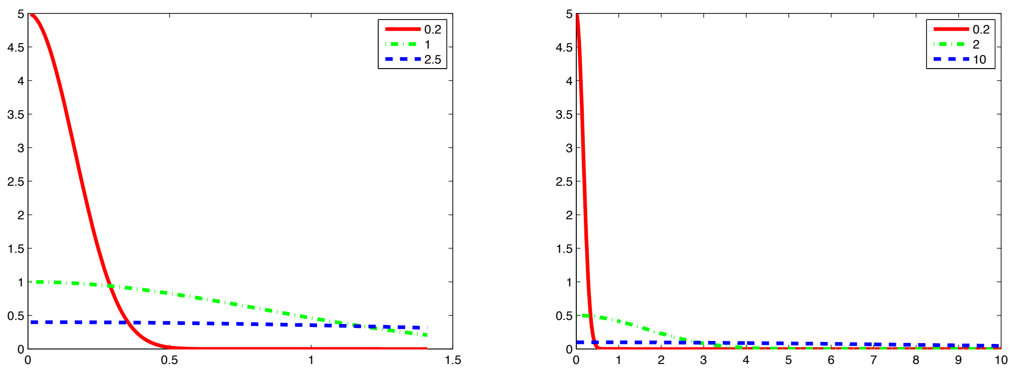
Figure 4.15: FCM type result using KFCM-K at higher σ.

To get a better understanding of the dependency on the kernel, we analyze the relation between $d^2 = ||\mathbf{x} - \mathbf{y}||^2$ and $kd = \sqrt{1 - e^{-||\mathbf{x}-\mathbf{y}||^2/\sigma^2}} = \sqrt{1 - e^{-d^2/\sigma^2}}$ (the constant $\sqrt{2}$ is dropped without any loss). Differentiating kd with respect to d we get $kd' = \frac{d}{\sigma^2} \frac{e^{-d^2/\sigma^2}}{\sqrt{1-e^{-d^2/\sigma^2}}} > 0$. Thus if the distance between a pair points in the original feature space increases then the corresponding distance between the same pair of points will also increase. This may give an impression that the structure of the data will be preserved in the kernel space irrespective of the choice of σ . It has been demonstrated that this is not true! The reason becomes clear upon examining Figure 4.16a. In Figure 4.16a we plot the variation of kd' with d for different values of σ. In Figure 4.16a, d is varied between 0 to 1.4142 as our data are normalized between -1 and $+1$ and the datasets are in 2-D. Figure 4.16a shows that when σ is low (for example, $\sigma = 0.2$, if d goes beyond 0.6, practically there is no change in kd and hence geometry of the points which are far apart would be difficult to preserve at least *numerically*. When σ becomes 1, the picture changes drastically because over the entire domain of d, a change in d is noticeably reflected in kd; while for $\sigma = 2.5$ kd varies almost linearly with d. Thus when $\sigma = 2.5$, the "structure" of the original data and that projected data are likely to be similar in terms of inter-point distances. For $\sigma = 10$ the derivative of kernel distance is practically constant, thus for higher σ, kd practically changes linearly with d.

If the domain of the features are bigger, then this will demand higher values of σ for better representation. For example, let us assume that the data points lies within a square of size 7×7. Then the inter-point distance varies between 0 to $\sqrt{(98)} = 9.9$. In this case, even with $\sigma = 2.0$ kd is not sensitive for $d > 5$. However, $\sigma = 10$, for example, would be a good choice (see Figure 4.16b).

A very high σ is also expected to have an adverse effect because higher σ will shrink the smallest hyper-box containing the projected data because inter-point distance in the projected space decreases as σ increase (derivative of kd with respect to σ is negative and kd goes to zero as σ goes to ∞). It must be noted that, for all these datasets, no adverse effects are detected on the crisp partitions obtained after hardening. However, as will be shown later, there is a significant effect on the fuzzy partitions obtained with higher σ values. Here also the practical (numerical) tolerance to σ depends on the domain of the input data. This fact is also revealed by Sammon's projection.

For the normalized Ring data, we generate the kernel matrix for different

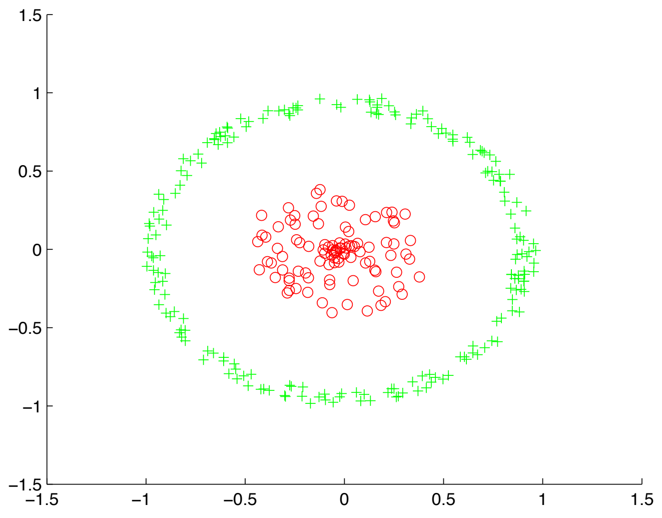


(a) (b)

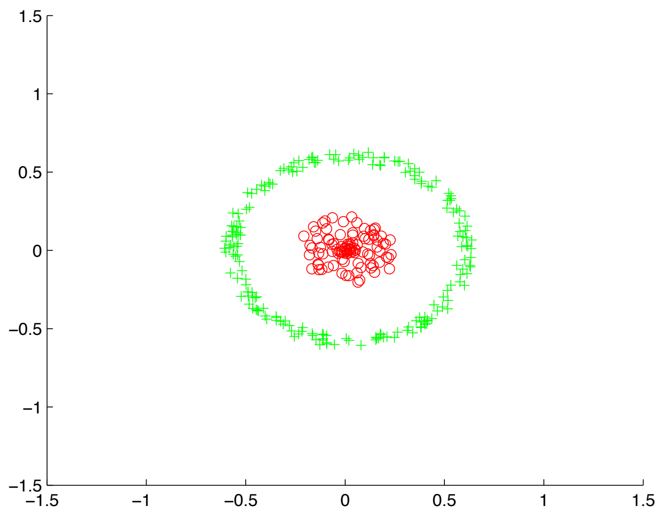
Figure 4.16: (a) Variation of the derivative of kd as a function of d for $\sigma = 0.2, 1.0, 2.5$ when d varies in [0, 1.414], i.e., data are within a unit square; (b) variation of the derivative of kd as a function of d for $\sigma = 0.2, 2.0, 10.0$ when d varies in [0, 9.9], i.e., when data are within a square of side 7 units (x-axis represents d and y-axis represents $kd^{'}$).

values of σ varying from 0.1 to 5.0. In each case, the height and width of the smallest rectangle containing the projected data, as computed by Sammon's algorithm, were determined. The area of this smallest rectangle is included in Table 4.7. Table 4.7 shows that the area shrinks with σ. In Figure 4.17a we show the scatterplot of the projected data by Sammon's algorithm for $\sigma = 1.7$ while Figure 4.17b depicts the same for $\sigma = 3.7$. If the data are not normalized, even then the same thing happens, but to see noticeable changes in the area of the smallest rectangle, we need to use higher values of σ as shown in columns 3 and 4 of Table 4.7. For the ring data Table 4.1 shows that one can achieve desired results for σ roughly between 0.4 and 1.4. Similarly, for the Five-cluster data usually up to about $\sigma = 1.0$ we get good clusters, but beyond that, the desirable partition is usually not found. In fact, from the NMI and ARI values in Table 4.1 we find that for some of the runs, KFCM-K could not extract the desired partition. For Bar, Disk and Uniform Disk an increase in σ does not cause any problem. This is possibly due to the fact that ever after shrinking the shape of the data remains the same, hence does not cause any problem. Note that, all these limit on the values of

σ that we mention in Table 4.1 are approximate values and usually for σ around those values the algorithm yields the type of partitions that are qualitatively of the type indicated in the column-3 of the Table 4.1.



(a) $\sigma = 1.7$



(b) $\sigma = 3.7$

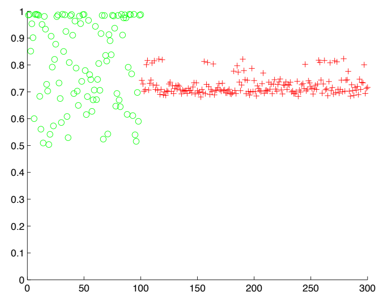
Figure 4.17: Shrinkage of areas of the smallest rectangles that contain the projected data computed by Sammon's algorithm for different σ.

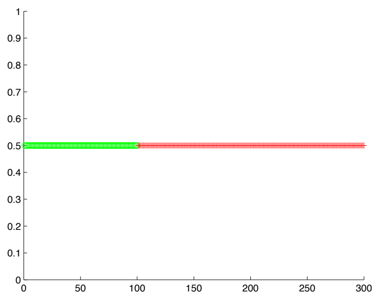
Table 4.7: Area of the smallest rectangle containing the projected data as computed by Sammon's projection for different choices for σ for the Ring dataset.

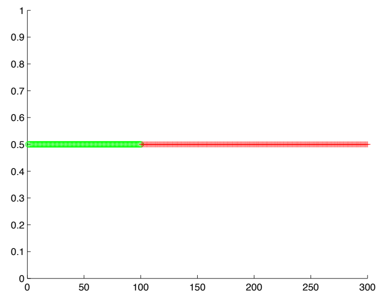
σ	Area associated with normalized data	σ	Area associated with unnormalized data
0.1	4.80	1.0	4.73
0.5	4.79	2.0	4.71
0.9	4.589	5	4.27
1.3	4.049	6	4.25
1.7	3.82	7	3.95
2.1	3.22	8	3.61
2.5	2.67	9	3.27
2.9	2.20	10	2.94
3.3	1.83	11	2.63
3.7	1.53	12	2.36
4.1	1.30	13	2.11
4.5	1.11	14	1.90
4.9	0.95	15	1.71

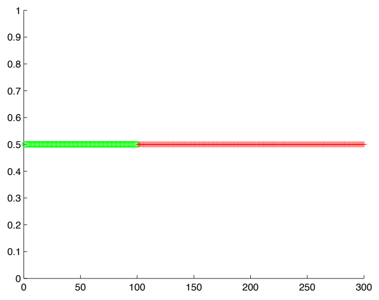
4.8 Effect of the fuzzifier (m)

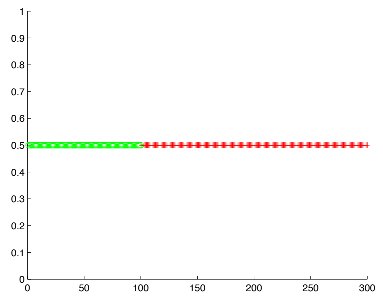
In our investigation, we have used $m = 1.2$. One would expect that with such a choice of m the partition would be almost crisp for all choices of σ. But this is not exactly true. For example, in case of Ring data with $\sigma = 1.5$ and $m = 1.2$, although for most of the points the highest membership value is close to one (1.0), there are several points whose membership values are close to 0.5. This picture changes drastically, when we consider $\sigma = 0.4$. In this case for most of the points the maximum membership values are quite far from the crisp value of unity. Moreover as the value of m increases the partition becomes more and more fuzzy (see Figures 4.19 and 4.18). In Figures 4.19 and 4.18 we plot the maximum membership values. If a point belongs to the outer shell it is denoted by a circle (o) and colored green while points belonging to the disk are represented by a plus (+) and colored red. Comparing the partitions obtained with different m for $\sigma = 0.4$ and $\sigma = 1.5$ (Figure 4.19 and Figure


(a) $m = 1.2$


(b) $m = 1.5$


(c) $m = 1.8$


(d) $m = 2.0$


(e) $m = 2.2$

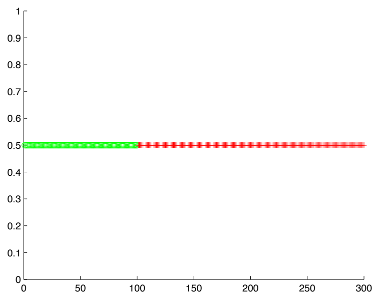

(f) $m = 2.3$

Figure 4.18: Effect of the fuzzifier (m) on membership values with $\sigma = 0.4$ for Ring data (x-axis represents the data points and y-axis represents membership values).

4.18) we find that corresponding partitions with lower σ are more fuzzy. There is one more surprising point that is revealed by Figure 4.19 and Figure 4.18: In these set of results the maximum membership values for the points in the disk are usually higher than the maximum membership values for the points in the outer shell. In other words, it suggests that in the Kernel space, the central disk is consistently represented better by its associated prototype compared to the representation of the outer shell by its prototype. It appears that the interaction between m, σ and u_{ik} is highly nonlinear and is difficult to visualize. However, one point is clear that for a given m, the choice of σ can also significantly influence the fuzziness in the partition and only with a lower value of m it may be possible to have crisper partitions for different choices of σ. Hence we have used $m = 1.2$ in all our experiments to avoid undesirable effect of improper choice of σ.

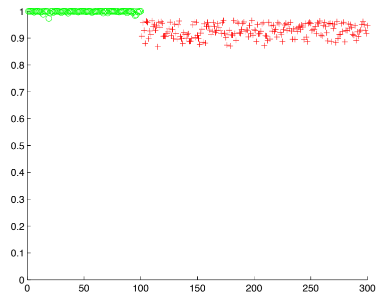
To further understand the interaction between fuzziness in the partition, and m and σ, we compute the partition entropy. The entropy of a fuzzy partition matrix $U = [u_{ik}]_{c\times n}$ is defined as [31]

$$PE(U) = -\frac{1}{n \log_2 c} \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik} \log_2 u_{ik}. \tag{4.15}$$

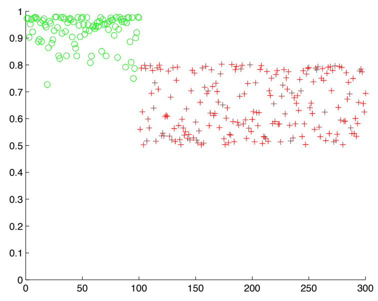
The PE attains the maximum value of 1 when the partition is most fuzzy, i.e., $u_{ik} = 1/c \ \forall i, k$. On the other hand, PE takes the minimum value of zero, when the partition is crisp; i.e., $u_{ik} = 0$ of $1 \ \forall i, k$. In Table 4.8 we summarize the entropy values for the Ring data for different combinations σ and m over 30 runs. Table 4.8 suggests that when σ is changed from 2.5 to 0.1, the fuzziness rapidly increases with m. With $\sigma = 0.1$, practically the maximum fuzziness is attained with $m = 1.2$, while with $\sigma = 2.5$ such a situation does not happen even for $m = 2.3$.

Table 4.8: Mean (Standard Deviation) of Partition Entropy for different values of fuzzifier m with four different values of σ for Ring data. The reported values are for 30 runs.

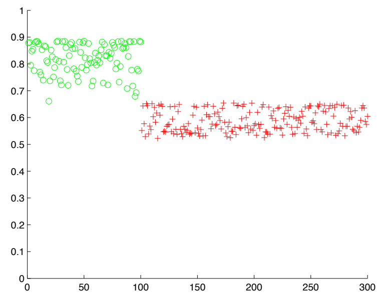
	m=1.2	m=1.5	m=1.8	m=2.0	m=2.2	m=2.3
$\sigma = 0.1$	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)
$\sigma = 0.4$	0.75 (0.0)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)
$\sigma = 1.5$	0.23 (0.03)	0.71 (0.0)	0.87 (0.0)	0.93 (0.0)	0.97 (0.0)	0.99 (0.0)
$\sigma = 2.5$	0.22 (0.0)	0.59 (0.0)	0.81 (0.0)	0.89 (0.0)	0.94 (0.0)	0.96 (0.0)



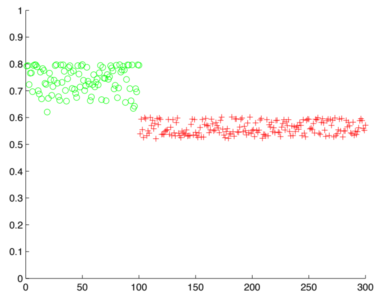
(a) $m = 1.2$



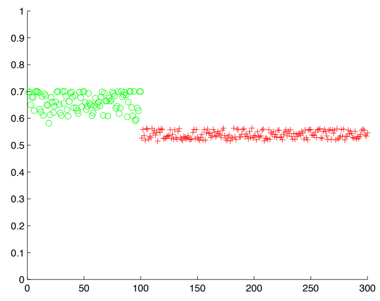
(b) $m = 1.5$



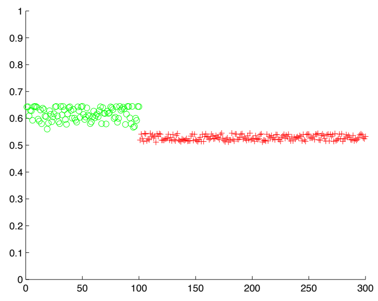
(c) $m = 1.8$



(d) $m = 2.0$



(e) $m = 2.2$



(f) $m = 2.3$

Figure 4.19: Effect of fuzzifier (m) on membership values with $\sigma = 1.5$ for Ring data (x-axis represents the data points and y-axis represents membership values).

4.9 Conclusion

A fundamental question has been raised and analyzed: Should we cluster in the kernel space? Can it help? If yes, when? These are philosophical issues and are not tied to any particular clustering algorithm or kernel. It has been demonstrated and justified that kernel clustering may lead to unexpected results. When the goal of clustering a dataset is to find the clusters / substructures in the original data (feature) space, unless one is careful, kernelization may impose undesirable structures on the data and hence the clusters obtained in the kernel space may not exhibit the structure of the original data. Note that, this is different from poor clustering results obtained due to local minimum problem or wrong choice of algorithmic parameters. In particular, for the Gaussian kernel, it has been justified that higher values of σ are likely to result in FCM-type partitions when using KFCM-K. Lower values of σ may produce partitions different from the FCM generated partition, but for high-dimensional data there is no easy way to assess whether the produced partition is the desirable one or not. One can argue that cluster validation indexes or measures of instability can be used to choose appropriate values of kernel parameters. We differ in this perspective because, in cluster validity, a model of the cluster exists in the original data space, and the assessment focuses on how well the obtained clusters fit this model. But here clustering is done on a transformed data, where the transformation could change the structure of the original data. Stability of partitions is also not necessarily useful because kernel transformation may impose a new but stable structure, which could be different from the substructures present in the original data. Thus, although for some choice of kernel and kernel parameters, kernel clustering may produce partitions that a user wants to have, there is no easy way to assess whether kernel clustering indeed has yielded such a partition. In our view there may be two ways to make kernel clustering useful: (i) incorporating domain knowledge (knowledge of what we want to find) into the kernel function and (ii) developing appropriate cluster-validation techniques. In this context, graph based cluster validity could be useful. For example, if a partition obtained by kernel clustering indeed extracts the "desired" cluster structure present in the original data, then two things are expected to happen. First, a minimal spanning tree (MST) obtained using the kernel transformed data should have a structure similar to that of an MST obtained using the original data. Second, in both MSTs there will be more edges within each cluster and few edges (ideally one edge) between every pair of clusters. However, a detailed investigation is needed to demonstrate the utility of such concepts.

In the next chapter (Chapter 5), an attempt is made to explore the application and potential use of FCM with spatial information for non-image data.

Chapter 5

Fuzzy Clustering Exploiting Neighbourhood Information for Non-image Data

5.1 Introduction

In the expansive landscape of clustering methodologies, Fuzzy C Means (FCM) has garnered widespread acceptance, offering a versatile approach to data partitioning [31]. Researchers have routinely leveraged FCM for clustering tasks, showcasing its effectiveness in various applications. Notably, in the domain of image segmentation, the literature abounds with studies employing FCM, with a particular emphasis on exploiting neighborhood pixel information to enhance segmentation accuracy [74, 75]. However, a conspicuous research gap becomes evident – while FCM with spatial information has been adeptly applied in image-based studies, its application to non-image datasets remains largely unexplored. In this chapter, local and contextual information is strategically incorporated, demonstrating enhanced clustering performance compared to conventional FCM.

The K-means clustering method is known for its simplicity, but it imposes a strict assignment of each image pixel to only one group. In contrast, Fuzzy C-Means (FCM) introduces membership degrees, allowing pixels to belong to multiple clusters simultaneously, determined by their membership degrees [76]. While FCM has proven significant in various applications, it has shortcomings, such as the lack of consideration for spatial context in images, making it susceptible to noise and imaging artifacts like intensity inhomogeneity. Additionally, FCM may converge to local optima due to poor initialization [77]. Consequently, modifications have been proposed to enhance its robustness for image segmentation.

In [78], they introduced Robust FCM (RFCM), incorporating a spatial penalty term into the objective function. However, RFCM's objective function exhibits complex variations in the membership function. Another approach, spatial FCM (SFCM) [79], adjusts the membership function by integrating spatial information, showing improved performance but remaining sensitive to serious noise. Fast Generalized Fuzzy C-Means (FGFCM) was proposed in [80], where they used spatial information for brain MRI segmentation, yet its performance degrades with significant noise. In [81], they maximized fuzzy partition entropy with 2D histogram for MRI segmentation but suffered from high time complexity. Modified versions of FCM have been proposed, including non-local FCM (NL/FCM) [82] and FCM with non-local spatial information [83], both addressing robustness against noise and inhomogeneity but showing limitations under high noise levels.

In [84], they developed an updated FCM approach by modifying the objective function to include a gray-difference coefficient, aiming to enhance performance against noise. These adaptations demonstrate ongoing efforts to address the limitations of FCM in image segmentation, with each method presenting its advantages and trade-offs.

The prevailing trend in the literature underscores a concentration of efforts towards leveraging FCM for image segmentation, where the incorporation of spatial information is pivotal for capturing contextual relationships among neighboring pixels. Yet, the omission of FCM with spatial information from non-image data analysis raises questions about the algorithm's untapped potential beyond the visual domain.

This study attempts to address this noteworthy gap in the literature. In this work, the aim is to bridge the divide by introducing FCM with spatial information to the realm of non-image data, thereby propelling the algorithm into uncharted territories. By extending FCM's application beyond the confines of image analysis, our research endeavors to shed light on its adaptability and efficacy in discerning cluster structures within diverse datasets.

