Application of K – Means Clustering Technique to Predict Rainfall in Forthcoming Years

Faculty of Engineering & Technology, Jadavpur University in fulfillment of the requirements for the Degree of Master of Computer Applications

Submitted by

Anisha Das

Registration No. : 120476 of 2012-13

Class Roll No. : 001610503013

Examination Roll No. : MCA196011

Under the supervision of

Dr. Sarmistha Neogy Lahiri

Professor, Dept. of Computer Science & Engineering

Jadavpur University

A thesis submitted in partial fulfilment of the requirements for degree of Master of Computer Application

Department of Computer Science & Engineering

Faculty of Engineering & Technology

Jadavpur University

May 2019

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work as part of my MCA studies and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Name : **Anisha Das**

Registration No. : 120476 of 2012-13

Class Roll No. : 001610503013

Examination Roll No. : MCA196011

Project Report Title : Application of K-means clustering technique to predict rainfall in forthcoming years

_____

Anisha Das

To whom it may concern

This is to certify that the work in this project report entitled as "Application of K-means clustering technique to predict rainfall in forthcoming years**" has** been satisfactorily completed by **Anisha Das.** It is a bona-fide piece of work for the fulfillment of the requirements for the degree of Master of Computer Application, of the Department of Computer Science & Engineering, Faculty of Engineering & Technology, Jadavpur University, during the academic year 2018-19.

_____

Dr. Sarmistha Neogy Lahiri
(Supervisor)
Professor
Department of Computer Science and Engineering
Jadavpur University

_____

Dr. Mahantapas Kundu
(Head of the Department)
Professor
Department of Computer Science and Engineering
Jadavpur University

_____

Dr. Chiranjib Bhattacharjee
(Dean)
Professor
Department of Computer Science and Engineering
Jadavpur University

Certificate of Approval

This is to certify that the project titled “Application of K-means clustering technique to predict rainfall in forthcoming years“ is a bona-fide record of work carried out by Anisha Das in partial fulfillments for the award of the degree of Master of Computer Application, of the Department of Computer Science & Technology, Jadavpur University during the period January 2019 to May 2019. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve this thesis only for the purpose for which it is submitted.

_____ _____

Signature of the Examiner Signature of the Examiner

Date: _____ Date:_____

Acknowledgement

With my sincere respect and gratitude, I would like to thank my advisor and project guide **Dr. Sarmistha Neogy Lahiri** for her continuous support for this project work, for her patience, motivation and enthusiasm. Her guidance helped me throughout the duration of the work. Her valuable suggestions inspired me a lot. I feel deeply honored that I got the opportunity to work under her guidance.

Regards,

Anisha Das

Examination Roll No. : MCA196011

Master of Computer Application

Jadavpur University

CONTENTS

INTRODUCTION

Agriculture in India has a giant history. Today, India ranks second global in farm output. The financial contribution of agriculture to India's GDP is gradually declining with the country's huge – based monetary growth. Rainfall and crops production and prediction has been the most important and challenging role in the matter of all living beings. It also plays an essential position for countries' economic growth and development in the modern world. Science and computer technology together has made significant advances over the past several years and using those advanced technologies and few past patterns, helps in growing the ability to predict the future. Accurate information on rainfall is essential for the planning and management of water resources and farming. Previously farmers predict the production on the foundation of assumption of experience, farmers except any assist of laptop as well as tender computing technology. The advances in computing and statistics storage have supplied widespread almost of data.

The challenge has been taken to extract expertise from this raw data, Data Mining can bridge the knowledge of the statistics to the crop yield estimation. Data Mining is known as the process of analyzing data to extract interesting patterns and knowledge. It is used for analysis purpose to analyze different type of data by using available data mining tools. This information is being currently used for wide range of applications like customer retention, education system, production control, healthcare, market basket analysis, manufacturing engineering, scientific discovery and decision making etc. It is being studied for different databases like object-relational databases, relational database, data warehouses and multimedia databases, etc. Clustering is a data mining technique of grouping set of data objects into multiple groups or clusters so that objects within the cluster have high similarity, but are very dissimilar to objects in the other clusters. Clustering algorithms are used to organize data, categorize data, for data compression and model construction, for detection of outliers etc.