The novelty of our work lies in the exploration of spatial information as a guiding principle for cluster formation in non-image datasets. Recent studies [166,167] have suggested that spatial relationships, albeit abstracted from pixel neighborhoods, play a crucial role in understanding the inherent structure of diverse data types. This departure from conventional FCM applications opens avenues for uncovering hidden patterns and relationships that may have been overlooked in conventional clustering approaches.

As we embark on this exploration, the potential impact of our work extends beyond a mere methodological innovation. The introduction of FCM with spatial information to non-image data analysis not only broadens the algorithm's applicability but also holds promise for unveiling nuanced cluster structures in datasets.

5.2 FCM with spatial information

The standard FCM objective function [31] to cluster a dataset $\{\mathbf{x}_k\}_{k=1}^{n} \subset \mathbb{R}^d$ is represented as

$$J_m = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} ||\mathbf{x}_k - \mathbf{v}_i||^2 \tag{5.1}$$

where c is the number of clusters and n is the number of data points. $|| \cdot ||$ stands for the Euclidean norm. The function is minimized under the following constraints:

- **C1** : $u_{ik} \epsilon [0, 1] \quad \forall i, k$
- **C2** : $\sum_{i=1}^{c} u_{ik} = 1 \quad \forall k$
- **C3** : $0 < \sum_{k=1}^{n} u_{ik} < n \quad \forall i$

In Equation 5.1, J_m provides a good result when the clusters are well grouped and separated from one another. Consider a dataset where the clusters are not well separated , rather one cluster is gradually approaching to other (Figure 5.1). Figure 5.2 displays the results for two clusters with $m = 1.5$. It is noted that FCM could not find the natural cluster structure. So the proximity of the data is not preserved in FCM. So, why FCM fails here? The reason is, FCM works on the principle of sum of squared error where the distance of each sample is measured from the center of each clusters. The points that are closer to a cluster center, their membership value for that cluster is higher. So, from the Figure 5.2, it is clear that the misclassified data points (marked in red in the green cluster) are much closer to the cluster center marked in red compared to the green cluster. Therefore, the FCM algorithm could not preserve the proximity of the data points in a cluster. This situation frequently arises because of the presence of outliers and brings up the problem of interpreting and evaluating the results of clustering.

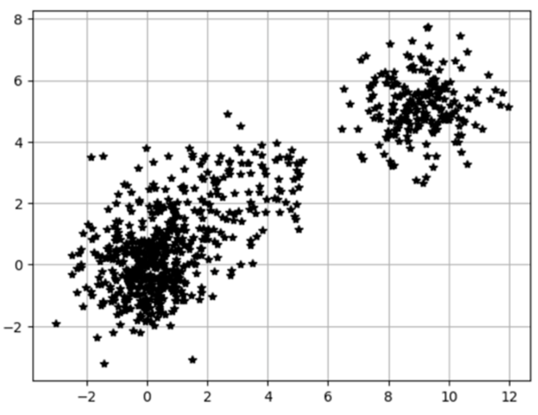


Figure 5.1: Synthetic dataset

In [168], a technique is proposed to enhance the robustness of the standard Fuzzy C-Means (FCM) algorithm by adding a penalty term with the objective function of FCM as below:

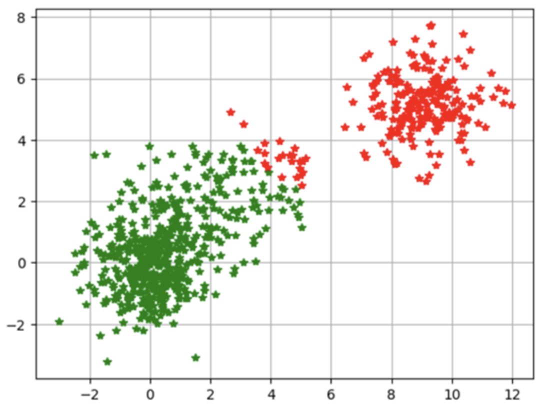


Figure 5.2: Clustering the synthetic dataset using FCM (c=2, m=1.5)

$$J_m = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^m ||\mathbf{x}_k - \mathbf{v}_i||^2 + \frac{\alpha}{N_R}\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^m \sum_{r \epsilon N_k} ||\mathbf{x}_r - \mathbf{v}_i||^2 \tag{5.2}$$

where N_k is the set of neighbours of x_k and N_R is its cardinality. The effect of the penalty term is controlled by the parameter α. The penalty term is added to exploit the spatial constraint so that the continuity on neighbouring pixels around x_k is maintained. The objective function can be minimized under the same constraints as standard Fuzzy C-Means. Note that, in [168], they have used it to segment image data, as in image we have the advantage to get the neighbouring information in terms of pixels. In [168, 169], they have used $||\mathbf{x}_r - \mathbf{v}_i||^2$ to measure the spatial distance in an image. In this study, the Euclidean distance is used to capture the spatial information. In Figure 5.1, although pixel information is not available, the model proposed in [168] is still used to exploit the spatial information.

Using the Lagrange multiplier technique, the membership value, u_{ik} of the k-th data point to the i-th cluster can be obtained as:

$$u_{ik} = \frac{(||\mathbf{x_k} - \mathbf{v_i}||^2 + \frac{\alpha}{N_R}\sum_{r\epsilon N_k} ||\mathbf{x_r} - \mathbf{v_r}||^2)^{-\frac{1}{(m-1)}}}{\sum_{j=1}^{c}(||\mathbf{x_k} - \mathbf{v_j}||^2 + \frac{\alpha}{N_R}\sum_{r\epsilon N_k} ||\mathbf{x_r} - \mathbf{v_j}||^2)^{-\frac{1}{(m-1)}}} \tag{5.3}$$

Equation 5.3 is derived using the first order necessary condition of optimality with respect to U, i.e., by setting the first order derivative of Equation 5.2 with respect to zero.

The expression of the prototype can be shown as

$$\mathbf{v_i} = \frac{\sum_{k=1}^{n} u_{ik}^{m}(\mathbf{x_k} + \frac{\alpha}{N_R}\sum_{r\epsilon N_k}\mathbf{x_r})}{(1+\alpha)\sum_{k=1}^{n} u_{ik}^{n}} \tag{5.4}$$

Equation 5.4 is obtained from the first order necessary condition of optimality of J_m with respect to $\mathbf{v}$.

We call this algorithm as FCMS (FCM-Spatial) and the corresponding algorithm is shown in Algorithm 5.1.

Algorithm 5.1 Algorithm of FCMS

```
(a) Fix $c$ (no. of clusters), $t_{max}$ (no. of iteration), $m > 1$ and $\epsilon > 0$, a tolerance
    level for termination.
(b) Initialize the membership value, $u$
(c) For $t = 1, 2, ..., t_{max}$ do
        Update $\mathbf{v}^{t+1} = [v_{ik}^{t+1}]_{c\times d}$ according to equation (5.4)
        Update $U = [u_{ik}]_{c\times n}$ according to equation (5.3)
    until
        $$\sum_{i=1}^{c}\sum_{k=1}^{n}|u_{ik}^{(t+1)} - u_{ik}^{(t)}| < \epsilon \tag{5.5}$$
    is satisfied.
(d) Return U and V.
```

In the next section, both synthetic and real datasets are used to study the performance of the model. It is again emphasized that the model is applied to non-image data, utilizing the proximity of samples to assign cluster labels. As per our knowledge, the model is only used so far on image data to use the neighbourhood pixels information.

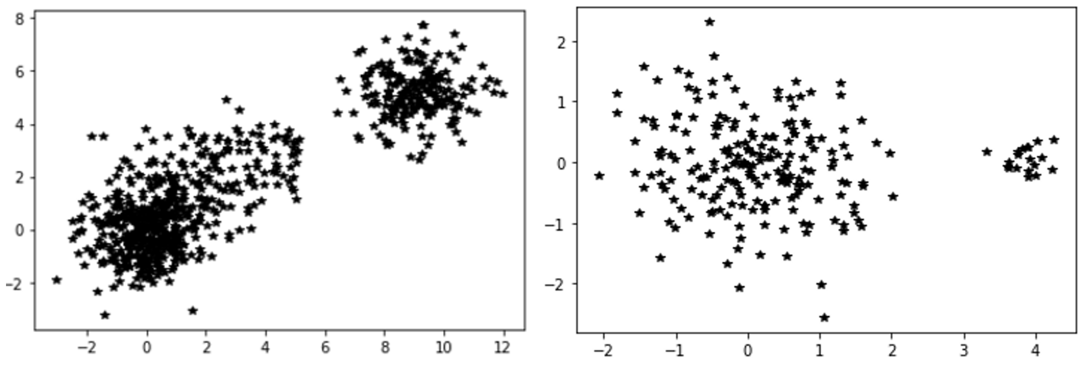
5.3 Results

Experiments are conducted on two synthetic datasets and four real datasets. First, two synthetic datasets are considered: Dataset-I (Figure 5.3a) and Dataset-II (Figure 5.3b), both in $\mathbb{R}^2$, as the ground truths for these datasets are known, allowing for visual assessment of the results. Finally, the performance is studied on four real datasets, as summarized in Table 5.5.

5.3.1 Studies on the synthetic datasets

To analyze the performance of the model (Equation 5.2), two synthetic datasets are first considered, as described below:

- **dataset-I** : Figure 5.3a represents the dataset-1, which contains two clusters out of which one is gradually approaching to the other. The bigger cluster contains 547 points and the smaller one contains 201 points.
- **dataset-II** : Figure 5.3b displays the dataset-II. The data points of each cluster is 200 and 20.



(a) dataset-I

(b) dataset-II

Figure 5.3: Synthetic datasets used in this study.

In all our clustering results, different clusters will be shown by different colours. The effects of FCM and FCMS are examined on Dataset-I and Dataset-II. For all examples we use $m = 1.5$ and 2, $\epsilon = 10^{-5}$, $t_{max} = 100$.

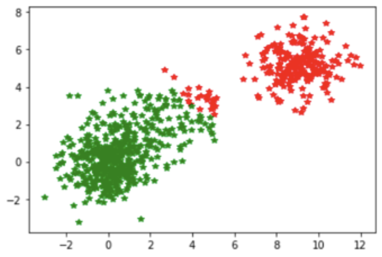
First we consider the dataset-I (Figure 5.3a). In the dataset two clusters are there and one cluster is gradually approaching to the other. The behavior of the FCM algorithm is first examined. The result is shown in Figure 5.2 for m=1.5. Note that FCM fails to find the natural partition. As discussed already, it is due to the fact that the proximity of a sample is not preserved in FCM. It is evident that points from the green cluster, which are close to the red cluster, are misclustered. This result has been tested across multiple runs and for different values of the fuzzifier (m).

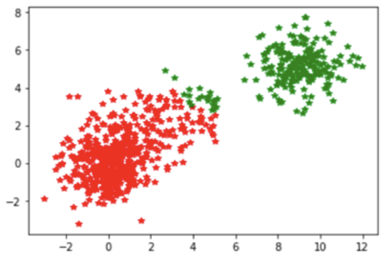
The FCMS algorithm is now applied to the same dataset. The behavior of FCMS is studied using the following parameters:

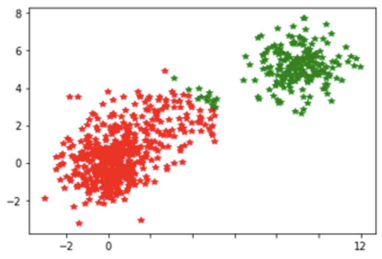
- $m = 1.5, 2$
- $\alpha = 2, 3$
- $N_k = 10, 15, 30, 50, 80,$

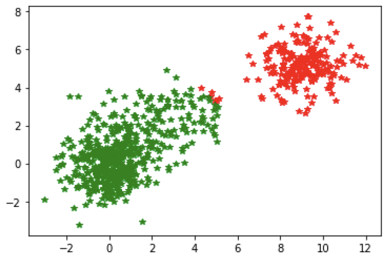
In Figure 5.4, we have shown the result of different clusters for $m = 1.5$, $\alpha = 2$ and $N_k = 10, 15, 30, 50, 80$. It is seen that, with the increase of the neighbours there is an improvement in the clustering result. If we compare Figure 5.4a with Figure 5.4e, it is easily shown that in Figure 5.4e the proximity of the data points is well preserved. The number of misclassification is reduced when $N_k = 80$. To quantitatively assess the clustering results, Normalized Mutual Information (NMI) [134] and Adjusted Rand Index (ARI) [135, 158] are used as cluster validity indices. Higher value of the index indicates good clustering result. Maximum value of both the index is 1. The results are summarized in Table 5.1. It is noted that both the index increases with the number of neighbours (N_k). It is observed that in spite of increasing the neighbour, we could not get the proper cluster as one point is still miss-clustered. To overcome that, we go for the hard clustering by setting $m = 1.01$. The corresponding result is shown in Figure 5.5 and the NMI and ARI in Table 5.1.

To check the sensitivity of the hyper-parameters, we have studied the effect of the neighbours on the Data Set-I (Figure 5.3a) for ($m = 1.5$ & $\alpha = 3$), ($m = 2$ & $\alpha = 2$) and ($m = 2$ & $\alpha = 3$). The corresponding results are shown in Figure 5.6, Figure 5.7 and Figure 5.8 respectively. The NMI


(a) $N_k = 10$


(b) $N_k = 15$


(c) $N_k = 30$


(d) $N_k = 50$

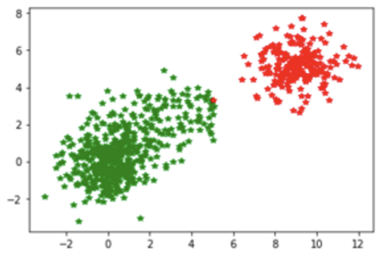

(e) $N_k = 80$

Figure 5.4: Clustering using FCMS with $c = 2$, $m = 1.5$ and $\alpha = 2$

Table 5.1: NMI & ARI of the clustering results produced by FCMS for different choices of the N_k. $c = 2$ and $\alpha = 2$.

m	N_k	NMI	ARI
1.5	5	0.85	0.91
1.5	10	0.87	0.92
1.5	15	0.87	0.92
1.5	30	0.88	0.93
1.5	50	0.97	0.98
1.5	80	0.98	0.99
1.01	50	1.00	1.00

and ARI of the above combinations are noted in Table 5.2, Table 5.3 and Table 5.4 respectively. In all the cases it is obvious that the proximity of the data is preserved while clustering.

In equation 5.1, J_m is an appropriate criterion when the clusters form compact clouds that are rather well separated from one another. A less obvious problem arises when there are great differences in the number of samples in different clusters. In that case it can happen that a partition that splits a large cluster is favoured over one that maintains the integrity of the natural clusters. To illustrate this issue, Data Set-2 (Figure 5.3b) is used. First, FCM is implemented with $m = 1.5$ and 2. The corresponding results are shown in Figure 5.9. It is clear that the result is not intuitive. In both the cases FCM fails to find the proper cluster.

The FCMS algorithm has now been implemented on Data Set-II. The result is shown in Figure 5.10a. It is noticed that for $m = 1.5$, $\alpha = 2$, and $N_k = 5$ (Figure 5.10a) and $m = 2$, $\alpha = 2$, and $N_k = 20$ (Figure 5.10b), FCMS produces intuitive result. It is seen that, for higher value of $'m'$, the number of neighbours is more to get the proper cluster.

5.3.2 Studies on Real datasets

Real datasets from the UCI Machine Learning Repository [132] are used in this study. Four datasets, listed in Table 5.5, are considered. The pa-

Table 5.2: NMI & ARI of the clustering results produced by FCMS for different choices of the N_k. $c = 2$, $m = 1.5$ and $\alpha = 3$.

N_k	**NMI**	**ARI**
5	0.85	0.91
10	0.86	0.92
15	0.85	0.92
30	0.87	0.92
50	0.95	0.98
80	0.97	0.99

Table 5.3: NMI & ARI of the clustering results produced by FCMS for different choices of the N_k. $c = 2$, $m = 2$ and $\alpha = 2$.

N_k	**NMI**	**ARI**
5	0.85	0.91
10	0.86	0.92
15	0.85	0.91
30	0.87	0.92
50	0.95	0.98
80	0.97	0.99

Table 5.4: NMI & ARI of the clustering results produced by FCMS for different choices of the N_k. $c = 2$, $m = 2$ and $\alpha = 3$.

$\mathbf{N_k}$	**NMI**	**ARI**
5	0.85	0.90
10	0.86	0.92
15	0.86	0.92
30	0.87	0.92
50	0.97	0.99
80	0.98	0.99

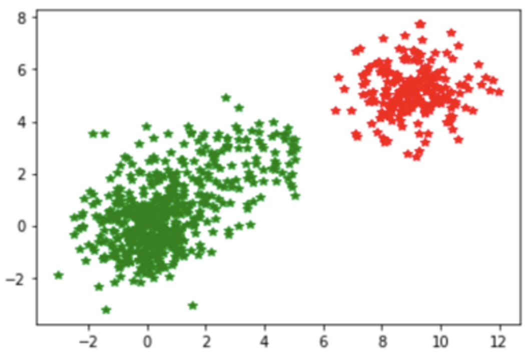


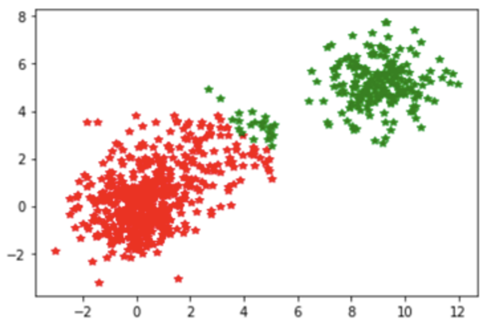
Figure 5.5: Clustering using FCMS with $c = 2$, $m = 1.01$, $\alpha = 2$ and $N_k = 50$.

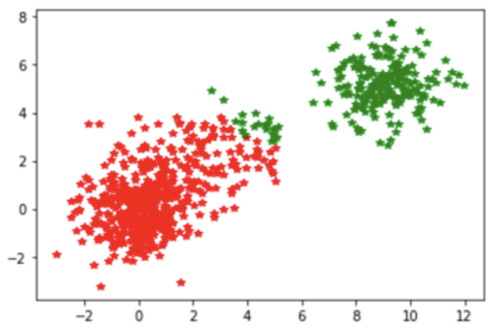
rameters used in the FCMS algorithm for each dataset are provided in Table 5.6. All the combinations of each parameter (m, α and N_k) as noted in Table 5.6 for each dataset is evaluated. To compare the result of FCM with FCMS, we cluster the data using FCM with $m = 1.5$ and $m = 2$. For better visualization, NMI values are reported for all combinations of the hyper-parameters, as shown in Table 5.6.

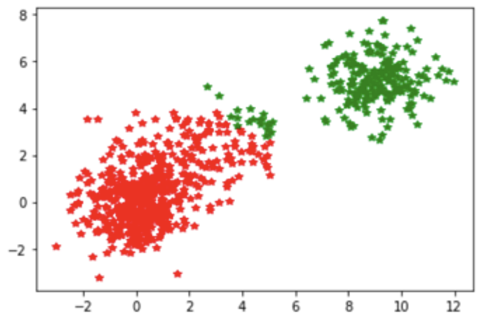
The corresponding value of NMI is shown in Figure 5.11. In each figure, last two bars indicate the NMI for FCM (one for $m = 1.5$ and other for $m = 2$) and the rest are for FCMS. The black horizontal line denotes the highest NMI among the two result of FCM, i.e, for $m = 1.5$ and $m = 2$. It is seen that in all the dataset, there are many cases, where FCMS outperforms over FCM.

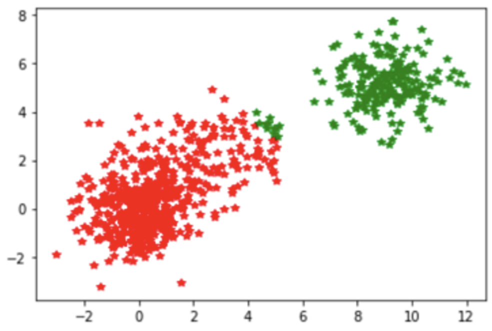
Table 5.5: Real dataset used in this study.

Dataset	**Instances**	**Features**	**Class**
IRIS	150	4	3
WINE	178	13	3
MUSK	476	166	2
SONAR	208	60	2


(a) $N_k = 10$


(b) $N_k = 15$


(c) $N_k = 30$


(d) $N_k = 50$

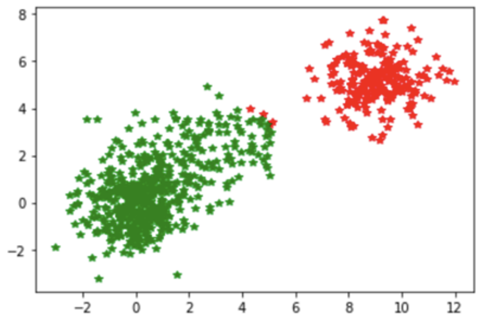
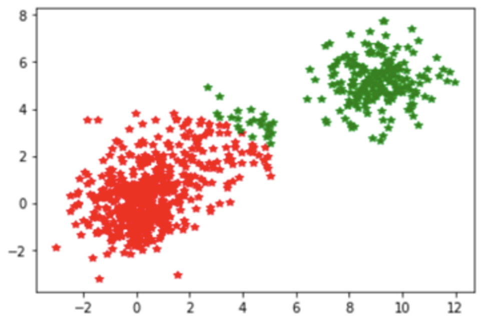
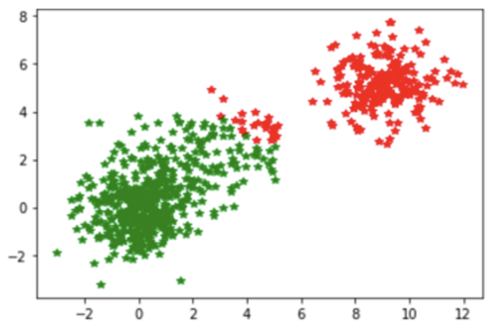

(e) $N_k = 80$

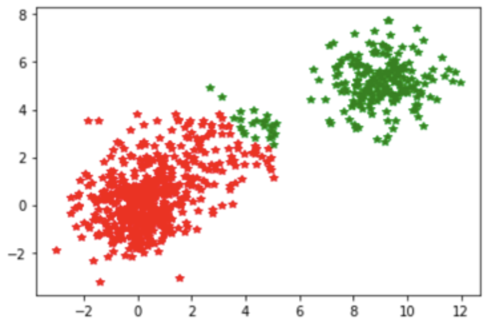
Figure 5.6: Clustering using FCMS with $c = 2$, $m = 1.5$ and $\alpha = 3$.