In general, weather and rainfall are highly non-linear and complex phenomena, which require advanced computer modeling and recreation for their accurate prediction. An Artificial Neural Network (ANN) can be used to foretell the behavior of such non-linear systems. There is a range of algorithms for prediction purpose. Hence the present investigation primarily involves data mining techniques. Firstly, in clustering, we used the K-means, one of the simplest and best method for prediction work. Second, we used the filtering technique. And finally we used, artificial neural network algorithm. In that we conclude that, these three techniques are useful in predicting the way of excellent and accurate result of agricultural crop production based on rainfall prediction, as it is being compared with the other algorithm. This research will help the farmers to know how much production will come in the next coming season and how much amount of rainfall will occur so that the farmers get awareness and they can manage themselves from huge loss.

This project work presents a methodology for forecasting rainfall of various regions of our country through clustering. This methodology uses a rainfall database for the prediction of

upcoming years being described in later chapters. The project is organized into four chapters. 1st chapter consists of explanation of the topics being used in the project, 2nd chapter elaborates the work done previously by the researchers, a brief background history is covered in this section. In 3rd chapter, methodology of this project is being mentioned along with the coding details. Also the proposed model is being explained in this section. In 4th chapter, the proposed technique is applied over the test dataset and results are captured. Also, system configuration, database used, language versions are mentioned here. 5th chapter includes conclusion and future scope of the proposed work.

BACKGROUND

Data mining techniques are being used as a domain and also used the median filtering technique for smoothening the data and we used the algorithm like multilayer neural network. Data mining is one of the process of extracting the useful information from huge amount of data.
Some of the aspects of Data Mining:

Outliers - It is a part of data not the trend of the data. When we provide a large set data, in that, some of the data are outliers and this is an exception of the data, so if the data contains outliers we cannot extract the meaningful information from data.

Filtering – Filtering means to find out if there are outliers of data and remove those outliers.

Clustering – Process of partitioning the data (or objects) into the same class is known as Clustering. The data in one class is more similar to each other than to those in other cluster.

Algorithm – Artificial Neural Network is defined as it contains several internal layers between the input and output layers, those internal layers are called hidden layers. When we append the loaded data, at that time artificial neural network can be used. ANNs are composed of multiple nodes, which take input data and perform simple operations. Each link between the nodes is associated with weight.

K-MEANS CLUSTERING*:*

We need and use k - means algorithm because it works on unlabeled numerical data and it will automatically and quickly group them together into k clusters. If we don't know how many groups we want, it's problematical, because K-means needs a specific number k of clusters in order to use it. As a result, whenever we have to optimize and solve a problem, we should know our data and on what basis we want to group them. Then we will be able to determine the no. of clusters we need.

K – means algorithm is composed of 3 steps:

Step 1 -> Initialization**:** First thing k-means does is, randomly choose K examples (data points) from the dataset as initial centroids (centres of clusters) since it does not know yet where the center of each cluster is.

Step 2 -> Cluster Assignment**:** All the data points that are closest to a centroid will create a cluster. If we are using the Euclidean distance between data points and every centroid, a straight line is drawn between two centroids, then a perpendicular bisector divides this line into clusters.

Step 3 -> Move the centroid**:** As we get new clusters, we need centroids. A centroid's new value will be the mean of all the examples in a cluster.

We keep on repeating steps 2 and 3 till the centroids stop moving, or in other words K-means algorithm gets converged.

Algorithm ->

randomly choose k examples as initial centroids
while true:
    create k clusters by assigning each example to closest centroid
    compute k new centroids by averaging examples in each cluster
    if centroids don't change:
        break

K-means is a fast and efficient method, because the complexity of one iteration is k*n*d where k is number of clusters, n is number of examples and d is time of computing the Euclidean distance between 2 points.

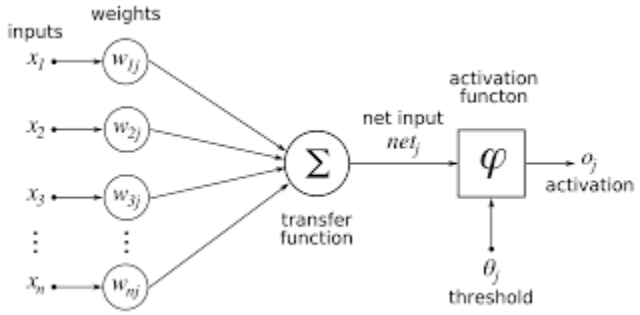
We try different values of k, evaluate them and choose the best k value as follows:

best = kMeans(points)
for t in range(numTrials):
    C = kMeans(points)
    if dissimilarity(C) < dissimilarity(best):
        best = C
return best

Dissimilarity(C) is the sum of all the variabilities of k clusters. **Variability** is the sum of all Euclidean distances between the centroid and each example in the cluster.

ARTIFICIAL NEURAL NETWORK

It is an artificial neural network that tries to mimic or copy the function of human neural systems. It is a complex, non-linear and parallel information processing system and is capable to organize its structural constituents to perform certain computations many times faster than the conventional computers today. Structure unit model of a neuron :--



Machine Learning in ANNs:

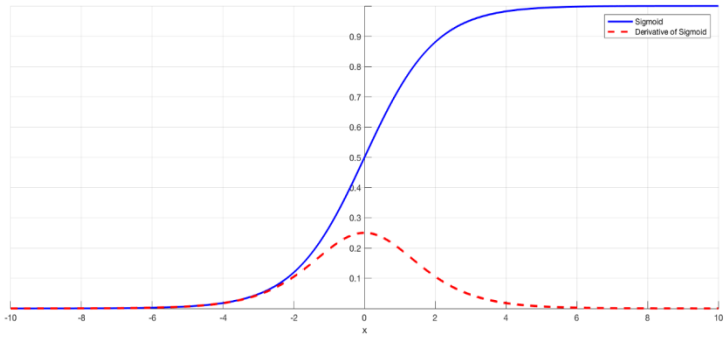
ANNs are capable of learning and they need to be trained. Several learning strategies are:

1) **Supervised Learning** (Classification) – It involves a teacher that is scholar than the ANN itself. For example, Pattern recognizing. The ANN comes up with guesses while recognizing, the teacher provides the ANN with answers. Network then compares and guesses with teacher's 'correct' answers and make adjustments according to errors. Here, new data is classified based on the training set.
2) **Unsupervised Learning** (Clustering) – It is required when there is no example dataset with known answers. For example, searching for a hidden pattern. In this case, clustering i.e., dividing a set of elements into groups according to some unknown pattern is carried out based on the existing datasets present.
3) **Reinforcement Learning** – This strategy is built on observation. The ANN makes a decision by observing its environment. If the observation is negative, network adjusts its weights to be able to make a different required decision the next time.

Feed Forward Neural Network

It is an ANN wherein connections between the nodes do not form a cycle. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes and to the output nodes. The simplest kind of neural network is a single layer perceptron network, which consists of single layer of output nodes, the inputs are fed directly to the outputs via a series of weights. The sum of the products of the weights and the inputs is calculated in each node, and if the value is above some threshold (typically 0), the neuron fires and takes the activated value (typically 1), otherwise takes the deactivated value (typically -1).

Perceptrons can be trained by a simple learning algorithm that is usually known as Delta Rule. It calculates the errors between the calculated output and sample output data and uses this to create an adjustment to the weights. Single-layer perceptrons are only capable of learning linearly separable patterns. A single-layer neural network can compute a continuous output instead of a step function. A logistic function $x = 1/ (1 + e^{-x})$ is being used. With this function, the single-layer network is identical to the logistic regression model, widely used in Statistical Modeling. The logistic function is also known as Sigmoid function. It has a continuous derivative, which allows it to be used in Back propagation. The graph for the Sigmoid function is --