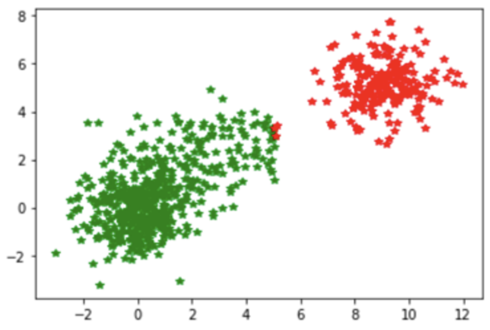
(a) $N_k = 10$



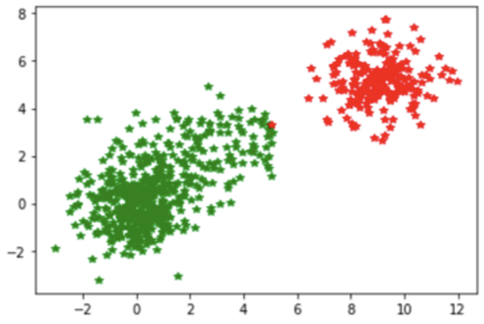
(b) $N_k = 15$



(c) $N_k = 30$

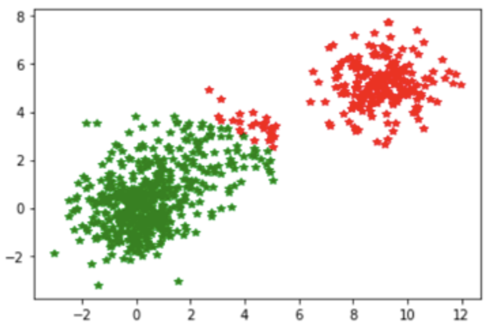


(d) $N_k = 50$

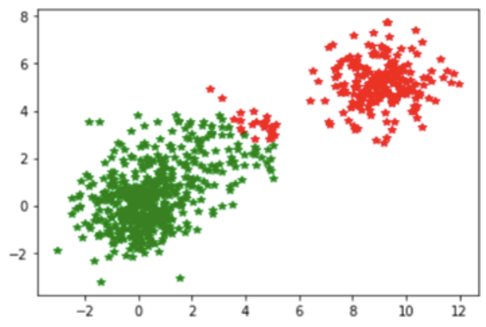


(e) $N_k = 80$

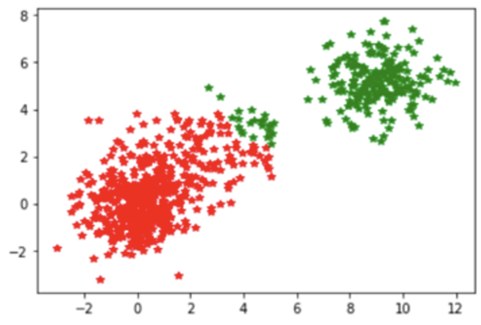
Figure 5.7: Clustering using FCMS with $c = 2$, $m = 2$ and $\alpha = 2$.



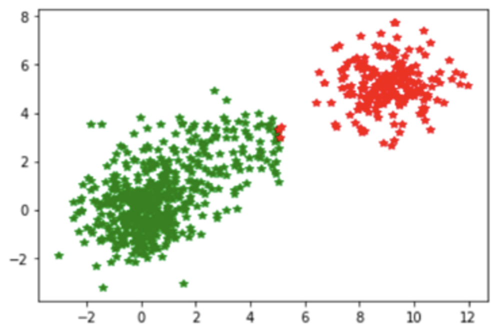
(a) $N_k = 10$



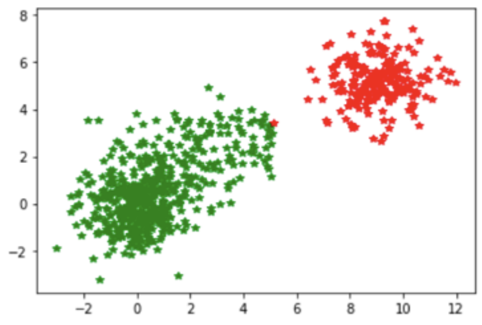
(b) $N_k = 15$



(c) $N_k = 30$

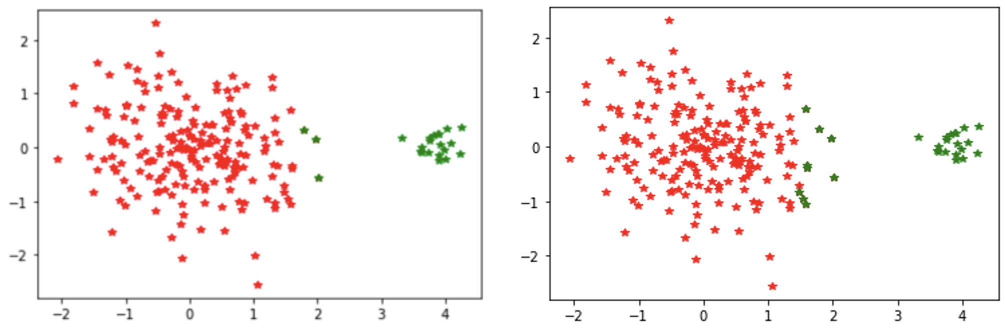


(d) $N_k = 50$



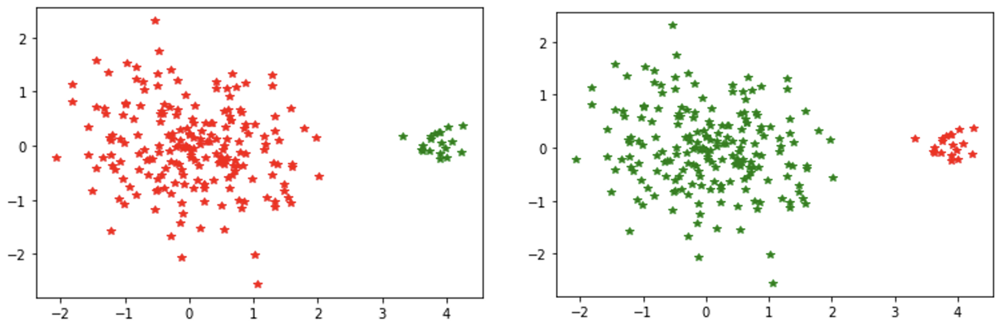
(e) $N_k = 80$

Figure 5.8: Clustering using FCMS with $c = 2$, $m = 2$ and $\alpha = 3$.



(a) $m = 1.5$ (b) $m = 2$

Figure 5.9: Applying FCM on Data Set-II with $c = 2$.



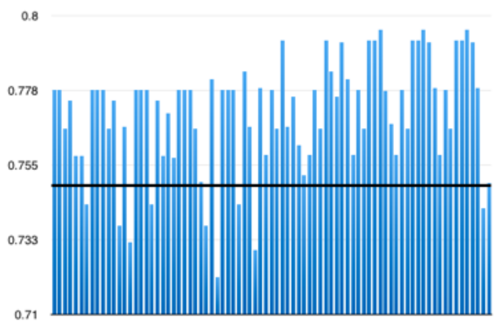
(a) $m = 1.5, \alpha = 2, N_k = 5$ (b) $m = 2, \alpha = 2, N_k = 20$

Figure 5.10: Applying FCMS on Data Set-II with $c = 2$.

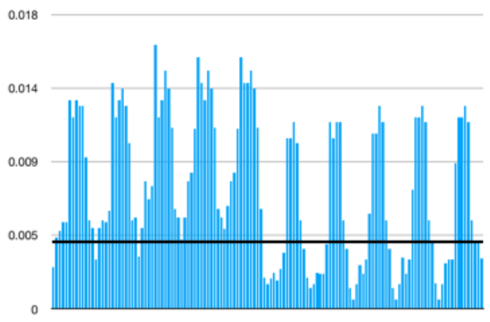
5.4 Conclusion

FCMS, an extended version of the Fuzzy C-Means (FCM) algorithm, emerges as a solution to the limitations encountered by conventional clustering methods in complex scenarios. While FCM excels in scenarios with well-defined, separated clusters, its performance falters when confronted with datasets exhibiting unclear cluster boundaries or spatial continuity. FCMS is specifically crafted to address this challenge by integrating spatial information, setting it apart from conventional FCM algorithms primarily applied to image data.

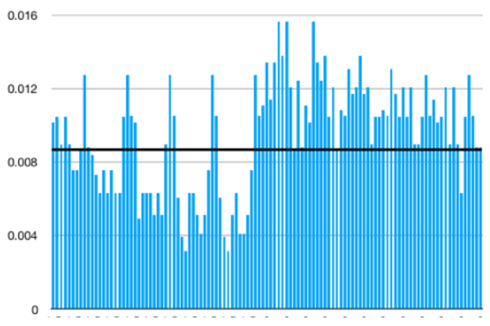
Typically, spatially constrained FCM algorithms find their application in image processing, considering spatial information in the proximity of pixels. However, FCMS extends this paradigm beyond the image data,



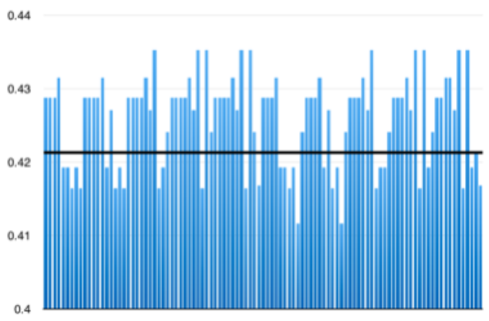
(a) IRIS



(b) MUSK



(c) SONAR



(d) WINE

Figure 5.11: Values of NMI for all combination as shown in Table 5.6. Black horizontal line indicates the NMI for FCM.

Table 5.6: Study on real datasets for $m = 1.5, 2$ and $\alpha = 2, 3, 5, 7, 10$ with different N_k.

Dataset	$\mathbf{N_k}$
IRIS	5, 10, 15, 20, 25, 30, 35, 40
WINE	5, 10, 15, 20, 25, 30, 35, 40, 50, 60
MUSK	5, 10, 15, 20, 30, 40, 50,70, 90, 120,
SONAR	150, 180, 200 5, 10, 15, 20, 25, 30, 35, 40, 50, 70, 90

showcasing its adaptability on non image datasets. The essence of spatial information proves pivotal, especially when dealing with datasets where natural clusters lack compactness and distinct separation.

The conventional FCM objective function relies on minimizing squared errors based on Euclidean distances between data points and cluster centers. However, FCMS introduces a novel dimension to the objective function—a penalty term. This addition facilitates the preservation of proximity among neighboring pixels or data points, particularly beneficial when handling datasets with less-defined cluster boundaries.

Our experiments, both on synthetic datasets and real-world scenarios, revealed FCMS's efficacy in diverse situations. Notably, in synthetic datasets featuring gradually converging clusters, FCMS outshines FCM by adeptly preserving proximity within clusters.

Parameter sensitivity emerges as a noteworthy aspect, with the fuzzifier (m), penalty parameter (α), and the number of neighbours (N_k) playing crucial roles in fine-tuning FCMS's performance. This adaptability ensures the algorithm's applicability across various dataset characteristics.

In summary, FCMS demonstrates a significant potential in elevating clustering performance, particularly in non-image datasets where conventional FCM methods may struggle. The incorporation of spatial information makes FCMS as a valuable asset in the realm of clustering analysis, with promising applications across diverse real-world scenarios. Notably, our future work aims to extend the concept of exploiting neighborhood information to manifold learning, potentially using geodesic distance—a testament to FCMS's evolving versatility.

In the final technical chapter (Chapter 6), a fuzzy rule-based method has been developed to identify synergistic pairs of genes associated with prostate cancer.

Chapter 6

Finding Synergy Networks from Gene Expression Data: A Fuzzy Rule Based Approach

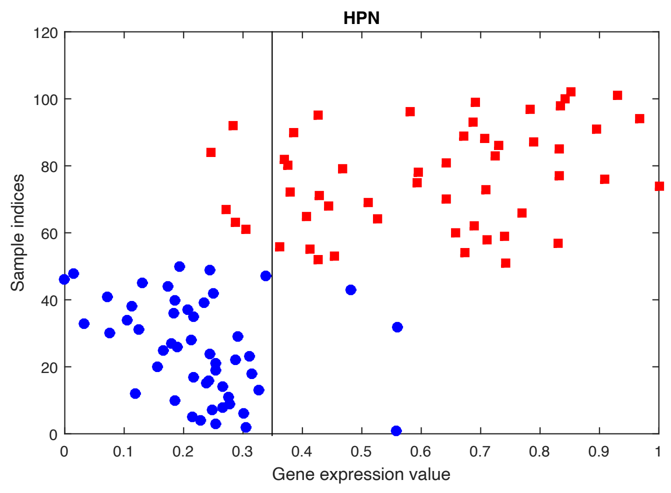
6.1 Introduction

Dimensionality reduction can be done through feature selection using a fuzzy rule-based framework as well as by feature extraction. If the problem is regression or classification, we can find the antecedent clauses of the initial rules by partitioning/clustering of the input data without using class-labels or the target output of the regression data. The initial rule antecedents can also be extracted separately clustering data for every class. However, rule labeling and refinement will need the supervisory information [170–172]. On the other hand, when dimensionality reduction is used for data visualization, it can be done in a completely unsupervised manner. As an example, for manifold learning there is no such supervisory information and it has to be and can be done in a completely unsupervised way [173].

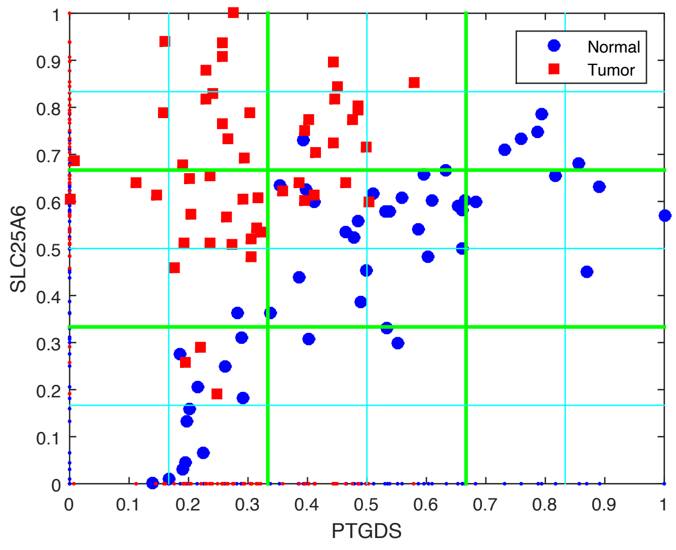
Usually, an objective function is used along with some penalty/regularizing terms to select features via fuzzy rules. When the target task is of classification, for feature selection using a fuzzy rule-based framework, typically, we take all the data points of each class, and as indicated earlier, we process it in an unsupervised way, i.e., use a clustering algorithm. Each cluster is then translated as the antecedent of a rule. It is needless to mention that generally clustering is an iterative process and computationally expensive. In this chapter, our objective is to find/select special type of features, specifically, pairs of features, where the features in a pair have a synergistic relation with respect to the classes. Thus, we shall see later, that finding rules to check synergistic association will involve clustering of training instances for each pair of features. In this chapter, we shall deal with a dataset having 12558 features (genes) and thus this translates to clustering of about 80 million pairs of genes to extract rules. This is computationally prohibitive. A novel, yet simple, scheme is proposed to overcome this issue. Our objective is to identify a special type of network called "synergy network" using the synergistic pairs of genes/features.

Genetic interactions are very important as they help to discover functional relationships among genes and hence have been studied extensively. Here we address the problem of identifying gene-gene interactions that are associated with a disease from a set of gene expression profiles on some diseased and normal subjects. Such interactions can be viewed as of collaborative nature where two genes cooperate with respect to a disease or phenotype. This kind of disease-related interactions can provide us with a better understanding of the biological processes and associated pathways responsible for the disease.

For analysis of microarray data, many methods [174–178] have been proposed. Out of which, Bayesian networks [175, 176], pairwise mutual information [177, 178] and Graphical Gaussian models [179, 180] are commonly used. An interesting application of Bayesian networks is proposed by Gevaert et al. [181] where both clinical data and microarray data are integrated via a Bayesian network. Authors then use their model to identify poor and good prognosis groups in publicly available breast cancer datasets. In this context, three different methods of information integration are proposed of which the partial integration method is found to



(a)



(b)

Figure 6.1: The scatter plots of one of the biomarker genes for prostate cancer and one synergistic pair identified using our Fuzzy Rule Based measure are shown in panel (a) and (b) respectively.

be the most promising one. This approach also finds the genes related to cancer but the resulting network may not reveal the synergistic relationship between those genes with respect to a disease. In our investigation, we try to discover an interaction network in which a pair of genes is con-

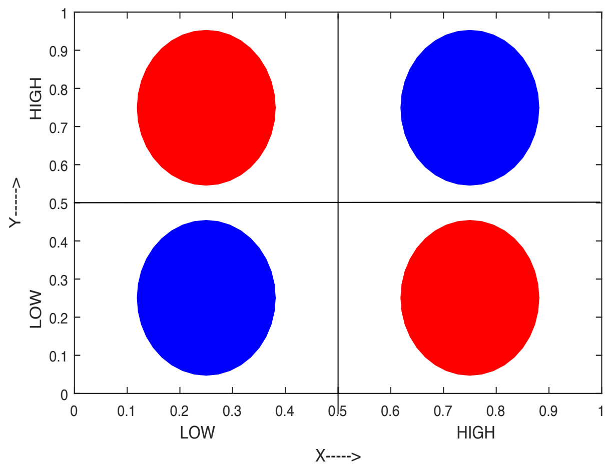


Figure 6.2: A hypothetical example of a synergistic gene pair that demonstrate a simple and interpretable logical relationship.

nected only when they interact synergistically with respect to cancer.

Now, the question is what is synergy and how do we quantify it? Consider, the scatterplot of the expression levels of HPN given in Figure 6.1a. As we can see, by taking a threshold on the expression levels of HPN (represented by the vertical line on the X-axis), we can separate the two classes of samples (tumorous and non-tumorous). From Figure 6.1a, we can say that when HPN is underexpressed the samples are non-tumorous, and tumorous otherwise. In other words, the expression levels of HPN demonstrate a causal linear relation with class. Therefore, HPN can be termed as a strong biomarker for cancer.

Now, let us consider another scenario. Looking at Figure 6.1b, we can see that, while the two genes, PTGDS and SLC25A6, are able to differentiate between the two classes jointly, none of them can do so separately. So, PTGDS and SLC25A6 are not biomarkers of cancer by themselves and yet they are able to act as a strong biomarker of prostate cancer jointly due to some form of collaborative interaction between them. Synergy is defined as a phenomenon where the combined effects of the components of a system are greater than the sum of their individual effects. Can we then describe the interaction between PTGDS and SLC25A6 in

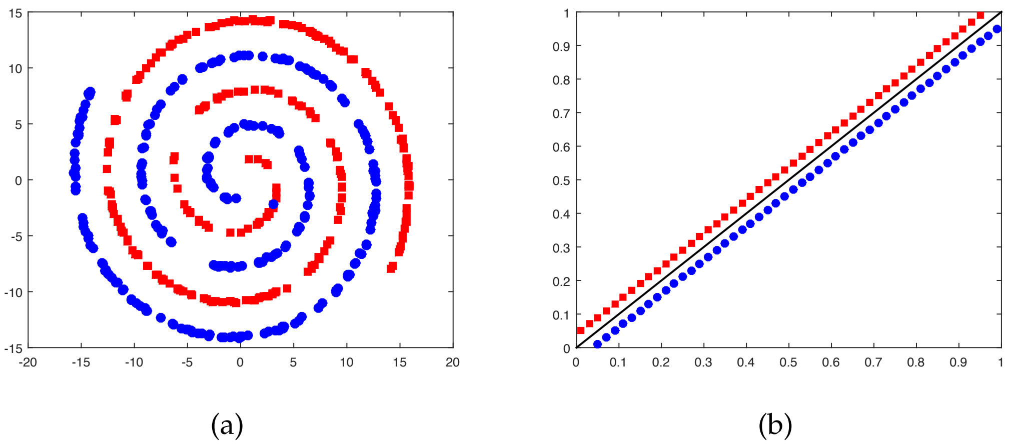


Figure 6.3: (a) Hypothetical example that is nonlinearly separable, (b) Another example that is linearly separable.

Figure 6.1b as synergistic? If we do so, then synergy, in context of gene interaction with respect to a disease, may be defined as the gain in discriminating power of a set of genes over that of the individual genes. To measure this discriminating power of a gene or a set of genes, with respect to a disease, we have a number of tools at our disposal such as information theoretic measures or various sophisticated classifiers. Interestingly, however, simply quantifying synergy as the gain in discriminating power may not be useful in the context of disease related gene interaction. This claim is illustrated with the help of Figures 6.2 and 6.3.

Consider, the interaction of two genes given in Figure 6.2. None of these two genes can differentiate between the two classes individually whereas they are able to do so jointly, i.e., the gain in discriminating power for the pair of genes over the individual genes is high. Moreover, as we can observe, the gene pair exhibits a “simple and interpretable” logical relationship between their expression levels and the classes. When, the expression levels of both genes are either low or high, the samples are non-tumorous, and tumorous otherwise. Compared to this, consider two hypothetical scenarios shown in Figure 6.3. It is evident that there exist no simple and interpretable logical relationship between the features in any of the cases. Nevertheless, using a sophisticated classifier, such as a

Support Vector Machine, we can show that, in Figure 6.3a, the two classes of samples are non-linearly separable and in Figure 6.3b, the two classes of samples are linearly separable when we take both features jointly into account. However, in both cases, none of the individual features would be able to discriminate between the two classes, i.e., the gain in discriminating power for the pair of features over the individual is high for both of the scenarios. Can we term these two cases as examples of synergistic interaction? The answer is definitely "NO" for Figure 6.3a as there is no simple and interpretable logical relationship between the two features and the class. These interactions do not make any sense from a biological perspective. For Figure 6.3b, one can debate whether it should be called synergistic or not - we prefer not to call it synergistic because practically there is no difference between the expression patterns of samples from the two classes. The relation between the features and class-1 is the same as that between the features and class-2. Thus, in our opinion, for quantifying synergy, the existence of a simple and interpretable logical relationship takes precedence over identifying just a gain in discriminating power. However, before proposing a definition for quantifying synergy, existing approaches are first reviewed at a glance.