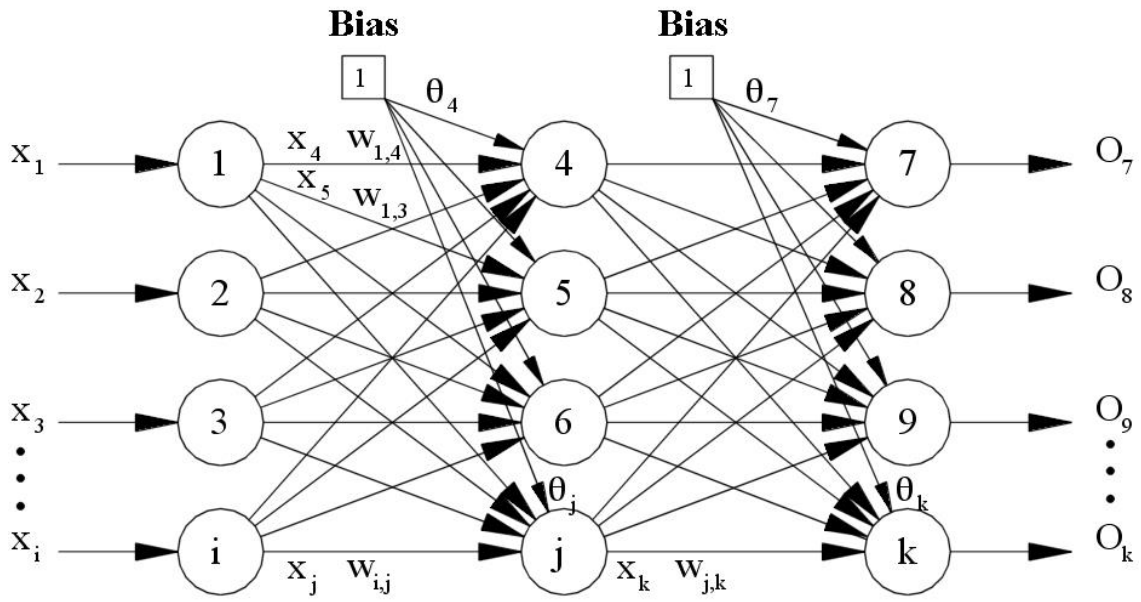
A two- layer neural network capable of calculating XOR. This class of networks consists of multiple layers of computational units, usually inter-connected in a feed-forward way. Multi-layer networks use a variety of learning techniques, the most popular being Back-propagation.

Back-Propagation

Back-propagation is used to calculate gradient of error of the network with respect to modifiable weights. This gradient is usually used to find weights that minimize error. Back-propagation is a multilayer feed-forward network with one layer of z-hidden units. The output units has b(i) bias and z-hidden units has b(h) as bias. Both output and hidden units have bias. It acts like weights on connection from units whose output is always 1. The increasing no. of hidden layers results in computational complexity of network. As a result, time taken for convergence and to minimize the error may be very high.

Training algorithm of back-propagation involves 4 stages :

- Initialization of weights – some small random values are assigned.
- Feed-forward – each input unit(x) receives an input signal and transmits this signal to each of the hidden units Z_1, Z_2,……, Z_n. Each hidden unit then calculates the activation function and sends its signal Z, to each output unit. The output unit calculates the activation function to form the response of the given input pattern.
- Back propagation of errors – each output unit compares activation value with its target value to determine the associated error for that unit. Based on the error, the factor δ_k (k=1,……,m) is computed and is used to distribute the error at output unit Y_k back to all units in the previous layer. Similarly, the factor δ_j (j=1,…,p) is compared for each hidden unit Z_j.
- Updation of the weights and biases.



Initialization of weights:

Initialize weight to small random values, typically between -1 and 1. We continue to do the process till the stopping condition is false. The stopping condition may be minimization of the errors, number of epochs, etc.

Feed Forward:

When a specified training pattern is given to the input layer, the weighted sum of the input to the j^{th} node in the hidden layer is given by $\boldsymbol{Net_j = \sum w_{i,j}\, x_j + \Theta_j}$, where Θ_j term is the weighted value from a bias node. The bias node is considered a pseudo input to each neuron in the hidden layer and the output layer, and is used to overcome the problems related to the situations where the values of input pattern are zero. Hence, if any input pattern has zero values, the neural network cannot be trained without the bias node. To decide whether a neuron should fire, the 'Net' term also known as **Action Potential** is passed onto an appropriate activation function. The resulting value from the activation function determines the neuron's output, and becomes the input value for the neurons in the next layer connected to it. A typical activation function used is the Sigmoid function $\boldsymbol{O_j = x_k = 1/(1 + e^{-Net_j})}$**.** These equations are used to determine output value for node k in the output layer.

Back Propagation of errors:

If the actual activation value of the output node, k is O_k, and the expected target output for node k is t_k , the difference between the actual output and the expected output is given by :

$\Delta_k = t_k - O_k$

Error signal for node k in the output layer can be calculated as :

$\delta_k = \Delta_k O_k (1 - O_k)$

$\delta_k = (t_k - O_k) O_k (1 - O_k)$ where $O_k (1 - O_k)$ term is the derivative of the sigmoid function.

With the Delta Rule, the change in the weight connecting input node j and output node k is proportional to the error at node k multiplied by the activation of node j.