As stated by Griffith and Koch [182], in the context of information theory, synergy is a property of a set of random variables, $\{X_1, \cdots X_n\}$ that collaborate/cooperate together to predict another random variable Y. A measure of synergy assesses the extent to which the collaborating effort is better than the individual effort. We shall focus only on synergy between two random variables or genes. Let G_1 and G_2 be two continuous random variables representing expression levels of two genes and C be a binary random variable representing the presence or absence of cancer, i.e., the phenotype. A well known measure of synergy is [182–184] :

$$S = I(G_1, G_2; C) - [I(G_1; C) + I(G_2; C)] \tag{6.1}$$

Here, I denotes the mutual information [185] and Equation (6.1) defines synergy as the whole minus sum. Here $I(G_1, G_2; C)$ denotes the mutual information of the gene pair (G_1, G_2) with the class C.

For a detailed discussion on this measure of synergy as well as on other measures, readers are referred to [182] and the references therein. Watkinson et al. [186] developed an information theoretic method for finding pairs of synergistic genes based on the measure of synergy defined in Equation (6.1). From Equation (6.1) we find that synergy is positive when the amount of information about cancer provided by a pair of genes together is more than the sum of information about cancer that are individually provided by the two genes forming the pair. Thus S is the amount of information about cancer that is gained because of the cooperation between $G1$ and $G2$ [187], [188]. Hence, it may be reasonable to call a pair of genes "synergistic" with respect to a phenotype when its synergy is positive. If S takes a negative value, it may be interpreted as redundancy. Authors in [186] have used Equation (6.1) to analyse a prostate cancer microarray dataset [189]. One of their important findings is that the gene RBP1 (cellular retinol-binding protein-1, also known as CRBP-1) exhibits synergistic relation with respect to prostate cancer with many other genes. Several of these genes linked with RBP1 are ribosomal genes.

The mutual information needed to find the synergy can be computed using conditional entropy. The computation of the conditional entropy would be very simple if we quantize the gene expression levels [190]. For example, if we binarize the expression levels into two states (0: "off" and 1:"on"), then each pair of genes can only be in one of the four possible states $(00, 01, 10, 11)$, and we have to count the number of times each of these four states appears in healthy and in diseased samples to compute synergy. But with such a quantization, we may lose information. To avoid this quantization problem, authors in [186] have used a hierarchical clustering of the gene expression levels. In the resulting dendrogram, expression levels of each gene as well as those of every pair of genes are clustered at all possible cut-off points up to a certain height and for every such partition the conditional entropy is computed and then aggregated using an weighted averaging. In other words, for every pair (G1, G2), we need to compute $H(C|G1), H(C|G2)$, and $H(C|G1, G2)$ many times where H denotes the conditional entropy of class C given gene expression levels. This procedure not only involves an enormous amount of computation, but also there is not much justification for using so many partitions to compute the conditional entropy. Furthermore, the estimation of conditional entropy using this approach may use different number of partitions for each pair of genes which, in turn, would make a comparison between the conditional entropies inconsistent.

Before discussing the proposed method, some issues related to the method in [186] are pointed out. The top most gene pair reported in [186] is (RBP1, EEF1B2) having synergy value 0.4025. Out of top 20 pairs reported in [186], 18 pairs involve RBP1 as one of the genes. But, looking at the scatterplot of the expression values of RBP1 against the corresponding sample numbers in Figure 6.4, we can see that RBP1 alone can correctly classify 81.37% of the samples in their respective classes. This is shown by the vertical line on the X-axis in the plot of RBP1. In Figure 6.4, only 19 samples are misclassified amidst the total number of 102 samples. It raises a question, should we consider the pair (RBP1, EEF1B2) as synergistic where a single gene has such a strong association with the disease (discriminating ability)? In our view the answer should be "No"! But why does it happen? There are two possible factors. The first factor appears to be the additive use of mutual information in defining synergy. To understand such an undesirable behavior, consider two hypothetical pairs of genes: A, B and X, Y where $I(A;C)$ is 0.75, $I(B;C)$ is 0.1, $I(A,B;C)$ is 0.95, $I(X;C)$ is 0.55, $I(Y;C)$ is 0.5 and $I(X,Y;C)$ is 0.95. Here the mutual information that these pairs provide about cancer are the same but clearly, the gain of information is higher for the pair $(X;Y)$. However, the value of synergy between $(A;B)$, using eqn. 6.1, is 0.1, while that between $(X;Y)$ is -0.1 (a negative value of synergy is considered redundancy thereby suggesting that the two genes together cannot provide any new information about cancer which is clearly not the case here). The second factor might be related to the way several partitions of the dendrogram are used to compute the conditional entropy of classes and hence the mutual information.

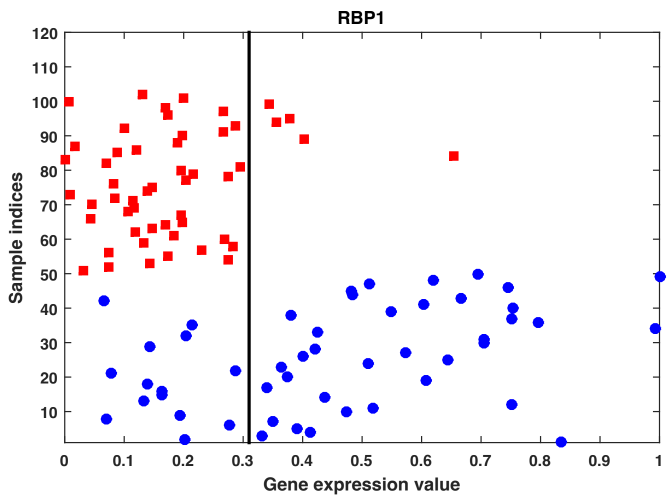


Figure 6.4: Gene expression values of RBP1.

6.2 Materials and Methods

6.2.1 Data

The prostate cancer microarray dataset is obtained from the Broad Institute's website [189]. It consists of expression values of 12,558 genes for 102 samples of which 52 are prostate tumor samples and the remaining 50 are non-tumor prostate samples. The dataset is normalized using the Robust Multi-array Average (RMA) method. This is referred to as dataset-1. For validation purposes, the CPDR prostate cancer dataset from the Walter Reed National Military Medical Center is used [191]. The dataset consists of expression levels of 54675 genes from 40 non-tumorous prostate tissue samples and 20 prostate tumor samples. This is referred to as the validation dataset. For experimental purposes, the expression values of each gene was normalized between 0 to 1 in both datasets.

6.2.2 Fuzzy Rule Based Method

Two genes G_i and G_j are in synergistic relation (collaborating with respect to a disease) if the following holds:

(a) neither G_i nor G_j can differentiate cancerous samples from normal ones satisfactorily but G_i and G_j can do so jointly,

(b) G_i and G_j together exhibit simple and interpretable logical relations to explain the phenotype.

We want to identify such pairs of synergistic genes. Any method for this, either implicitly or explicitly, will involve two steps: measuring the discriminating power of a pair of genes and its component genes with respect to classes and then assessing whether a pair enjoys a synergistic coupling or not. As we have discussed previously, for the first step, many information theoretic approaches have been proposed which rely on estimating either mutual information or conditional entropy of classes. The method proposed in [187] binarizes the gene expression levels for estimating conditional entropy of classes and thereby loses information. However, despite the loss of information, this method has one clear advantage over others, i.e., it is able to identify simple and interpretable logical relationships that exist between sets of genes due to binarization of the expression levels as demonstrated in Figure 6.2. So, quantizing expression levels provides us with simple logical relationships but it is also guilty of information loss. The loss of information is the highest for binarization but the logical relationships identified are also the simplest. Clearly, this is a matter of trade-off. In this context, for a natural extension of binarization and also for the sake of identifying interpretable logical relationships, an obvious choice is to use a fuzzy rule based system. That is what we try to do. Based on previous discussion we note that there are two requirements for finding synergistic pairs:

(a) explicit/implicit assessment of sufficient increase in discriminating power with respect to class for the gene pair over the individual genes of that pair,

(b) explicit/implicit identification of simple logical relations between gene expression values to explain the phenotype.

It is also worth noting here that, for the first task, any classification method can be used as we have labelled data. But unfortunately, such a method must be computationally very efficient because even for a dataset with 12558 genes, we need to design/train $\binom{12558}{80}$, i.e., about 80 million classifiers! Also we have previously mentioned that use of a sophisticated classifier may not be the best choice in this case as even though it may be able to provide a good estimation of the discriminating power, there may not be any easily interpretable relationship between the pairs of genes (eg. Figure 6.3a) which could provide useful biological insight [190]. For the second step, there is NO ground truth and either some expert provided rules or some other method not requiring ground truth has to be used. Fuzzy rule based systems appear to be a feasible and useful choice here. Possibly, fuzzy rule based system is the most appropriate choice of tool as it naturally finds simple and logical relations, can be designed at a low cost, and can be provided by expert in absence of ground truth.

To obtain the fuzzy rules, we define three linguistic values (membership functions), LOW, MEDIUM and HIGH, on each of the two linguistic variables (the two variables are g_i and g_j, which represent gene expression values of two genes G_i and G_j, respectively). Note that, one can define more than three membership functions, but for this problem three membership functions are found to be adequate. These membership functions can be Gaussian, triangular or trapezoidal in shape. Here we have used the Gaussian membership function defined by $\mu(x)=e^{-(x-c)^2/2\sigma^2}$ where c is the center and σ is the spread of the membership function. Suppose we extract three rule bases, R_i, R_j and $R_{i,j}$, using the expression profiles of G_i, G_j, and (G_i, G_j), respectively. Suppose we also find that R_i and R_j are poor performers to discriminate cancer samples from the normal ones, while $R_{i,j}$ is a good performer. In this case, G_i and G_j collaborate with respect to the disease. Furthermore, since we have opted to use three membership functions, i.e., LOW, MEDIUM and HIGH, the relationship between G_i and G_j describing the phenotype should also be easily interpretable. Hence, G_i and G_j are in synergistic relation.

One of the many methods available in the literature can be used to extract the rule bases [192–195]. An algorithm similar in spirit to the method by Ishibuchi et al. [193] is used, but it has been significantly simplified to reduce computational overhead. The domain of G_i is first divided into three equal parts/sections. These three parts correspond to our definition of LOW, MEDIUM and HIGH. The middle point of each of these parts is taken as the center of the corresponding membership function. In Figure 6.5 the thinner lines indicate the centers of each Gaussian membership function and the solid lines represent the boundaries of different sections. These choices are the most unbiased choices of centroids. We can use the sum of firing strengths like Ishibuchi et al. [193] to extract the rules.

Now, for a pair of genes, there are nine possible antecedents and two possible consequents : diseased and normal. So there are 18 possible rules of which nine could be consistent. For every pair of genes, we need to find three sets of fuzzy rules, one defined using G_i, one using the expression values of G_j and the rest using both genes G_i and G_j. One possible rule base is shown in Table 6.1 where N and D represents Normal and Diseased samples respectively. The entry in the cell $(2, 3)$ of Table 6.1 is D. This means that when G_i is medium and G_j is high, then the sample is diseased. In other words, the cell $(2, 3)$ represents a fuzzy rule : If G_i is MEDIUM and G_j is HIGH, then the sample is Diseased. In a similar manner, we can also define rules using expression values of just one gene, say, G_i. For example, a rule can be: If G_i is HIGH, then the sample is Normal.

Given a pair of genes (G_i, G_j), the set of rule antecedents are well defined and can be computed. What is left is the assignment of the consequent with each antecedent. For this purpose, like Ishibuchi et al. [193], we can use the sum of firing strengths (computed using a T-norm) for each antecedent provided by the normal and cancer samples and decide on the rule consequent. This will require computation of exponentiation to find membership values and use of a T-norm. To reduce computation, a fixed value of σ ($\sigma = 0.2$) is used for the membership functions, and the minimum is applied as the T-norm for computing firing strengths. Now the rules can be extracted without computing the membership values (avoiding exponentiation) and firing strengths as explained next.

For every data point, first we find the antecedent which best explains that data point (we find the antecedent which results in the highest firing strength for that data point which is equivalent to finding the cell in which the data point falls). For every antecedent (cell) we keep a count of such data points divided into two groups, points from diseased class and points from the normal class. Suppose for the antecedent $A_{i,j}$ the total number of points for which it enjoyed the maximum firing strength is $n_{ij} = n_{ij_D} + n_{ij_N}$. Note that, these n_{ij} points would be decided by the $(i, j)^{th}$ rule *irrespective* of the consequent of the rule. If the rule consequent is set to "Diseased" then this rule will make n_{ij_N} mistakes, while it will make n_{ij_D} mistakes, if the consequent of the rule is set as "Normal". Thus, the best consequent for this rule would be "Diseased" if $n_{ij_D} > n_{ij_N}$; otherwise, it would be "Normal". In this way we not only extract the rules for a particular pair of genes, but also simultaneously find how good these rules can model the behaviour of the data using each of the two genes as well as using both genes. We call such fuzzy rules as *Dynamic Fuzzy Rules* (DFR) as the antecedents are fixed but the consequents are not, they may change with each gene. We have three rule bases defined using G_i, G_j and the pair (G_i, G_j), which we denote by R_i, R_j, and $R_{i,j}$, respectively. Let the classification accuracy yielded by the three rule bases be P_i, P_j, $P_{i,j}$, respectively.

A detailed procedure for DFR generation for two genes, G_i and G_j, is provided in Algorithm 6.1. It is emphasized that the objective is not to find precise rules or any other classifier to achieve the best accuracy, as this may even have a derogatory effect (see Figure 6.3(a)).

Next, we need a mechanism to determine if a pair (G_i, G_j) is synergistic or not. Note that, we do not have any ground truth available on this and therefore NO supervised method can be used. Due to huge computational overhead (calculation of synergy for approximately a billion of gene pairs), *NO* computationally expensive procedure can be used either. Another set of fuzzy rules is used to identify whether a pair of genes forms a synergistic pair or not. This is referred to as Synergistic Rules (SR). For example, if P_i is LOW, P_j is LOW but $P_{i,j}$ is HIGH, then definitely the pair (G_i, G_j) forms a synergistic pair. This can be expressed using a fuzzy rule as follows:

Algorithm 6.1 Dynamic Fuzzy Rules (DFR)

```
(a) set $C^D_{l,m} = C^N_{l,m} = 0; l, m = 1, 2, 3$.
(b) Let the expressions levels for the two genes be $\mathbf{g}_i$ and $\mathbf{g}_j$ respectively
    where $\mathbf{g}_i = (g_{i,1}, g_{i,2}, \cdots, g_{i,N})'$ and $\mathbf{g}_j = (g_{j,1}, g_{j,2}, \cdots, g_{j,N})'$
(c) Define three membership functions/fuzzy sets , LOW (L), MEDIUM (M)
    and HIGH (H) on each of the two genes $G_i$ and $G_j$ as explained earlier.
(d) Repeat for $k = 1, \cdots, N$
    i Take the $k^{th}$ sample from each gene, $g_{ik}$ and $g_{jk}$ and compute the firing
      strength, $f_{l,m}$ for each of the nine antecedent clauses, i.e., for each of
      $l, m = 1, 2, 3$.
    ii Let $(L, M) = \max_{l,m} = \{f_{l,m}\}$. If sample $k$ is from the diseased class
      then $C^D_{L,M} = C^D_{L,M} + 1$ else $C^N_{L,M} = C^N_{L,M} + 1$
(e) End Repeat
(f) for $l, m = 1, \cdots, 3$ $C_{l,m} = \max\{C^D_{l,m}, C^N_{l,m}\}$
    Note that the above step is equivalent to dynamically assigning con-
    sequent of the $(l, m)^{th}$ antecedent as the class (Diseased or Normal)for
    which this antecedent best explains the training data.
(g) Compute $TC_{ij} = \sum_{l=1}^{3} \sum_{m=1}^{3} C_{l,m}$. $TC_{ij}$ is the total number of samples
    correctly explained (classified) by the gene pair $G_i$ and $G_j$. Let us define
    $P_{ij} = \frac{TC_{ij} \times 100}{N}$, the normalized (percentage) of data explained by the gene
    pair.
(h) Repeat steps similar to Step 3 through Step 5 with gene $G_i$ only and com-
    pute $P_i$, the fraction of data that is explained by the DFR defined on gene
    $G_i$ only.
(i) Repeat steps similar to Step 3 through Step 5 with gene $G_j$ only and com-
    pute $P_j$, the fraction of data that is explained by the DFR defined on gene
    $G_j$ only.
```

R_k: If P_i is LOW and P_j is LOW and $P_{i,j}$ is HIGH Then the pair (G_i, G_j) is synergistic.

The extent to which a rule is satisfied by a pair (G_i, G_j) is defined by the firing strength of the rule which can be computed using a T-Norm [120]. In this investigation, the T-norm min is used to compute the firing strength. For example, if G_i, G_j and (G_i, G_j) explain respectively p_i, p_j, and p_{ij} percent of data points correctly, then we find the membership values $\mu_{LOW}(p_i)$, $\mu_{LOW}(p_j)$, and $\mu_{HIGH}(p_{ij})$ and compute the firing strength of the above rule as $f_k = min|\{\mu_{LOW}(p_i), \mu_{LOW}(p_j), \mu_{HIGH}(p_{ij})\}|$. In our system, the firing strength plays a very important role as we use it to compute the p-values.

Table 6.2 depicts such a rule base which is defined using just three linguistic values LOW, MEDIUM and HIGH as shown in Figure 6.6 for the three linguistic variables P_i, P_j, $P_{i,j}$. The equations defining these membership functions are shown in Equations 6.2, 6.3 & 6.4. The same definitions have been used for all three linguistic variables. Note that, although the Synergistic Rule base remains the same for all pairs of genes, the dynamic fuzzy rule base for different pairs could be different.

$$\begin{aligned}\mu_{LOW}(x;a,b) &= 1, x \leq a \\ &= \frac{b-x}{b-a}, a \leq x \leq b \\ &= 0, b \geq x \end{aligned} \tag{6.2}$$

where $a = 50, b = 65$.

$$\begin{aligned}\mu_{MEDIUM}(x;a,b,c) &= 0, x \leq a \\ &= \frac{x-a}{b-a}, a \leq x \leq b \\ &= \frac{c-x}{c-b}, b \leq x \leq c \\ &= 0, c \leq x \end{aligned} \tag{6.3}$$

where $a = 55, b = 70, c = 85$.

$$\begin{aligned}\mu_{HIGH}(x;a,b) &= 0, x \leq a \\ &= \frac{x-a}{b-a}, a \leq x \leq b \\ &= 1, x \geq b \end{aligned} \tag{6.4}$$

where $a = 75, b = 90$.

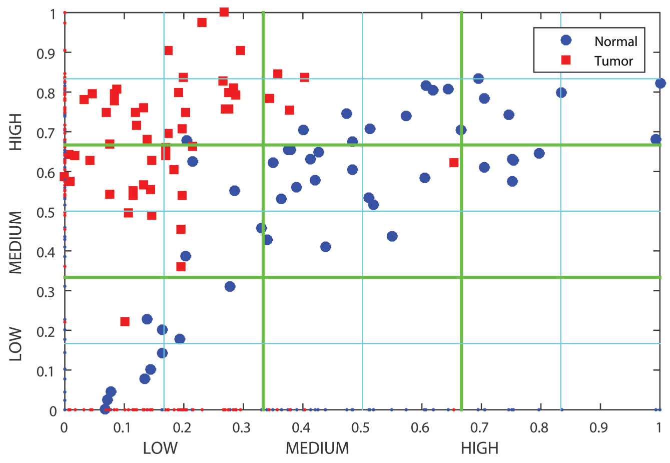


Figure 6.5: The fuzzy membership functions LOW, MEDIUM, HIGH are defined with centers at the thinner lines.

Table 6.1: A rule base that best explains the data on a pair of genes.

	LOW	MEDIUM	HIGH
HIGH	D	N	D
MEDIUM	N	N	D
LOW	D	D	N

There are 27 intuitive rules in Table 6.2. The first rule in Table 6.2 says the following : If P_i is LOW AND P_j is LOW AND P_{ij} is HIGH then the pair (G_i , G_j) is Synergistic.

This rule says that neither gene G_i nor gene G_j is able to explain the data, but the pair (G_i, G_j) can. This is an obvious rule for determining syn-

Table 6.2: Rule base to find synergistic pairs

Rule	P_i	P_j	P_{ij}	Status
1	LOW	LOW	HIGH	Synergistic
2	MEDIUM	LOW	HIGH	Synergistic
3	LOW	MEDIUM	HIGH	Synergistic
4	MEDIUM	MEDIUM	HIGH	Synergistic
5-27	For all other 23 possible antecedents			Non-Synergistic

ergism. Similarly, Rules 2-4 are also very intuitive. Clearly for each of the remaining 23 possible rules, the consequent is non-synergistic. Note that one can use other choices of membership function and may refine the membership functions based on training data, provided we know whether a pair is synergistic or not for every pair in the training data.

6.3 Choice of parameters

As already mentioned, the parameters for the three membership functions, μ_{low}, μ_{medium} and μ_{high} were set as $a_{low} = 50$, $b_{low} = 65$; $a_{medium} = 55$, $b_{medium} = 70$, $c_{medium} = 85$ and $a_{high} = 75$, $b_{high} = 90$.

Intuitively, they appear good choices for the fuzzy membership functions which define the classification accuracies/discriminating power of the individual genes and the gene pairs with respect to cancer. Experimentally we have found that these choices are quite useful as in all the synergistic gene pairs found by the proposed method, the individual genes had low or medium discriminating power while the pairs are able to classify the cancer and normal tissue samples sufficiently well. From this, we find that our objective of generating a fuzzy system to identify a synergistic network has been accomplished. This is further validated by utilizing the same model on another prostate cancer dataset (CPDR prostate cancer dataset) and demonstrating that the gene pairs identified by the proposed method are indeed synergistic.

To establish that the system is robust to the choice of parameters, the following experiments have been performed: First, we have computed the synergy between all possible gene pairs from the original dataset and by taking a threshold of 10^{-5} on the estimated p-values, we have selected the top 10167 gene pairs. Let us call this set of gene pairs S. Next, we have randomly selected any four out of the seven parameters and randomly shifted the value of each one of this four parameters by $+2\%$ or -2% with equal probability. The value of synergy between all possible gene pairs has been recomputed using the new set of parameters, and the top 10,167 pairs have been selected. The above step has been repeated 100 times. Let us call the set of gene pairs generated in each iteration as S_i ; $i = 1, ..., 100$. Now, to prove the robustness of our system, we intend to show that the similarity between our original set S and each of S_i is sufficiently high. To demonstrate this, we have computed the similarity between S and S_i as: $Similarity_i = \frac{|S \cap S_i|}{|S|}$. The average of $Similarity_i (i = 1, ..., 100)$ is 0.86 and standard deviation is 0.11 which suggests that our model is quite robust.

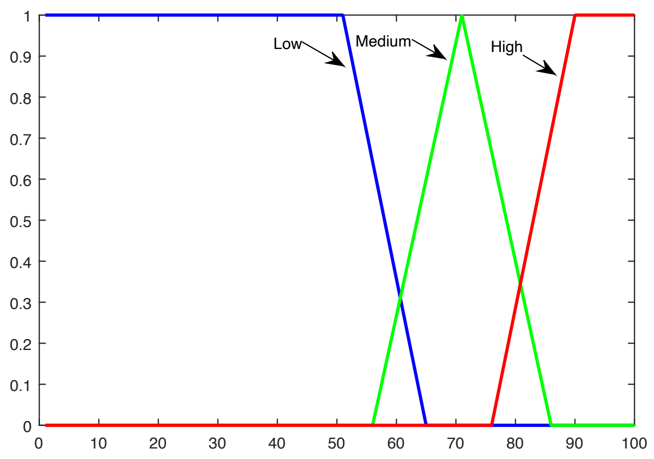


Figure 6.6: Fuzzy membership functions to find synergistic pairs.

In order to find synergistic pairs, we need to compute the firing strength of all 27 rules for every pair of genes, and then find the maximum firing strength. If the maximum firing strength corresponds to any of the first four rules in Table 6.2 then the pair is synergistic, otherwise, the pair is non-synergistic. Since we have to go through these process for all pairs of genes, to reduce the computation we apply the following schemes.

For any pair of genes, we first compute the firing strength of only the first four rules. If the maximum firing strength is zero, i.e., if a pair does not fire any of the four rules, then that pair is clearly a non-synergistic pair and we do not need to worry about that. In this way, we first find the set of potential synergistic pairs. Let there be N_1 such potential synergistic pairs of genes. Of these N_1 pairs, there could be some pairs for which the maximum firing strength of a non-synergistic rule may be higher than the maximum firing strength of a synergistic rule. If that happens, then those pairs are non-synergistic pairs. So we apply the i^{th} pair of the N_1 pairs of genes on the 23 rules for the non-synergistic class and find the maximum of the 23 firing strengths, f_i^N. For the same pair, let the maximum of the four firing strengths of the synergistic rules be f_i^S. Then if $f_i^N > f_i^S$, then we remove the i^{th} pair from the set of N_1 pairs of potential synergistic genes. In this way, we generate $N_S(N_s \leq N_1)$ pairs of potential synergistic genes. A schematic algorithm for finding synergistic pairs is shown in Algorithm 6.2.

The proposed method for finding synergy between a pair of genes can be directly extended to identify synergy among triplets or even larger groups of genes. Suppose we want to evaluate synergy for a set S of n genes with respect to a phenotype C. For the set S to be synergistic, the discriminating power of the whole set S should be high and the discriminating power of each of the possible subsets of S $(S_i \subseteq S|0 < |S_i| < n)$ should be low or medium. The number of such S_i will be $2^n - 2$. However, the discriminating capability of a set with respect to C should be higher than any of it's subsets. Therefore, we only need to consider the subsets of S having cardinality $n - 1$. There can be n such subsets S_i such that $S_i \subseteq S$ and $|S_i| = n - 1$. So, for each such S_i we need to evaluate 3^{n-1} antecedents and 3^n antecedents for S. The number of synergistic rules will be 2^n and the number of non-synergistic rules will be $(3^{n+1} - 2^n)$. We can follow the same procedure as used for bivariate synergy to evaluate these rules and compute synergy for S with respect to

Algorithm 6.2 Synergistic Rules (SR)

```
(a) Let N be the total number of gene pairs
(b) set $N_1 = 0; S_1 = \phi$
(c) Repeat i=1,2,...,N
     i. Compute the firing strength of the $i^{th}$ pair only on the first four rules.
    ii. Let $f_{im}$ be the maximum of the four firing strengths.
   iii. if $f_{im} \neq 0$ , then the $i^{th}$ pair is a potential synergistic pair; include $i^{th}$
        pair in $S_1$ and increment $N_1$ by 1.
(d) End repeat.
(e) $S_1$ is the set of potential synergistic pairs, $|S_1| = N_1$.
(f) Repeat i=1,2,...,$N_1$
     i. Compute the firing strength of the $i^{th}$ pair from $S_1$ on the 23 non-
        synergistic rules.
    ii. Let $f_i^N$ be the maximum of the 23 firing strengths. and $f_i^S$ be the
        maximum firing strength of the four synergistic rules for the $i^{th}$ pair.
   iii. if $f_i^N > f_i^S$,then $i^{th}$ pair is a non-synergistic pair, so remove it from $S_1$.
(g) End repeat.
(h) $N_s(= |S_1|)$ is the number of potential synergistic pairs where $N_s \leq N_1$.
```

C. However, to assess synergistic relation between more than two genes, we need more samples, irrespective of the method that we use. Since, the number of available samples is usually very limited, most of the studies that we could find are restricted to bivariate synergy.

Finally, *p*-values computed using permutations, as described in [186], are used to determine the final list of synergistic pairs. For *p*-value computation, the firing strengths play a very important role.

6.4 Statistical validation

To assess the statistical significance of the identified synergistic pairs, *p*-values computed using permutations, as applied in many other studies, are used. In particular, we follow the same permutation scheme as used in [186]. Let us assume that the expression data are stored in a 2-dimensional matrix, where each column represents expression values for a particular sample unit or person and each row corresponds to expression values of a particular gene over all samples. Each column is labeled as normal or cancerous. The expressions of each gene are independently shuffled once within the normal samples and once within the tumor samples. In this way the individual genes' association with the class is retained. But the genes' integrity in individual samples is destroyed. A total of 100 permutation experiments are performed, and 10^6 pairs are randomly selected from each set of permuted data, resulting in $N = 10^8$ pairs. The proposed method is then applied to compute the firing strength of each pair. To validate the results, it is examined whether the identified synergistic pairs occur by mere chance or genuinely exhibit synergistic relationships. Given a synergistic pair of genes found by our method from the given dataset, we count the total number of pairs of genes among these $N = 10^8$ pairs from the permuted datasets, which are at least as synergistic as the given pair in terms of the maximum firing strength over the four synergistic rules. Let, F_i be the maximum firing strength of the i^{th} synergistic pair of genes found by our method. Let f_k be the maximum firing strength of the synergistic rules for the k^{th} pair of genes selected from the permuted data, $k = 1, \cdots, N = 10^8$. The p-value for i^{th} synergistic pair is computed as in equation (6.5).

$$P_i = \frac{1}{N}\#(f_k \geq F_i); k = 1, \cdots, N \tag{6.5}$$

Note that we can compute the p-value for every pair of 78845403 genes, but that is not needed. It is enough to compute the P values for the $N_S =$ 59911 synergistic pairs of genes only, as identified by our method.

At this point two natural questions arise: why are we not using a non-fuzzy method and why are we not using a rigorous rule extraction method. These are explained in the following section.

6.5 Why not use any non-fuzzy method?

It is evident from the discussion and the figures that the relation between synergistic gene pairs is non-linear in nature. Therefore, to capture this non-linear relationship, a non-linear model is necessary. The non-linear model could be a Fuzzy System, an Artificial Neural Network, Support Vector Machine or any other appropriate learning system. One may argue that it is NOT necessary to use a fuzzy system, but there are some other considerations:

(a) If neural networks are used, a total of $78,845,403 + 12,558$ neural networks need to be trained for the prostate cancer dataset, corresponding to $12,558$ individual genes and $78,845,403$ gene pairs. Similarly, if we have 30000 genes, then we need $(449985000 + 30000)$ neural networks to be trained! This immediately suggests the necessity of using a tool capable of generating models at a very low computational cost.

(b) The second issue is that since we cannot use mechanisms like cross validation to find a good system (again because of the fact that we have to generate too many models), we need to make sure, to the extent possible, that none of the systems learned memorizes the data to yield a very high (false) accuracy.

(c) Moreover, use of machine learning tools like Support Vector Machines or Artificial Neural Networks, exhibit very high enhancement of discriminating power for data like Figure 6.3a, where the relation absolutely is not synergistic in nature.

(d) Finally, For such tools, finding simple and logical relations become an almost impossible task. These attributes suggest us to use the fuzzy rule based system.

6.6 Why not use a rigorous rule extraction procedure?

The proposed system incorporates two types of fuzzy rule bases: DFR and SR. The first type of rules, DFR, compute discriminating power of single and pair of genes and the second type of rules, SR, determine whether a pair is synergistic or not. For the synergistic rules, we cannot use any rigorous method of rule extraction as there is NO labeled training data; i.e., NO ground truth is available on synergistic relations. For the DFR, theoretically speaking, we can use any supervised rule extraction scheme, we can even refine the rules, but this is neither desirable nor practically feasible. A refined rulebase which is able to separate the classes with high precision may not be a desirable choice here as it would possibly generate very specialized rules to accomplish this task, thereby destroying the "simple and logically interpretable" nature of the rules (see Figure 6.3a). It is also not feasible because of the massive computational overhead. For example, we have already mentioned that even for a dataset with 12558 genes, we need to extract $(78845403 + 12558)$ rule bases. Moreover, rules extracted by any exploratory data analysis based method, may be optimal, but will not be as interpretable as the rules defined using human provided linguistic values.

6.7 Computational complexity

Let k be the computational cost of evaluating a rule involving two antecedent clauses. Then for a pair of genes, total cost of evaluating the classification accuracy would be $9nk$, where n is the number of samples. The total cost of evaluating the synergy rule base is 27k as there are 27 rules in the rule base. Thus the total worst case cost of evaluating synergy for a pair of genes is $O(n)$; while the best case computational complexity of the method by Watkinson et al. is $O(n^2)$.

Further because of the simplification of implementation in Algorithm

Table 6.3: Top ten pairs based on the firing strength along with the p-value from dataset-1

Rank	Gene Symbol	Probe ID	Gene Symbol	Probe ID	Firing strength	p-value
1	PTGDS	38406_f_at	SLC25A6	40436_g_at	0.91	$< 10^{-8}$
2	PTGDS	216_at	SLC25A6	40436_g_at	0.88	$< 10^{-8}$
3	PTGDS	216_at	HSPD1	37720_at	0.81	4×10^{-8}
4	NME2	1980_s_at	EIF4EBP2	35263_at	0.81	4×10^{-8}
5	ACSL3	33880_at	EFS	33883_at	0.81	4×10^{-8}
6	PTGDS	38406_f_at	XBP1	39756_g_at	0.76	4×10^{-8}
7	EIF4EBP2	35263_at	XBP1	39756_g_at	0.76	4×10^{-8}
8	CLU	36780_at	XBP1	39756_g_at	0.76	4×10^{-8}
9	PLP2	37326_at	XBP1	39756_g_at	0.76	4×10^{-8}
10	PTGDS	216_at	XBP1	39756_g_at	0.76	4×10^{-8}

Synergy, the total cost of evaluation is further reduced. Let there be N pairs of genes. If we follow the normal way a fuzzy rule based system works, the computational complexity would be $(9nk + 27k)N + 3n$. To reduce the computational complexity we follow the trick as explained in the Algorithm SR. First, we apply only the four rules for the synergistic class and this will involve a cost of $4Nk$. Let N_1 be the number of gene pairs for which at least one of the four synergistic rules fire with a non-zero firing strength. The remaining 23 rules are applied to these N_1 potential synergistic gene pairs. The corresponding cost is $23N_1k$. So the total complexity is $9nkN + 3n + 4Nk + 23N_1k$ and our savings is $23(N - N_1)k$. Although the order of complexity remains the same, it can lead to a substantial saving in computation. For dataset-1, $N = 78845403$ and $N_1 = 833144$ which results in a huge saving in computation.

6.8 Results

Dataset-1, as mentioned earlier, has $12,558$ genes and hence there are 78845403 pairs which have to be checked for possible synergistic behavior. Out of this, based on only the four rules for synergistic pairs, we have found 833144 pairs of suspected synergistic candidates.

These pairs are then further filtered using the 23 rules describing the non-synergistic relation and that resulted in 59911 pairs of genes. We then computed p-values for each of these 59911 pairs to generate the final list of synergistic pairs, which resulted in 1501 pairs of genes with a threshold of 10^{-6} on the p-value. Table 6.3 shows the top ten synergistic pairs found by our method along with their p-values. The maximum firing strength we have found is 0.91 for the rule : If P_i is MEDIUM and P_j is MEDIUM and P_{ij} is HIGH then Synergistic.

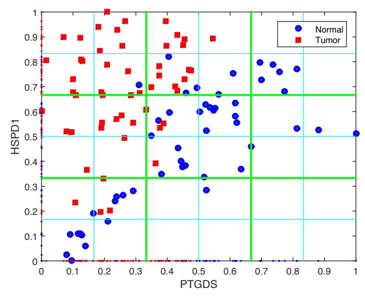
It is worth noting here that the fuzzy rule bases are not optimized or tuned, yet these are able to find useful synergistic pairs. In Figure 6.8 we display the scatterplots of the top two pairs of genes given in Table 6.3, which clearly reveal the synergistic behavior of the gene pairs found by the fuzzy rule based system. Scatterplots of the top $3-10$ gene pairs given in Table 6.3, are shown in Figure 6.7.

A natural question comes: Is the fuzzy method better than the state of the art method such as [186]? There is no ground truth for such a problem. So, like classification problems, we cannot compare metrics like misclassification rates. Yet, we can say a few things :

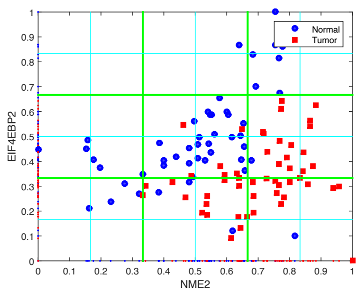
(a) our method is computationally much more efficient than existing method. This is important because we need to compute synergy for nearly a billion pairs of genes;

(b) our method naturally reveals simple and logical relations for every synergistic pair; (3) We have already explained the problem associated with synergy expression in eqn. (6.1).

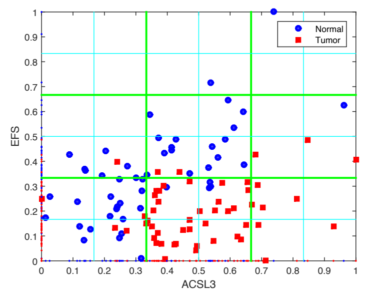
To demonstrate the superiority of the proposed fuzzy method, let us now further consider the top ten synergistic pairs that are found by the method in [186]. These 10 pairs are depicted in Table 6.4. Comparing Table 6.4 with Table 6.3, we find that there is only one common pair of genes, which is (PTGDS, SLC25A6). Table 6.4 also depicts that among the top 10 pairs, RBP1 appears 8 times and PTGDS appears two times suggesting that RBP1 is a very important member of the synergistic pairs.



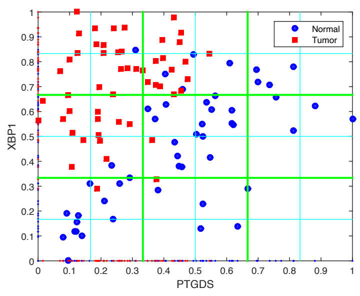
(a)



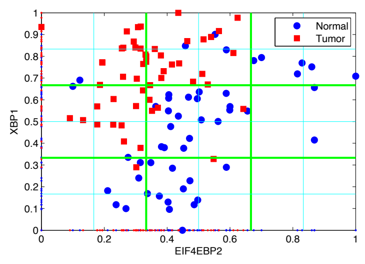
(b)



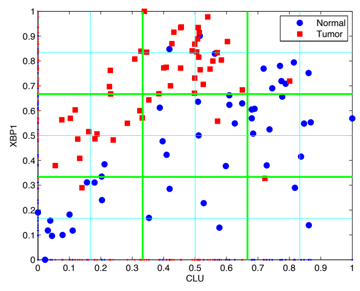
(c)



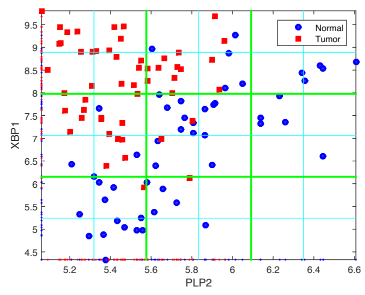
(d)



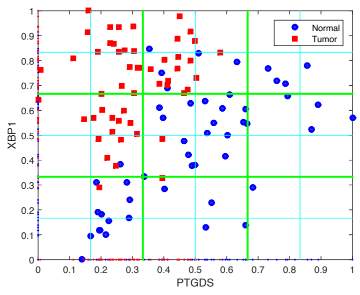
(e)



(f)



(g)



(h)

Figure 6.7: Scatterplots of top 3-10 gene pairs.

However, RBP1 does not even appear once in the top 10 synergistic pairs found by the fuzzy system. In fact, in [186] RBP1 is found to be the largest hub gene. However, in our method, with a threshold of 10^{-6} on the p-Value, there was no pair involving RBP1 in the top 1501 synergistic pairs. Why is it so? As mentioned earlier that RBP1 can correctly classify 81.37% samples by just based on a threshold and hence RBP1 alone is a strong biomarker/discriminator of prostate cancer and hence it should not be in a synergistic relation with any other genes. This fact demonstrates that the behavior and performance of the fuzzy rule based system is significantly better that the state of the art method.

Table 6.4: Top 10 gene pairs based on synergy [186]

Gene symbol	Gene symbol	Synergy
RBP1	EEF1B2	0.4025
RBP1	FTL	0.3653
RBP1	HLA-DPB1	0.3493
PTGDS	YWHAQ	0.3408
RBP1	UQCRH	0.3348
RBP1	UBC	0.3331
RBP1	SNRPB	0.3287
RBP1	ZNF146	0.3271
RBP1	EEF1D	0.3239
PTGDS	SLC25A6	0.3202

An analysis of the synergistic pairs reveals a number of hub genes. The details of the top 10 hub genes (top in terms of number of genes connected to it) are shown in Table 6.5. The topmost hub gene that we have found is TRGC2 which is connected with 388 genes in its network with a threshold on p-value at 10^{-6}. Figure 6.9 depicts part of the synergy network considering the top four hub genes (Table 6.5). In particular, to keep the figure size manageable we have used $100, 50, 25$ and 10 genes associated with the top four hub genes respectively. As we can see from Figure 6.9, the synergy network exhibits a small world structure.

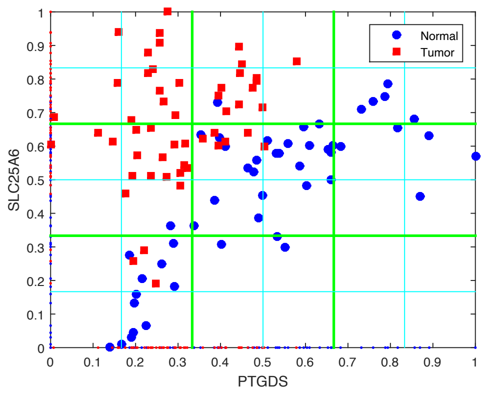
Table 6.5: Top 10 hub genes

Rank	Symbol	Size of hub
1	TRGC2	388
2	CFD	221
3	TACSTD1	199
4	S100A4	98
5	GSTM1	54
6	CCND2	48
7	PTGDS	43
8	PTGDS	41
9	GOS2	35
1 0	TSPAN1	34

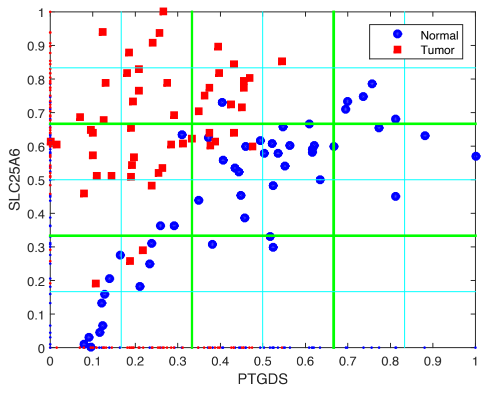
Probe ID of PTGDS in rank-7 is 38406_f_at
Probe ID of PTGDS in rank-8 is 216_at

Table 6.6: Firing strength along with *p*-value of the top 10 synergistic gene pairs found using the fuzzy rule based method from the CPDR dataset.

Rank	Probe ID	Probe ID	Firing Strength	*p*-value
1	200949_x_at	38892_at	0.78	$< 10^{-8}$
2	1555942_a_at	213377_x_at	0.67	1.1×10^{-7}
3	1555942_a_at	218474_s_at	0.67	1.1×10^{-7}
4	1555942_a_at	224598_at	0.67	1.1×10^{-7}
5	1557055_s_at	241523_at	0.67	1.1×10^{-7}
6	1559392_s_at	201820_at	0.67	1.1×10^{-7}
7	1563073_at	216920_s_at	0.67	1.1×10^{-7}
8	200877_at	229309_at	0.67	1.1×10^{-7}
9	201064_s_at	201820_at	0.67	1.1×10^{-7}
10	201064_s_at	205141_at	0.67	1.1×10^{-7}



(a)



(b)

Figure 6.8: Scatterplots of top two gene pairs in Table 6.3.

Table 6.7: Synergy values of the top 10 synergistic gene pairs found by Watkinson et al's method from the CPDR dataset.

Rank	Probe ID	Probe ID	Synergy
1	224573_at	220401_at	0.10
2	1569086_at	224573_at	0.09
3	224573_at	219512_at	0.09
4	1553020_at	206452_x_at	0.09
5	224573_at	206773_at	0.09
6	214351_x_at	1569459_a_at	0.09
7	220337_at	224573_at	0.09
8	212734_x_at	232043_at	0.09
9	219178_at	204294_at	0.09
10	204376_at	218205_s_at	0.09

6.8.1 Validation on independent Data

The validation dataset contained the expression levels of 54675 genes. To reduce the number of computations, we have sorted the genes by the standard deviation of their expression levels. As we have seen in previous examples, the genes participating in a synergistic interaction usually have high standard deviations. Therefore, we have selected the top 8000 genes with the highest standard deviation from the total number of 54675 genes. This is done just to reduce the computational overhead. Thus, the

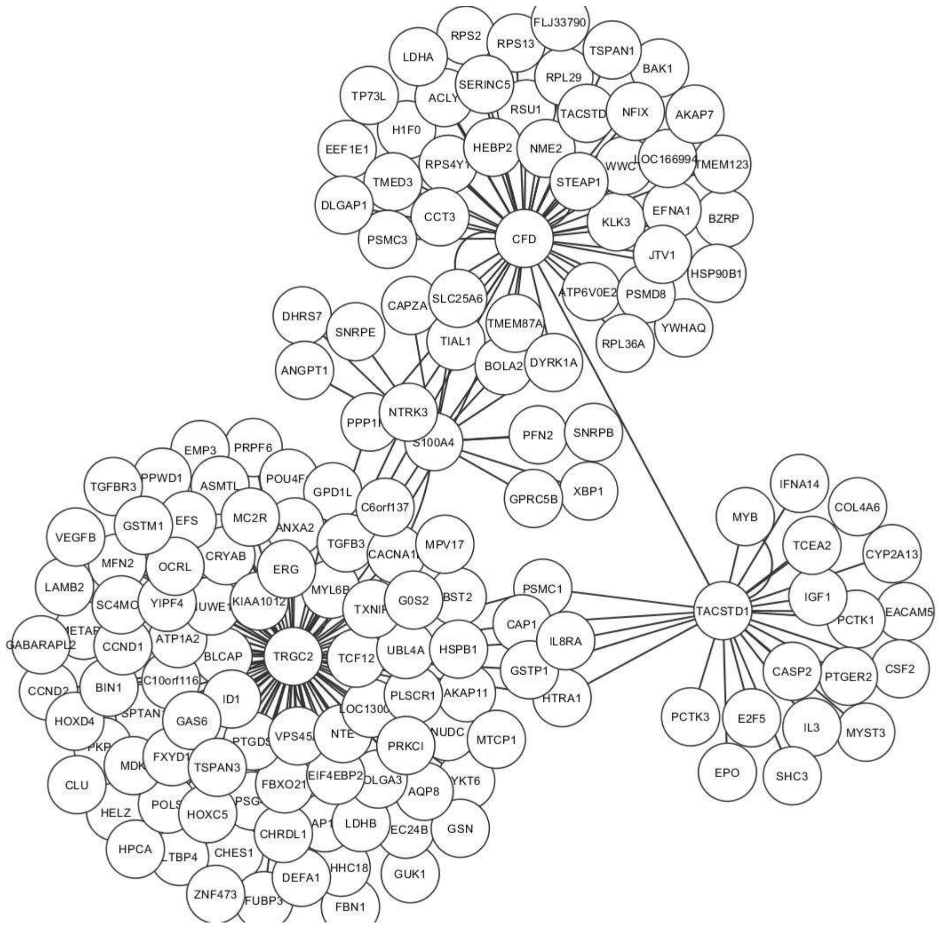
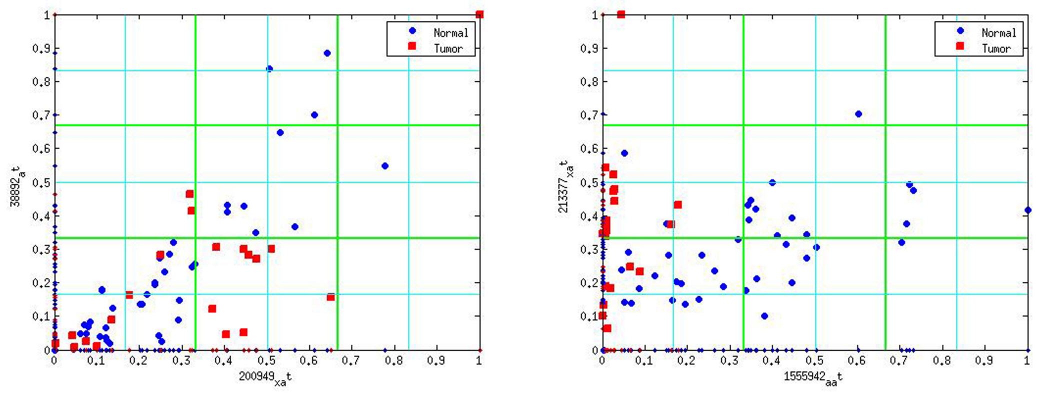


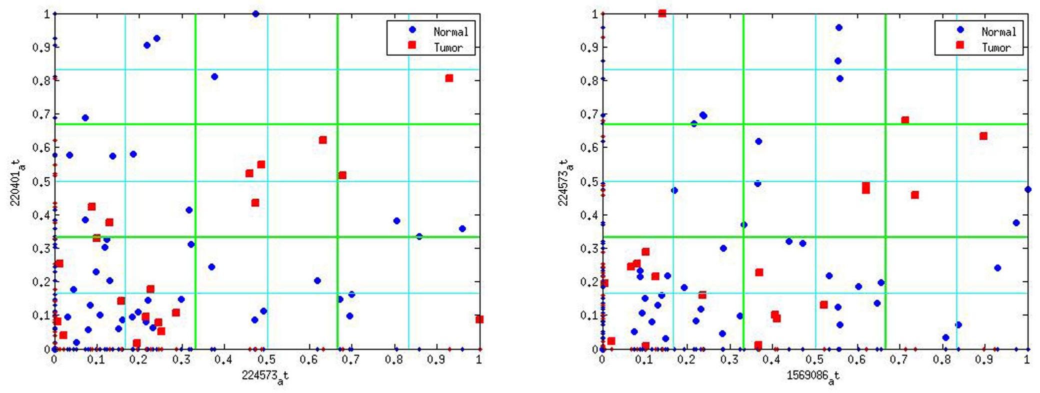
Figure 6.9: Synergy network.

total number of synergistic pairs to be evaluated has been reduced to $\binom{8000}{2}$ from $\binom{54675}{2}$. We have computed the firing strength for these $\binom{8000}{2}$ gene pairs and reported the top 10 gene pairs along with their p-values in Table 6.6. The scatterplots of the top two synergistic gene pairs are shown in Figure 6.10. These gene pairs indeed display some kind of synergistic behaviour with respect to prostate cancer. For the sake of comparison with the method proposed by Watkinson et al., we have also computed the synergy scores of the same set of $\binom{8000}{2}$ gene pairs using their method and depicted the top 10 gene pairs in Table 6.7. The scatterplots of the top two gene pairs are shown in Figure 6.11. As evident from Figure 6.10 and Figure 6.11, the fuzzy rule based method is able to find comparatively better synergistic gene pairs from the validation dataset than those by the method of Watkinson et. al. [186].



(a) (b)

Figure 6.10: Scatter plots of top two synergistic gene pairs found using the fuzzy rule based method from the CPDR dataset.



(a) (b)

Figure 6.11: Scatter plots of top two synergistic gene pairs found by Watkinson et al's method from the CPDR dataset.

6.9 Biological significance

Here we discuss the biological relevance of some of the genes that we have found.The biological importance of the top four hub genes, presented in Table 6.5, is elaborated in the context of cancer pathways.

In this study the topmost hub gene that we have found, is T cell receptor gamma constant 2 (TRGC2) which is a biomarker for Stage-I Lung Adenocarcinoma [196].This gene is found to be significantly down-regulated in stage I lung cancer compared to the control groups [197].

CFD (Complement Factor D), the second hub gene, is found to be over expressed in gastric cancers relative to paired-normal and apparently-normal as reported in [198].

Our third dense hub gene is TACSTD1. The gene TACSTD1 or tumor-associated calcium signal transducer 1 is also know as EpCAM (Epithelial cell adhesion molecule). This gene encodes a carcinoma-associated antigen [199]. Initially EpCAM was considered a dominant antigen in human colon cancer. Authors in [199] have reported that EPCAM is practically found to be expressed in all human adenocarcinoma. This gene is also found to be expressed in retinoblastoma and in hepatocellular carcinoma [199]. EpCAM is frequently upregulated in carcinomas but is not expressed in cancers of non-epithelial origin. All these suggest the utility of EpCAM as a diagnostic biomarker for various cancers.

According to [200], during the progression of prostrate cancer, the gene S100A4 (the fourth hub gene in our list) is over expressed. The gene S100A4, a calcium binding protein, plays a key role in accelerating invasion of human prostate cancer and it may also be linked with its metastatic spread. This gene also plays an important role in the progression of breast cancer and could be an independent marker of breast cancer [201]. Gupta et al. [202] stated that S100A4 is frequently over expressed both in metastatic tumors and normal cells with uninhibited movement. The gene S100A4 has been found to be expressed in different cancers such as breast, ovary, thyroid, gastric and colon. Higher expression of S100A4 for prostate cancer is correlated with higher tumor grade. For breast cancer, an over expression of S100A4 is closely correlated with a fatal outcome. Thus we see that S100A4 plays very important roles in biology of different types of cancers.

Table 6.3 shows that PTGDS appears five times in the top ten pairs of genes. Existing literature reveals that over expression PTGDS or L-PGDS along with downregulation of P13-K contributes to PMA induced apoptosis [203]. PTGDS translates into a protein called glutathione-independent prostaglandin D synthase. This protein acts as a catalyst in the the conversion of prostaglandin H2 (PGH2) to prostaglandin D2 (PGD2). PGD2 has been shown to be a inhibitor of cell migration and invasion of PC-3 prostate cancer cells [204].

The partner genes of PTGDS are SLC25A6, HSPD1 and XBP1 in the list of top 10 synergistic pairs (Table 6.3). SLC25A6 (solute carrier family 25 member 6) is a member of the mitochondrial carrier subfamily of solute carrier protein genes. Several studies have shown that this gene is an important marker gene for prostate cancer, see [205] and references there in.

XBP1, on the other hand, appears five times and it has been shown to exhibit overexpression in breast cancer cells and colorectal tumors and is also a critical transcriptional regulator of ER stress which is responsible for tumor malignancy [206]. XBP1 also plays an important role in preventing oxidative stress as XBP1 deficient cells were found to have lost mitochondrial membrane potential which resulted in subsequent cell death [207]. Recently it has been reported that in Triple-Negative Breast Cancer, a highly aggressive malignancy, XBP1 plays a crucial role in the tumorigenicity and progression of this cancer [208]. Apart from that, the reduction of human X-box binding protein 1 (hXBP-1) expression may be a useful marker for prostate adenocarcinoma differentiation and progression as reported in [206].

Thus, it is found that the identified synergistic network has strong relevance in cancer biology.

6.9.1 Pathway analysis

Analysis of gene expression data to find the interaction of individual genes with phenotype is useful, but it cannot give the bigger picture of how a set of genes together interact. Pathway analysis is one of the methods to look for changes in many genes with a common function. Therefore, to interpret the role of a gene, we go for Gene Ontology or curated databases of pathways.

In this study we explore the KEGG (Kyoto Encyclopedia of Genes and Genomes) database using Gene Trail [209]. Table 6.8 shows the pathways involving five of more genes with a *p*-value of less than 0.05. This table also reports the number of genes associated with different subcategories along with their *p*-values.

The largest hub gene, TRGC2, along with its directly connected genes, is selected for pathway analysis. It is found that eight of these genes are associated with the Endocytosis pathway. The Endocytosis can make an important contributions to the hallmark of cancer [210]. It also increases the invasiveness and avoidance of apoptosis. The effect of reactive oxygen species (ROS) on endocytosis may translate to survival advantages during the chemotherapy or radiotherapy as reported in [211]. It is also mentioned that very few studies have been done on these issues. Researchers suspect that Endocytosis is one of the physiological processes that turns into cancer [211].

Gonadotropin-releasing hormone (GnRH) is expressed in the castration-resistant prostate cancer even when the tumor has escaped steroid dependence [212]. Further GnRH-R is expressed in prostate, breast, endometrial, and ovarian cancer. We have found seven genes in this pathway.

Six genes have been identified as involved in the prostate cancer pathway. Apart from these, six other genes are found to be associated with Chronic myeloid leukemia and ErbB signaling pathway. The ErbB receptor tyrosine kinases play important roles in human cancers including breast cancer. The role of Epidermal growth factor receptor (EGFR) and ErbB2 in cancer development and progression has been observed by different researchers [213, 214].

Five genes have been identified as associated with the Janus kinase (JAK) signal transducer and activator of transcription (STAT) signaling pathway. JAK-STAT signaling pathway is critical to many cytokine receptors. It is also important in blood formation and pathogen resistance.

Table 6.8: Summary of significant subcategories of KEGG database

Subcategory	Observed	p-value
Glioma	6	$1.22 * 10^{-3}$
ErbB signaling pathway	6	$1.49 * 10^{-3}$
Jak-STAT signaling pathway	5	$2.59 * 10^{-3}$
Neurotrophin signaling pathway	5	$2.92 * 10^{-3}$
GnRH signaling pathway	7	$6.18 * 10^{-3}$
Chronic myeloid leukemia	6	$6.19 * 10^{-3}$
Prostate cancer	6	$6.73 * 10^{-3}$
Endocytosis	8	$1.61 * 10^{-2}$
Fc gamma R-mediated phagocytosis	5	$1.8 * 10^{-2}$
Long-term depression	7	$2.25 * 10^{-2}$
Colorectal cancer	5	$3.2 * 10^{-2}$
Phosphatidylinositol signaling system	5	$4.13 * 10^{-2}$

The JAK-STAT pathway plays an important role in cancer and the dysregulation of this pathway has been observed in different cancers [215]. Authors in [216] have suggested that successive activation of members of the STAT family is related to different types cancers, including blood malignancies. Five genes have been identified in the Colorectal cancer (commonly known as colon cancer) pathway.

6.10 Conclusion

In this chapter, a new philosophy has been proposed to identify pairs of genes that interact synergistically with respect to prostate cancer. A novel fuzzy rule-based framework has been used to develop an algorithm for this purpose. The proposed system uses two kinds of rule bases, Dynamic Fuzzy Rules (DFR) and Synergistic Rules (SR). The DFR is able to identify simple and interpretable logical relationships between pairs of genes and it is the natural extension of a previous approach which used binarization of gene expression levels. The proposed method has at least four advantages over the existing approaches:

(a) for the first rule base (DFR), use of any non-fuzzy model will require training of billions of systems and that makes it computationally impractical,

(b) maximizing the training accuracy may produce misleading results (non-synergistic relations can be labelled as synergistic as explained using the two spiral dataset in Figure 6.3a), and hence, there is no need for an expensive training method,

(c) non-fuzzy techniques will not help biologists to understand the synergistic relations as the relations may not be interpretable, and

(d) for the second rule-base (SR) we cannot use other machine learning techniques as there is NO ground truth.

For each of these four issues, fuzzy rule based systems give a natural and effective solution.

However, the proposed method is also limited in a few ways. While it can be extended straightway to find synergy between triplets or higher number of genes, doing so will exponentially increase the number of rules as we have shown before. Therefore, the proposed method may be computationally prohibitive for identifying synergy between a large number of genes. Since, the number of samples are usually small, evaluating a large rule base may not be sensible as well. Furthermore, the parameters of the DFR fuzzy rules has to be provided by expert.

To demonstrate the efficacy of the proposed method, it has been applied to two prostate cancer datasets. The synergistic pairs identified by our algorithm play significant roles in cancer biology and may have potential therapeutic use. This is the first attempt to exploit fuzzy systems in identification of synergistic gene pairs. The advantage of this method is that it can be easily extended to find synergy involving more than two genes. It is computationally much more efficient than the existing information theoretic methods. More importantly, a human expert can provide/interpret the rules defining synergy. In this study, to reduce the computation and to demonstrate its effectiveness, we have not tuned any parameter of the dynamic fuzzy rule bases. Thus, further improvement in results might be possible with tuning of the rule bases, but it can become computationally prohibitive.

As a final note, it is emphasized that synergistic relationships are not exclusive to gene interactions; therefore, the proposed method can also be useful for identifying meaningful, readable non-linear relationships between features in other contexts. One example of this would be to discover meaningful patterns in online shopping behavior.

Chapter 7

Conclusions and Future Scopes

7.1 Conclusions

In this thesis, five research problems related to pattern recognition have been addressed using a Fuzzy set theoretic framework. In Chapter 2, a method is proposed for feature selection during clustering to achieve a target partition. For clustering sometimes we use Kernel space (a high dimensional space) with a hope to get better results specially for a complex datasets. In Chapter 3, a critical question is raised regarding the use of the Kernel trick for clustering datasets in high-dimensional spaces, highlighting how it may potentially mislead researchers. Chapter 4 builds on this discussion by further analyzing the problem presented in Chapter 3, demonstrating the sensitivity of Kernel parameters, and providing guidelines for their appropriate use. A method to visualize the data structure in high-dimensional space has also been discussed. We all know that FCM is a good clustering method and it is frequently used on image data with a spatial constraint. Note that, spatial information is only relevant for image where we have the neighbouring pixel information. In Chapter 5, a method is presented for utilizing Fuzzy C-Means (FCM) by exploiting neighboring information for non-image datasets. Finally, in Chapter 6, the focus returns to feature selection, specifically targeting a special type of features known as synergistic pairs of features. The utility and effectiveness of this approach are demonstrated in addressing a specific, yet important, problem related to prostate cancer.

In Chapter 2, a novel feature selection technique tailored for clustering was introduced. While most unsupervised feature selection models rely on penalty or regularizer-based approaches within the FCM framework, the proposed innovative approach seeks to select features that can maintain the cluster structure found by FCM in the original data space within a lower-dimensional feature space. The uniqueness lies in using the FCM partition matrix, or any other partition matrix from a different clustering algorithm, to guide feature selection. The experiments, conducted on synthetic data and real datasets, demonstrated the model's ability to identify relevant features, particularly in scenarios with well-separated clusters in a lower-dimensional subspace. Emphasizing the importance of data structure during feature selection, the proposed method distinguishes itself by aiming to preserve the original high-dimensional cluster structure, setting it apart from other unsupervised feature selection approaches using the FCM framework.

In Chapter 3, a fundamental question was explored regarding the utility of clustering in kernel spaces, particularly with polynomial kernels. The investigation focused on whether the clusters identified in the kernel space genuinely correspond to the same groups present in the original feature space. The chapter highlighted the complexities involved in kernel clustering, emphasizing that kernelization may impose undesirable structures on the data, leading to unexpected results. The exploration demonstrated the significant impact of kernel parameters, particularly with the Gaussian kernel. It was argued that selecting appropriate parameters for kernel clustering in unsupervised scenarios presents notable challenges, as traditional cluster validation methods may not be directly applicable.

In Chapter 4, the exploration of clustering in kernel space was continued, addressing philosophical questions regarding its utility and the conditions under which it can be beneficial. It was emphasized that poor clustering results arising from local minimum issues or incorrect algorithmic parameters are distinct from the philosophical questions under consideration. Notably, for the Gaussian kernel, it was discussed how higher values of the parameter σ tend to produce FCM-like partitions in kernel clustering, whereas lower values may result in partitions that differ from those generated by FCM. It was argued that assessing the desirability of such partitions in high-dimensional data remains a challenging task. The perspective emphasized the importance of incorporating domain knowledge into kernel functions and developing specialized cluster-validation techniques to enhance the meaningfulness of kernel clustering.

In Chapter 5, FCMS was introduced as an extension of the Fuzzy C-Means (FCM) algorithm, specifically designed to address the limitations of conventional clustering methods in scenarios characterized by complex spatial relationships. While traditional FCM excels in well-defined clusters, FCMS incorporates spatial/neighbourhood information to handle datasets with unclear cluster boundaries or spatial continuity. This chapter highlighted FCMS's adaptability beyond image data, showcasing its effectiveness in non-image datasets. The inclusion of a penalty term in the objective function facilitates the preservation of proximity among neighboring data points, proving beneficial in datasets where natural clusters lack compactness. Experiments on synthetic and real-world datasets demonstrated FCMS's efficacy in diverse scenarios. The chapter emphasized parameter sensitivity, highlighting the roles of the fuzzifier, penalty parameter, and the number of neighbors in fine-tuning FCMS's performance.

In Chapter 6, the focus shifted to identifying synergistic gene pairs related to prostate cancer using a fuzzy rule-based framework. A novel philosophy was proposed for discovering gene pairs that interact synergistically, with an emphasis on the interpretability of the generated rules. The proposed method differentiated itself by addressing computational challenges and providing interpretable rules, particularly in the context of identifying synergistic gene pairs. The method's efficiency, potential therapeutic implications, and the ease with which it can be extended to capture synergy involving more than two genes have been highlighted.

The application of fuzzy systems in the identification of synergistic gene pairs marks a significant contribution, demonstrating its computational efficiency and interpretability compared to existing methods.

7.2 Limitations and Future Scopes

Like most investigations, the works presented in this thesis also have some limitations. These points are discussed below, along with potential directions for future research.

In Chapter 2, the focus was on selecting a single set of features applicable to all clusters. However, it's worth considering applications where distinct clusters may exist in different subspaces. Expanding the concept to accommodate such cases could enhance its applicability. Additionally, instead of relying solely on the cluster partition matrix, exploring the effectiveness of the proposed framework in a supervised mode, utilizing class labels, could offer valuable insights.

In Chapters 3 and 4, the application of the kernel trick and the manipulation of kernel parameters were thoroughly explored. In essence, leveraging the kernel space effectively requires domain knowledge and the application of suitable cluster validation techniques. In this context, the incorporation of graph-based cluster validity could be a valuable approach. Consider, for instance, the scenario where a partition achieved through kernel clustering accurately captures the "desired" cluster structure inherent in the original data. In such cases, two expectations arise: Firstly, a minimal spanning tree (MST) derived from the kernel-transformed data should mirror (at least to agreat extent) the structure of an MST derived from the original data. Secondly, both MSTs should exhibit a higher density of edges within each cluster and fewer edges (ideally just one) connecting every pair of clusters. Nonetheless, a comprehensive investigation is necessary to substantiate the practical utility of such frameworks.

In Chapter 5, the exploration demonstrated a significant enhancement in clustering performance through the application of FCMS, particularly in non-image datasets where traditional FCM methods encounter challenges. By integrating spatial information, FCMS emerges as a valuable tool in the domain of cluster analysis, holding great promise for diverse real-world applications. Future endeavors aim to broaden the scope by extending the utilization of neighborhood information to manifold learning. The exploration of geodesic distance in this context serves as a testament to the evolving versatility of FCMS.

In Chapter 6, the approach prioritized efficiency by eliminating the need for parameter tuning in generating dynamic fuzzy rule bases, aiming to demonstrate its effectiveness while minimizing computational load. Further enhancements in results could potentially be achieved through parameter tuning. However, it is important to note that this may become computationally prohibitive. As a concluding remark, we emphasize that synergistic relations, as demonstrated in our method, extend beyond gene interactions. Therefore, the proposed approach holds utility in uncovering insightful and interpretable non-linear relationships between features in diverse scenarios. An illustrative example includes its application in discovering meaningful patterns in online shopping behaviour. Moreover, the study was restricted to identifying only synergistic pairs. Extending the method to identify synergistic triplets presents a valuable direction for future research.

Bibliography

[1] R. O. Duda, P. E. Hart et al.,*Pattern classification*. John Wiley & Sons, 2006. 5, 19, 32

[2] H. Hotelling, "Relations between two sets of variates", *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936. 5

[3] R. A. Fisher, "The use of multiple measurements in taxonomic problems", *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936. 5

[4] H. Akaike, "Factor analysis and aic", *Hirotugu Akaike*. Springer, pp. 371–386, 1987. 5

[5] X. He and P. Niyogi, "Locality preserving projections", *Advances in neural information processing systems*, pp. 153–160, 2004. 5

[6] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Transactions on computers*, vol. 100, no. 5, pp. 401–409, 1969. 5, 100

[7] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction", *science*, vol. 290, no. 5500, pp. 2319– 2323, 2000. 5, 32

[8] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding", *science*, vol. 290, no. 5500, pp. 2323–2326, 2000. 5

[9] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", *science*, vol. 313, no. 5786, pp. 504–507, 2006. 5

[10] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne”, *Journal of machine learning research*, vol. 9, no. 11, pp. 2579–2605, 2008. 5

[11] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, “Feature selection: A data perspective”, *ACM computing surveys (CSUR)*, vol. 50, no. 6, pp. 1–45, 2017. 6, 7

[12] K. Nag and N. R. Pal, “A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification”, *IEEE transactions on cybernetics*, vol. 46, no. 2, pp. 499–510, 2016. 6, 7

[13] Q. Song, H. Jiang, and J. Liu, “Feature selection based on FDA and F-score for multi-class classification”, *Expert Systems with Applications*, vol. 81, pp. 22–27, 2017. 6

[14] J. R. Vergara and P. A. Estévez, “A review of feature selection methods based on mutual information”, *Neural computing and applications*, vol. 24, pp. 175–186, 2014. 6

[15] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy”, *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005. 6, 19

[16] P. A. Estévez, M. Tesmer, C. A. Perez, and J. M. Zurada, “Normalized mutual information feature selection”, *IEEE Transactions on neural networks*, vol. 20, no. 2, pp. 189–201, 2009. 6

[17] M. Tesmer and P. A. Estévezz, “AMIFS: Adaptive feature selection by using mutual information”, *IEEE International Joint Conference on Neural Networks*, pp. 303–308, 2004. 6, 7

[18] J. R. Vergara and P. A. Estévez, “CMIM-2: an enhanced conditional mutual information maximization criterion for feature selection”, *Journal of Applied Computer Science Methods*, vol. 2, no. 1, pp. 5–20, 2010. 6

[19] N. R. Pal and K. Chintalapudi, “A connectionist system for feature selection”, *Neural, Parallel and Scientific Computation*, vol. 5, No. 3, pp 359-381, 1997. 7, 20, 33, 34

[20] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection”, *Journal of machine learning research*, vol. 3, pp. 1157–1182, 2003. 7

[21] A. Jović, K. Brkić, and N. Bogunović, “A review of feature selection methods with applications”, *IEEE International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pp. 1200–1205, 2015. 7

[22] J. T. Chien, “Linear regression based bayesian predictive classification for speech recognition”, *IEEE transactions on speech and audio processing*, vol. 11, no. 1, pp. 70–79, 2003. 8

[23] A. Biem, “Minimum classification error training for online handwriting recognition”, *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 7, pp. 1041–1051, 2006. 8

[24] S. Arora, R. Mittal, H. Kukreja, and M. Bhatia, “An evaluation of denoising techniques and classification of biometric images based on deep learning”, *Multimedia Tools and Applications*, vol. 82, no. 6, pp. 8287–8302, 2023. 8

[25] S. Jiang, J. Hu, C. L. Magee, and J. Luo, “Deep learning for technical document classification”, *IEEE Transactions on Engineering Management*, 2022. 8

[26] W. Rawat and Z. Wang, “Deep convolutional neural networks for image classification: A comprehensive review”, *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017. 8

[27] M. Trupthi, S. Pabboju, and G. Narasimha, “Improved feature extraction and classification—sentiment analysis”, *IEEE International Conference on Advances in Human Machine Interaction (HMI)*, pp. 1–6, 2016. 8

[28] J. B. MacQueen, “Some Methods for classification and analysis of multivariate Observations”, *Proceedings of 5th Berkeley Symposium*

on Mathematical Statistics and Probability, University of California Press. pp. 281–297, 1967. 9

[29] F. Nielsen, "Hierarchical Clustering", *Introduction to HPC with MPI for Data Science*, Springer, pp. 195–211, 2016. 9

[30] J. C. Dunn,"A Fuzzy Relative of the ISODATA Process and its use in detecting compact well-separated clusters", *Journal of Cybernetics*. vol. 3, no. 3, pp. 32–57, 1973. 9

[31] J. Bezdek, *Pattern recognition with Fuzzy objective function algorithms*, Plenum Press, New York, 1981. 9, 11, 22, 36, 38, 62, 78, 119, 123, 125

[32] L. Magdalena, "Fuzzy rule-based systems", *Springer handbook of computational intelligence*, pp. 203–218, 2015. 15

[33] L. S. Riza, C. Bergmeir, F. Herrera, and J. M. Beníez, "frbs: Fuzzy rule-based systems for classification and regression in R", *Journal of statistical software*, vol. 65, pp. 1–30, 2015. 15

[34] J. Deng and Y. Deng, "Information volume of fuzzy membership function", *International Journal of Computers Communications & Control*, vol. 16, no. 1, 2021. 15

[35] T. Takagi, M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control". *IEEE Transactions on Systems, Man, and Cybernetics*. vol. SMC-15, issue 1, pp. 116–132, 1985. 16

[36] M. Masaeli, J. G. Dy and G. Fung, "From transformation-based dimensionality reduction to feature selection", *Proceedings of the 27th International Conference on Machine Learning*, pp. 751–758, 2010.

[37] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*, Springer, 2001.

[38] H. Liu and H. Motoda, *Computational Methods of Feature Selection*, Chapman and Hall/CRC Press, 2007. 18

[39] J.G. Dy and C.E. Brodley, "Feature subset selection and order identification for unsupervised learning", *Proc. 17th International Conference on Machine Learning*, Morgan Kaufmann, pp. 247–254, 2000. 18, 31

[40] S. Alelyani, J. Tang, and H. Liu, "Feature selection for clustering: A review. Data Clustering: Algorithms and Applications", *CRC Press*, 2013. 18

[41] J. Weston, A. Elisseff, B. Schoelkopf, and M. Tipping, "Use of the zero norm with linear models and kernel methods", *Journal of Machine Learning Research*, vol. 3, pp. 1439–1461, 2003. 18

[42] L. Song, A. Smola, A. Gretton, K. Borgwardt, and J. Bedo, "Supervised feature selection via dependence estimation", *International Conference on Machine Learning*, 2007. 18

[43] J.G. Dy and C.E. Brodley, "Feature selection for unsupervised learning", *The Journal of Machine Learning Research*, vol. 5, pp.845–889, 2004. 18

[44] P. Mitra, C. A. Murthy, and S. Pal, "Unsupervised feature selection using feature similarity", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp.301–312, 2002. 18

[45] Z. Zhao and H. Liu, "Semi-supervised feature selection via spectral analysis", *Proc. 7th SIAM Int. Conf. Data Mining*, pp. 641–646, 2007. 18, 20

[46] Z. Xu, R. Jin, J. Ye, M. Lyu, and I. King, "Discriminative semi-supervised feature selection via manifold regularization", *Proceedings of the 21th International Joint Conference on Artificial Intelligence*, 2009. 18

[47] P. Kohavi and G. H. John, "Wrappers for feature subset selection", *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997. 18, 36

[48] M. R. Sikonja and I. Kononenko, "Theoretical and empirical analysis of Relief and ReliefF", *Machine Learning*, vol. 53, pp. 23–69, 2003. 18

[49] Z. Zhao and H. Liu, "Spectral Feature Selection for Data Mining", *Chapman & Hall/Crc Data Mining and Knowledge Discovery*, Taylor & Francis, 2011. 18, 32

[50] M. Dash, K. Choi, P. Scheuermann, and Hu. Liu, "Feature selection for clustering - a filter solution", *Proceedings of the Second International Conference on Data Mining*, pp. 115–122, 2002. 19

[51] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection", *Advances in Neural Information Processing Systems*, 18:507, 2006. 19

[52] Y.S. Kim, W. N. Street and F. Menczer, "Evolutionary model selection in unsupervised learning", *Intelligent Data Analysis*, vol. 6, no. 6, pp. 531–556, 2002. 19

[53] Z. Wu, Y. Zhao, W. Wang and C. Li, "Adaptive weighted fuzzy clustering based on intra-cluster data divergence", *Neurocomputing*, vol. 552, 126550, 2023. 19

[54] J. Li, X. Gao and H. Ji, "A feature weighted FCM clustering algorithm based on evolutionary strategy", *Proceedings of the 4th World Congress on Intelligent Control and Automation*, Cat. No.02EX527), Shanghai, China, vol.2, pp. 1549-1553, 2002. 20

[55] H. Frigui, O. Nasraoui, "Unsupervised learning of prototypes and attribute weights", *Pattern Recognition* , vol. 37, no. 3, pp.567-581, 2004. 20, 36

[56] M. Yang, Y. Nataliani, "A Feature-reduction Fuzzy clustering algorithm based on feature-weighted entropy", *IEEE Transactions on Fuzzy systems*, vol. 26, no. 2, 2017. 20, 36

[57] M. Hashemzadeh, A. G. Oskouei and N. Farajzadeh, "New Fuzzy c-means clustering method based on feature-weight and cluster-weight learning", *Applied soft computing*, vol. 78, pp. 324-345, 2019. xxv, 20, 36, 45, 46, 47

[58] J. Zhou, C. L. Philip Chen, "Attribute weight entropy regularization in fuzzy C-means algorithm for feature selection", *Proceedings 2011 International Conference on System Science and Engineering*, 2011. 21, 36

21, 36, 37

[59] H. J. Xing, X. Wang and M. Ha, "A comparative experimental study of feature-weight learning approaches", *SMC*, pp. 3500–3505, 2011. 21, 36

[60] X. Wang, Y. Wang and L. Wang, "Improving fuzzy c-means clustering based on feature-weight learning", *Pattern Recognition Letters*, vol. 25, no. 10, pp. 1123–1132, 2004. 21

[61] D.S. Yeung, X.Z. Wang, Improving performance of similarity-based clustering by feature weight learning, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 4, pp. 556–561, 2002. 21

[62] L. Ling,L. Huang, J. Wang, Li. Zhang,Y. Wu,Y. Jiang and K. Xia2,3, "An improved soft subspace clustering algorithm for brain MRI segmentation", *Computer modeling in Engineering & Science*, vol. 137, no. 3, 2023. 21

[63] J. Liying, Z. Shengdun, Z. Congcong, G. We, Dou. Yao, Lu. Mengkang, "Adaptive soft subspace clustering combining within-cluster and between-cluster information", *Journal of Intelligent & Fuzzy Systems*, vol. 38, no. 3, pp. 3319-3330, 2020. 22

[64] Z. Li, C. Tang, X. Zheng, Z. Wan, K. Sun, W. Zhang and X. Zhu, "Mutual structure learning for multiple kernel clustering", *Information Sciences*, vol. 647, 119445, 2023. 22

[65] K. Ani Davis, M. Raj, "A novel approach to fuzzy c-Means clustering using kernel function", *Intelligent Decision Technologies*, vol. 16, no. 4, pp. 643-651, 2022. 22

[66] A. Bouchachia and W. Pedrycz, "Enhancement of fuzzy clustering by mechanisms of partial supervision", *Fuzzy Sets and Systems*, vol. 157, no. 13, pp. 1733–1759, 2006. 22

[67] J.H. Chiang and P.Y. Hao, "A new kernel-based fuzzy clustering approach: support vector clustering with cell growing", *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 4, pp. 518–527, 2003. 22

[68] D.W. Kim, K.Y. Lee, D. Lee, K.H. Lee, "Evaluation of the performance of clustering algorithms kernel-induced feature space", *Pattern Recognition*, vol. 38, no. 4, pp.607–611, 2005. 22, 62, 64, 78, 83

[69] H. Shen, J. Yang, S. Wang and X. Liu, "Attribute weighted Mercer kernel based fuzzy clustering algorithm for general non-spherical datasets", *Soft Computing*, vol. 10, no. 11, pp. 1061–1073, 2006. 22

[70] S. Singh and S. Srivastava, "Optimizing kernel possibilistic fuzzy C-means clustering using metaheuristic algorithms", *evolving systems*, 2023. 22

[71] D.Q. Zhang and S.C. Chen, "Clustering incomplete data using Kernel-based Fuzzy C-Means algorithm", *Neural Processing Letters*, vol. 18, no. 3, pp. 155–162, 2003. 22

[72] L. Zeyu, T. Shiwei, X. Jing and J. Jun, "Modified FCM clustering based on kernel mapping", *Proc. of the Internat. Society for Optical Engineering*, vol. 4554, pp. 241–245, 2001. 22

[73] D. Zhang and S. Chen, "Fuzzy clustering using kernel method", *Proc. of the Internat. Conf. on Control and Automation*, pp. 123-127, 2002. 22, 62, 64, 78, 83

[74] J. C. Bezdek and S. Pal, "Fuzzy models for pattern recognition: Methods that search for structures in data". *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 26, no 3, pp. 388-402, 1996. 22, 62, 64, 78, 83

[75] Y. Cheng, and H. Zhang, " Fuzzy c-means clustering and classification algorithms". *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. 28, no. 3, pp. 418-428, 1996. 22, 123

[76] I. Despotovic, B. Goossens and W. Philips, "MRI segmentation of the human brain: challenges, methods, and applications", *Computational and mathematical methods in medicine*, 2015. 22, 123

[77] S. K. Adhikari, J.K. Sing, D.K. Basu and M. Nasipuri, "Conditional spatial fuzzy c-means clustering algorithm for segmentation of MRI images", *Appl Soft Comput*, vol. 34, pp. 758-769, 2015. 23, 124

[78] D. L. Pham, “Fuzzy clustering with spatial constraints”, *Proceedings of IEEE International Conference on Image Processing*, vol. 2, 2002. 23, 124

23, 124

[79] K. S. Chuang, H. L. Tzeng, S. Chen, J. Wu, and T. J. Chen, “Fuzzy c-means clustering with spatial information for image segmentation”, *Comput Med Imaging Graph*, vol. 30, no. 1, pp. 9-15, 2006. 23, 124

[80] W. Cai, S. Chen and D. Zhang, “Fast and robust fuzzy c-means clustering algorithms incorporating local information for image segmentation”, *Pattern Recognition*, vol. 40, no. 3, pp. 825-838, 2007. 23, 124

[81] H. Y. Yu and J. L. Fan, “Three-level image segmentation based on maximum fuzzy partition entropy of 2-D histogram and quantum genetic algorithm”, *International Conference on Intelligent Computing*, Springer, pp. 484-493, 2008. 23, 124

[82] B. Caldairou, N. Passat, P. A. Habas, C. Studholme and F. Rousseau, “A non-local fuzzy segmentation method: application to brain MRI”, *Pattern Recogn*, vol. 44, no. 9, pp. 1916-1927, 2011 23, 124

[83] F. Zhao, L. Jiao and H. Liu, “Fuzzy c-means clustering with non local spatial information for noisy image segmentation”, *Frontiers of Computer Sciience in China*, vol. 5, no. 1, pp. 45-56, 2011. 23, 124

[84] W.Q. Deng, X. M. Li, X. Gao and C.M. Zhang, “A modified fuzzy c-means algorithm for brain MRI image segmentation and bias field correction”, *Journal of Computer Science Technology*, vol. 31, no. 3, pp. 501-511, 2016. 23, 124

[85] Kaushik Sarkar, Rajani K. Mudi and Nikhil R. Pal, “Weighted Fuzzy C-Means: Unsupervised Feature Selection to Realize a Target Partition”, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2024. (Revision submitted). 25

[86] K. Sarkar and N. R. Pal, "Is It Rational To Partition A Data Set Using Kernel-Clustering?", *IEEE International Conference on Fuzzy Systems*, FUZZ-IEEE, 27-30 June, Taipei, Taiwan, 2011. 26, 80, 92

[87] N. R. Pal and K. Sarkar, "What and When Can We Gain From the Kernel Versions of C-Means Algorithm?," *IEEE Transactions on Fuzzy Systems*, vol 22, no. 2, pp. 363-379, 2014. 26

[88] K. Sarkar and R. K. Mudi, "Fuzzy Clustering Exploiting Neighbourhood Information for Non-image Data", *International Journal of Computer Sciences and Engineering*, vol. 12, no. 2, 2024. 28

[89] K. Sarkar, P. Chatterjee and N. R. Pal, "Finding Synergy Networks From Gene Expression Data: A Fuzzy-Rule-Based Approach", *IEEE Transactions on Fuzzy Systems*, vol 24, issue 6, pp. 1488-1499, 2016. 28

[90] D. Chakraborty and N. R. Pal, "Selecting Useful Groups of Features in a Connectionist Framework", *IEEE Transactions on Neural Networks*, vol. 19, no. 3, pp. 381-396, 2008. 31, 32, 33, 34

[91] E. R. Dougherty, "Small sample issue for Microarray based classification", *Comparative Functional Genomics*, vol. 2, no. 1, pp. 28–34, 2001. 31

[92] I. Jolliffe. "Principal component analysis", *Wiley Online Library*, 2002. 32

[93] B. Scholkopft and K. Mullert. "Fisher discriminant analysis with kernels", *Neural networks for signal processing*, IX, 1:1, 1999. 32

[94] G. H. Golub and C. F. Van Loan, "Matrix computations", *JHU Press*, vol. 3, 2012. 32

[95] R. Tibshirani, "Regression shrinkage and selection via the lasso", *Journal of the Royal Statistical Society*. Series B (Methodological), vol. 58, no. 1, pp. 267–288, 1996. 32, 33

[96] T. M. Cover and J. A. Thomas, *Elements of information theory*, John Wiley & Sons, 2012. 32

[97] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines", *Machine Learning*, vol. 46, pp. 389–422, 2002. 33, 34

[98] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu, "Unsupervised feature selection using nonnegative spectral analysis", in *Proc. AAAI Conf. Artif. Intell.*, pp. 1026–1032, 2012. 33

[99] Z. Li, J. Liu, Y. Yang, X. Zhou, and H. Lu, "Clustering-guided sparse structural learning for unsupervised feature selection", *IEEE Transactions on Knowledge Data Engineering*, vol. 26, no. 9, pp. 2138–2150, 2013. 33

[100] J. Wang, H. Zhang, J. Wang, P. YiFei, N. R. Pal, "Feature Selection using a Neural Network With Group Lasso Regularization and Controlled Redundancy", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32(3), pp. 1110-1123, 2020. 33, 34

[101] H. Zhang, J. Wang, Z. Sun, J. M. Zurada, and N. R. Pal, "Feature Selection for Neural Networks Using Group Lasso Regularization", *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 4, pp. 659 - 673, 2020. 33, 34

[102] L. Zhao, Q. Hu, & W. Wang, "Heterogeneous feature selection with multi-modal deep neural networks and sparse group lasso", *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 1936-1948, 2015. 33, 34

[103] R. Chakraborty and N. R. Pal, "Feature Selection Using a Neural Framework with Controlled Redundancy", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1 pp 35-50, 2015. 33

[104] N. R. Pal, K. Aguan, A. Sharma and S. Amari, "Discovering biomarkers from gene expression data for predicting cancer subgroups using neural networks and relational fuzzy clustering", *BMC Bioinformatics*, 8:5, 2007. 34

[105] N. R. Pal, A. Sharma, S. Sanadhya and Karmeshu, "Identifying Marker Genes from Gene Expression Data in a Neural Framework

through Online Feature Analysis", *International Journal of Intelligent Systems*, vol. 21, no. 4, pp. 453-467, 2006. 34

[106] S. Agarwal, P. Ghanty, N. R. Pal, "Identification of a small set of plasma signaling proteins using neural network for prediction of Alzheimer's disease", *Bioinformatics*, vol. 1, no. 15, pp. 2505-2513, 2015. 34

[107] H. W. Wang, H. J. Sun, T. Y. Chang, H. H. Lo, W. C. Cheng, G. C. Tseng, C. T. Lin, S. J. Chang, N. R. Pal and I-Fang Chung, "Discovering monotonic stemness marker genes from time-series stem cell microarray data", BMC Genomics, 2015. 34

[108] H. Lee, C. Chen, J. Chen, and Y. Jou, "An efficient fuzzy classifier with feature selection based on fuzzy entropy", *IEEE Transactions on Systems, Man Cybernetics*, Part B: Cybern., vol. 31, no. 3, pp. 426–432, 2001. 34

[109] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection", *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003. 34

[110] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu, "Unsupervised feature selection using nonnegative spectral analysis", *Proceedings of the AAAI Conference on Artificial Intelliigence*, pp. 1026–1032, 2012. 36

[111] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "$L_{2,1}$-norm regularized discriminative feature selection for unsupervised learning", in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, pp. 1589– 1594, 2011. 36

[112] M. Dash, H. Liu, and J. Yao, "Dimensionality reduction for unsupervised data", *Proc. 19th IEEE Int. Conf. Tools with Artif. Intell.*, pp. 532–539, 1997. 36

[113] R. Varshavsky, A. Gottlieb, M. Linial, and D. Horn, "Novel unsupervised feature filtering of biological data", *Bioinformatics*, vol. 22, no. 14, pp. e507–e513, 2006. 36

[114] M. Banerjee and N. R. Pal, "Unsupervised Feature Selection with Controlled Redundancy (UFeSCoR)", *IEEE Transactions on knowl-*

edge and data engineering, vol. 27, no. 12, pp. 3390-3403, 2015. 36, 48

[115] L. Sun, L. Wang, W. Ding, Y. Qian and J. Xu, “Feature Selection Using Fuzzy Neighborhood Entropy-Based Uncertainty Measures for Fuzzy Neighborhood Multigranulation Rough Sets”, *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 1, pp. 19-33, 2021. 36

[116] J. Handl and J. D. Knowles, “Semi-supervised feature selection via multiobjective optimization”, *Proc. Int. Joint Conf. Neural Netw.*, pp. 3319–3326, 2006. 36

[117] J. Ren, Z. Qiu, W. Fan, H. Cheng, and P. S. Yu, “Forward semi-supervised feature selection”, in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*, pp. 970–976, 2008. 36

[118] J. Zhao, K. Lu, and X. He, “Locality sensitive semi-supervised feature selection”, *Neurocomputing*, vol. 71, no. 10-12, pp. 1842–1849, 2008. 36

[119] M. S. Yang and J. B. M. Benjamin, “Feature-Weighted Possibilistic c-Means Clustering With a Feature-Reduction Framework”, *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 5, pp. 1093-1106, 2021. 37

[120] J. C.Bezdek, J. M. Keller and R. Krishnapuram, “Fuzzy Models and Algorithms for Pattern Recognition and Image Processing”, *Springer-Verlag New York Inc*, 1999. 37, 81, 85, 157

[121] Y. Mojtaba and S. Dick. “Classification via deep fuzzy c-means clustering”. *IEEE international conference on fuzzy systems* (FUZZ-IEEE), pp. 1-6, 2018. 37

[122] G. Maoguo, Y. Zhao, H. Li, A. K. Qin, L. Xing, J. Li, Y. Liu, and Y. Liu. “Deep Fuzzy Variable C-Means Clustering Incorporated with Curriculum Learning”. *IEEE Transactions on Fuzzy Systems*, vol. 31, no. 12, 2023. 37

[123] F. Hoppner, F. Klawonn, R. Kruse and T. Runkler, *Fuzzy Cluster Analysis*. Wiley, Chichester, 1999. 38

[124] J. Wang, S. Wang, F. Chung, and Z. Deng. “Fuzzy partition based soft subspace clustering and its applications in high dimensional data”, *Information Sciences*, 246, pp.133-154, 2013. 39, 40, 56

[125] Y. Chan, W. Ching, M.K. Ng, J.Z. Huang, “An optimization algorithm for clustering using weighted dissimilarity measures”, *Pattern Recognition*, vol. 37, no. 5, pp. 943–952, 2004. 40

[126] G. Gan and J. Wu, “A convergence theorem for the fuzzy subspace clustering (FSC) algorithm”, *Pattern Recognition*, vol. 41, no. 6, pp. 1939–1947, 2008. 40

[127] L. Jing, M. Ng, J. Huang, “An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data”, *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 8, pp. 1026–1041, 2007. 40

[128] V. Singh and N. K. Verma, “Gene Expression Data Analysis Using Feature Weighted Robust Fuzzy-Means Clustering”, *IEEE Transactions on NanoBioscience*, vol. 22, no. 1, pp.99-105, 2022. 40, 56

[129] E. E. Gustafson and W. Kessel, “Fuzzy Clustering with a Fuzzy Covariance Matrix”, *IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes*, pp. 761-766, 1979. 41, 62, 78

[130] M. Bilenko, S. Basu, R. J. Mooney, “Integrating constraint and metric learning in semi-supervised clustering”, *Proceedings of the 21st International Conference on Machine Learning*, pp. 81-88, 2004. 41

[131] I. S. Dhillon and D. S. Modha, “Concept decompositions for large sparse text data using clustering”, *Machine Learning*, vol. 42, pp. 143-175, 2001. 42

[132] *https://archive.ics.uci.edu* xxix, 45, 53, 104, 132

[133] H. Zhu, M. Zhou and R. Alkins, “Group Role Assignment via a Kuhn–Munkres Algorithm-Based Solution”, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 42, no. 3, 2012. 48, 50

[134] A. Strehl and J. Ghosh, "Cluster Ensembles-A knowledge reuse framework for combining multiple partitions", *Journal of Machine Learning Research*, vol. 3, pp. 583-617, 2002. 49, 79, 94, 130

[135] D. G. fisher, "The adjusted rand statistics: A SAS MACRO", *PSYCHOMETRICA*, vol. 53, no. 3, pp. 417-423, 1988. 49, 79, 94, 130

[136] M.R. Anderberg, *Cluster Analysis for Application*, Academic Press, New York,1973. 62, 78

[137] E. Backer and A.K. Jain, "A clustering performance measure based on fuzzy set decomposition", *IEEE Transaction Pattern Analysis and Machine Intelligence*. vol. 3, no. 1, pp. 66-74, 1981. 62, 78

[138] I.S. Dhillon and Y. Guan, B. Kulis, "A unified view of kernel k-means, spectral clustering and graph partitioning", *Technical Report*, TR-04-25, UTCS, 2005. 62

[139] N. R. Pal, K. Pal, J. M. Keller and J. C. Bezdek, "A Possibilistic Fuzzy c-Means Clustering Algorithm", *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 4, 2005. 62, 78

[140] D.Q. Zhang and S.C. Chen, "Clustering incomplete data using Kernel-based Fuzzy C-Means algorithm", *Neural Processing Letters*, vol. 18, no. 3, pp. 155–162, 2003.. 62, 64, 78, 81

[141] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 2003. 62

[142] J. C. Bezdek, R. J. Hathaway and N. R. Pal, "Norm-induced shell-prototypes (NISP) clustering", *Neural, Parallel and Scientific Computations* , vol. 3, pp. 431-450, 1995. 62, 78

[143] R. Krisnapuram, H. Frigui and O. Nasroui, "Fuzzy and possibilistic shell clustering algorithms and their application to boundary detection and surface approximation", *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 1, 1995. 62, 78

[144] R. N. Dav and K. Bhaswan, "Adaptive fuzzy c-shells clustering and detection of ellipses", *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 643-662, 1992. 62, 78

[145] D. W. Kim, K. Y. Lee, D. Lee and K. H. Lee, "Evolution of the performance of clustering algorithms kernel-induced feature space", *Pattern Recognition*, vol. 38, no. 4, pp. 607-611, 2005. 62, 78

[146] Z. D. Wu, W. X. Xie and J. P.Yu, "Fuzzy c-means clustering algorithm based on kernel method", *Proc. of the Internat. Conf. on Computational Intelligence and Multimedia Applications*, pp. 49-54, 2003. 62, 78

[147] S. Zhou and J. Gan, "Mercer kernel fuzzy c-means algorithm and prototypes of clusters", *Proceedings of Conf. on Internat. Data Engineering and Automated Learning*, vol. 3177, pp. 613–618, 2004. 62, 64, 65, 78, 83, 84

[148] D. Graves and W. Pedrycz, "Kernel-based fuzzy clustering and fuzzy clustering: A comperative experimental study", *Fuzzy Sets and Systems*, vol. 161, pp. 522-543, 2010. 62, 64, 65, 67, 74, 78, 79, 81, 83, 84, 86, 104

[149] R. Herbrich, *Learning Kernel Classifiers*, MIT Press, Cambridge, MA, 2002. 62, 64, 78, 81

[150] K.R. Muller, S. Mika, G. Ratsch, K. Tsuda and B. Scholkopf, "An introduction to kernel-based learning algorithms", *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001. 62, 64, 78, 81

[151] B. Scholkopf, A. Smola and K. R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem", *Neural Computation*, vol. 10, pp. 1299–1319, 1998. 62, 64, 78, 81

[152] H. Shen, J. Yang, S. Wang, X. Liu, "Attribute weighted Mercer kernel based fuzzy clustering algorithm for general non-spherical datasets", *Soft Computing*, vol. 10, no. 11, pp. 1061–1073, 2006. 64, 78, 81

[153] J D MacCuish and N E. MacCuish, *Clustering in Bioinformatics and Drug Discovery*, CRC Press, 2010. 78

[154] U. V.Luxburg, R. C. Williamson and I. Guyon," Clustering: Science or Art?", *Journal of Machine Learning Research - Proceedings*, Track 27, pp. 65-80, 2012. 78

[155] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall and M. Palaniswami, “Fuzzy c-Means Algorithms for Very Large Data”, *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 6, pp. 1130 - 1146, 2012. 78

[156] O. Linda and M. Manic, “General Type-2 Fuzzy C-Means Algorithm for Uncertain Fuzzy Clustering”, *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 5, pp. 883 - 897, 2012. 78

[157] L Chen, L. P. Chen and M Lu, “A Multiple-Kernel Fuzzy c-Means for image segmentation”, *IEEE Transactions on SMC-B*, vol. 41, no. 5, pp. 1263-1274, 2011. 78

[158] E. Hullermeier, M. Rifqi, S. Henzgen and R. Senge, “Comparing Fuzzy Partitions: A Generalization of the Rand Index and Related Measures”, *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 3, pp. 546-556, 2012. 79, 130

[159] U. V. Luxburg, *Clustering Stability: An Overview*, Now Publishers Inc, 2010. 79, 94

[160] O. Shamir and N. Tishby, “Stability and model selection in k-means clustering”, *Machine Learning*, 80(2-3), pp. 213-243, 2010. 79

[161] N. X. Vinh, J. Epps and J. Bailey, “Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance”, *Journal of Machine Learning Research*, vol. 11, pp. 2837-2854, 2010. 79

[162] J. Kleinberg, “An impossibility theorem for clustering”, *Proceedings of International Conferences on Advances in Neural Information Processing Systems*, pp. 463-470, 2003. 80

[163] M. Ackerman, “Towards Theoretical Foundations of Clustering”, PhD Thesis, University of Waterloo, Dept. of Computer Science, 2012. 80

[164] H. C. Huang, Y. Y. Chuang and C. S. Chen, “Multiple kernel fuzzy clustering”, *IEEE Transactions on fuzzy systems*, vol. 20, no. 1, pp. 120 - 134, 2012. 80, 104, 107

[165] B. Zhao, J. Kwok, and C. Zhang, "Multiple kernel clustering", *Proc. 9th SIAM Int. Conf. Data Mining*, pp. 638-649, 2009 80

[166] H. Zhang, H. Li and N. Chen, "Novel fuzzy clustering algorithm with variable multi-pixel fitting spatial information for image segmentation", *Pattern recognition*, vol. 121, 2022. 125

[167] C. Wang, W. Pedrycz, Z. Li, M. Zhou and S. S. Ge, "G-image segmentation: similarity-preserving fuzzy c-means with spatial information constraint in wavelet space"., *IEEE Transactions on Fuzzy systems*, pp: 3887-3898, 2020. 125

[168] M. N. Ahmed, S. M. Yamany, N. Mohamed, A. A. Farag, and T. Moriarty, "A modified fuzzy C-means algorithm for bias field estimation and segmentation of MRI data," *IEEE Trans. Med. Imaging*, vol. 21, pp. 193-199, 2002. 126, 127

[169] S. Chen and D. Zhang, "Robust Image Segmentation Using FCM With Spatial Constraints Based on New Kernel-Induced Distance Measure", *IEEE Transaction on Systems, MAN, and CYBERNETICS—PART B: CYBERNETICS*, vol. 34, no. 4, 2004. 127

[170] I. F. Chung, Y. C. Chen, and N. R. Pal, "Feature selection with controlled redundancy in a fuzzy rule-based framework", *IEEE Transactions on Fuzzy Systems*, 10.1109/TFUZZ.2017.2688358, vol. 26, no. 2, pp. 734-748, 2017. 143

[171] Y. C. Chen, N. R. Pal and I. F. Chung, "Integrated mechanism for feature selection and fuzzy rule extraction for classification", *IEEE Transactions on Fuzzy Systems*, vol. 20, Issue: 4, Page(s): 683 - 698, 2012. 143

[172] A. Laha, N. R. Pal and J. Das, "Land cover classification using fuzzy rules and aggregation of contextual information through evidence theory", *IEEE Trans. Geoscience and Remote Sensing*, vol. 24, No. 6, pp. 1633-1641, 2006. 143

[173] S. Das and N. R. Pal, " Nonlinear dimensionality reduction for data visualization: An unsupervised fuzzy rule-based approach". *IEEE Transactions on Fuzzy Systems*, vol. 30(7), pp. 2157-2169, 2021. 143

[174] M. Bansal, V. Belcastro, A. A. Impiombato and D. D. Bernardo, " How to infer gene networks from expression profiles", *Molecular System Biology*, vol. 3, no. 78, 2007. 144

[175] J. Pearl, *Probabilistic Reasoning Intelligent Systems: Networks of Plausible Inference*, San Francisco, CA: Morgan Kaufmann Publishers, 1988. 144

[176] N. Friedman, "Inferring cellular networks using probabilistic graphical models", *Science*, vol. 303, no. 5659, pp. 799-805, 2004. 144

[177] A. J. Butte and I. S. Kohane, "Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements", *Pacific Symposium Biocomputing*, pp. 415-426, 2000. 144

[178] K. Basso, A. A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera and A. Califano, "Reverse engineering of regulatory networks in human B cells", *Nature Genetics*, vol. 37, pp. 382-390, 2005. 144

[179] H. Kishino and P. J. Waddell, "Correspondence analysis of genes and tissue types and finding genetic links from microarray data", *Genome Informatics*, vol. 11, pp. 83-95, 2000. 144

[180] J. Schafer and K. Strimmer, "An empirical Bayes approach to inferring large-scale gene association networks," *Bioinformatics*, vol. 21, pp. 754-764, 2005. 144

[181] O. Gevaert, F. D. Smet, D. Timmerman, Y. Moreau and B. D. Moor,"Pre-dicting the prognosis of breast cancer by integrating clinical and microarray data with Bayesian networks", *Bioinformatics*, vol. 22, pp. 184-190, 2006. 144

[182] V. Griffith and C. Koch,"Quantifying synergistic mutual information", *arXiv preprint arXiv*, 1205.4265, 2012. 148, 149

[183] E. Schneidman, W. Bialek and M. J. Berry II, "Synergy, Redundancy, and Independence in Population Codes", *The Journal of Neuroscience*, vol. 23, no. 37, pp. 11539-53, 2003. 148

[184] G. Itay and N. Tishby, "Synergy and redundancy among brain cells of behaving monkeys", *Proceedings of the 1998 conference on Ad-*

vances in neural information processing systems II, pp.111-117, 1998. 148

[185] T. M. Cover and J. A. Thomas, *Elements of information theory, 2nd edition*, Wiley Interscience, 2006. 148

[186] J. Watkinson, X. Wang, T. Zheng and D. Anastassiou, “Identification of gene interactions associated with disease from gene expression data using synergy networks”, *BMC Systems Biology*, vol. 2, no. 10, 2008. xxxi, 149, 150, 162, 163, 167, 169, 173

[187] D. Anastassiou, “Computational analysis of the synergy among multiple interacting genes”, *Molecular System Biology*, vol. 3, no. 83, 2007. 149, 152

[188] V. Varadan, D. M. Miller 3rd andD. Anastassiou, “Computational infer-ence of the molecular logic for synaptic connectivity in C. elegans”, *Bioinformatics*, vol. 22, no. 14, pp. e497-e506, 2006. 149

[189] http://www.genome.wi.mit.edu/MPR/prostate. 149, 151

[190] V. Varadan and D. Anastassiou, “Inference of disease-related molecular logic from systems-based microarray analysis”, *PLoS Computational Biology*, vol. 2, no. 6, pp. e68, 2006. 149, 153

[191] http://www.cpdr.org/basic-science/gene-data.php. 151

[192] M. Sugeno and T. Yasukawa, “A fuzzy-logic-based approach to qualitative modeling”, *IEEE Transactions on fuzzy systems*, vol. 1, no. 1, pp. 7-31, 1993. 153

[193] H. Ishibuchi, K. Nozaki, N. Yamamoto and H. Tanaka, “Selecting fuzzy if-then rules for classification problems using genetic algorithms”, *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 3, pp. 260-270, 1995. 153, 154

[194] Y. C. Chen, N. R. Pal and I. Chung, “An integrated mechanism for feature selection and fuzzy rule extraction for classification”, *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 683-698, 2012. 153

[195] N. R. Pal and S. Saha, “Simultaneous structure identification and fuzzy rule generation for Takagi-Sugeno models”, *IEEE Transac-*

tions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 38, no. 6, pp. 1626-1638, 2008. 153

[196] M. Rotunno, N. Hu, H. Su, C. Wang, A. M. Goldstein, A. W. Bergen1, D. Consonni, A. C. Pesatori, P. A. Bertazzi, S. Wacholder, J. Shih, N. E. Caporaso, P. R. Taylor and M. T. Landi, "A Gene Expression Signature from Peripheral Whole Blood for Stage I Lung Adenocarcinoma", *Cancer Prevention Research*, vol. 4, no. 10, pp. 1599-1608, 2011. 174

[197] R. Melissa, N. Hu, H. Su, C. Wang, A. M. Goldstein, A. W. Bergen, D. Consonni et al. "A gene expression signature from peripheral whole blood for stage I lung adenocarcinoma", *Cancer Prevention Research*, vol. 4, no. 10, pp. 1599-1608, 2011. 174

[198] T. Rajkumar, N. Vijayalakshmi, G. Gopal, K. Sabitha, S. Shirley, U. M. Raja and S. A. Ramakrishnan, "Identification and validation of genes involved in gastric tumorigenesis", *Cancer Cell International*, vol. 10:45, 2010. 174

[199] M. Munz, P. A. Baeuerle, and O. Gires, "The emerging role of EpCAM in cancer and stem cell signaling", *Cancer research*, vol. 69, no. 14, pp. 5627-5629, 2009. 174

[200] M. Saleem, M. H. Kweon, J. J. Johnson, V. M. Adhami, I. Elcheva, N. Khan, B. B. Hafeez, K. M. Bhat, S. Sarfaraz, S. Reagan-Shaw, V. S. Spiegelman, V. Setaluri and H. Mukhtar, "S100A4 accelerates tumorigenesis and invasion of human prostate cancer through the transcriptional regulation of matrix metalloproteinase 9", *Proceedings of National Academy of Sciences of U S A*,vol. 103, no. 40, pp. 14825–14830, October 2006. 174

[201] N. I. Ismail, G. Kaur, H. Hashim and M. S. Hassan, "S100A4 overexpression proves to be independent marker for breast cancer progression", *Cancer Cell International*, vol. 8, no. 12, 2008. 174

[202] S. Gupta, T. Hussain, G. T. MacLennan, P. Fu, J. Patel, and H. Mukhtar, "Differential Expression of S100A2 and S100A4 During Progression of Human Prostate Adenocarcinoma", *Journal of Clinical Oncology*, vol. 21, no. 1, pp. 106-112, 2003. 174

[203] L. Ragolia, T. Palaia, E. Paric, and J. K. Mae-saka, "Elevated l-pgds activity contributes to pma-induced apoptosis concomitant with downregulation of pi3-k", *American Journal of Physiology-Cell Physiology*, vol. 284, no. 1, pp. C119-C126, 2003. 175

[204] K. Nithipatikom, M. A. Isbell, P. F. Lindholm, A. Kajdacsy-Balla, S. Kaul, and W. B. Campell, "Re-quirement of cyclooxygenase-2 expression and prostaglandins for human prostate cancer cell invasion", *Clinical and experimental metastasis*, vol. 19, pp. 593-601, 2002. 175

[205] L. Ge, J. Gao, N. Du and A. Zhang, "Finding informative genes for prostate cancer: a general framework of integrating heterogeneous sources", *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, pp. 274-281, 2012. 175

[206] T. Fujimoto, K. Yoshimatsu, K. Watanabe, H. Yokomizo, T. Otani, A. Matsumoto, G. Os-awa, M. Onda, and K. Ogawa, "Overexpression of human x-box binding protein 1 (xbp-1) in colorectal adenomas and adenocarcinomas", *Anticancer research*, vol. 27, no. 1A, pp. 127-131, 2007. 175

[207] Y. Liu, M. Adachi, S. Zhao, M. Hareyama, A. C. Koong, D. Luo, T. A. Rando, K. Imai, and Y. Shinomura, "Preventing oxida-tive stress: a new role for xbp1", *Cell Death and Differentiation*, vol. 16, no. 6, pp. 847-857, 2009. 175

[208] X. Chen, D. Iliopoulos, Q. Zhang, Q. Tang, M. B. Greenblatt, M. Hatziapostolou, E. Lim et al. "XBP1 promotes triple-negative breast cancer by controlling the HIF1 [agr] pathway", *Nature*, vol. 508, pp. 103-107, 2014. 175

[209] http://genetrail.bioinf.uni-sb.de/. 176

[210] D. Hanahan and R. A. Weinberg, "Hallmarks of cancer: The next generation", *Cell*, vol. 144, no. 5, pp. 646-74, 2011. 176

[211] I. Mellman and Y. Yarden, "Endocytosis and Cancer", *Cold Spring Harb Perspect Biology*, vol. 5, no. 12, 2013. 176

[212] P. Limonta, M. M. Montagnani , S. Mai, M. Motta, L. Martini and R. M. Moretti, “GnRH receptors in cancer: from cell biology to novel targeted therapeutic strategies”, *Endocr Rev.* vol. 33, no. 5, pp.784-811, 2012. 176

[213] H. NE and G. MacDonald, “ErbB receptors and signaling pathways in cancer”, *Current Opinion in Cell Biology*, vol. 21, no. 2, pp. 177-84, April 2009. 176

[214] N. Normanno, D. L. Antonella , C. Bianco , S. Luigi, M. Mario, R. M. Monica , A. Carotenuto, D. F. Gianfranco , F. Caponigro, and D. S. Salomon, “Epidermal growth factor receptor (EGFR) signaling in cancer”, *Gene*, vol. 366, no. 1, pp. 2-16, 2006. 176

[215] P. Dutta and W. X. Li, “Role of the JAK-STAT Signalling Pathway in Cancer”, eLS, Wiley Online Library, 2013. 177

[216] W. Vainchenker and S. N. Constantinescu, “JAK/STAT signaling in hematological malignancies”, *Oncogene*, vol. 32, no. 21, pp. 2601-2613, 2013. 177

[illegible]
29/2/24