Updation of weights and biases

The formula used to modify the weight $w_{j,k}$, between the output node k, and the node, j is:

$\Delta w_{j,k} = l_r \delta_k x_k$ ---------(A)

$w_{j,k} = w_{j,k} + \Delta w_{j,k}$

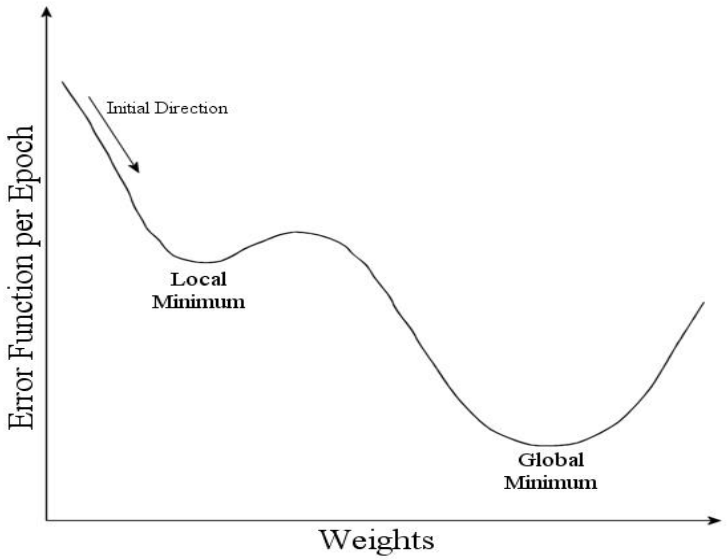
where $\Delta w_{j,k}$ is the change in the weight between nodes j and k , and l_r is the learning rate. The learning rate is relatively small constant that indicates the relatively change in the weights. If the learning rate is too low, the network will learn slowly. If the learning rate is too high, the network may oscillate around minimum or the lowest point. Some modifications of Back Propagation algorithm allows the learning rate to decrease from a large value during the learning process.

Since it is assumed that the network initiates at a state that is distant from the optimal set of weights and the training will initially be rapid. As the learning progresses, learning rate gradually decreases as it approaches the optimal point in the minima. Slowing the learning process near the optimal point encourages the network to converge to a solution while reducing the possibility of overshooting. If the learning process initiates close to the optimal point, system may initially oscillate but this effect is reduced with time as the learning rate decreases.

It is noted that in equation A , the x_k variable is input value to node k and is same value as the output from node j. In order to improve the process of updating weights, a modification of this equation is made :

$\Delta w_{j,k}^{n} = l_r \delta_k x_k + \Delta w_{j,k}^{(n-1)} \mu$

Here the weight update during the nth iteration is determined by including a momentum term μ , which is multiplied to the (n-1)th iteration of the $\Delta w_{j,k}$. Typically , the momentum term has a value between 0 and 1. The introduction of the momentum term is used to accelerate the learning process by 'encouraging' the weight changes continuing in the same direction with larger steps. Also, the momentum term prevents the learning process from settling in a local minima by 'over stepping' the small 'hill'.



Global and Local Minima of Error function

The error signal for node j in the hidden layer can be calculated as:

$\delta_k = (t_k - O_k)\, O_k \sum(w_{j,k}\, \delta_k)$

where the summation term adds the weighted error signal for all nodes k, in the output layer.

Similarly, the formula to adjust the weight $w_{i,j}$ between the input node i and the node j is :

$\Delta w_{i,j}^{n} = l_r\, \delta_j\, x_j + \Delta w_{i,j}^{(n-1)} \mu$

$w_{i,j} = w_{i,j} + \Delta w_{i,j}$

Finally , Back Propagation is derived by assuming that it is desirable to minimize the error n the output nodes over all the patterns presented to the neural network. The following equation is used to calculate the error function E, for all patterns :

$E = {}^{1}/_{2} \sum (\sum (t_k - O_k)^2)$

Ideally, the error function should have a value of zero when the neural network has been correctly trained. This, however, is numerically unrealistic.

Merits of Back Propagation

- Relatively simple implementation.
- Mathematical Formula used in algorithm can be applied to any network.
- Computing time is reduced if the weights chosen are small at the beginning.
- Batch update of weights exist, which provides a smoothing effect on the weight correction terms.

Demerits of Back Propagation

- Slow and inefficient. Can get stuck in local minima resulting in sub-optimal solutions.
- It is easy to create a number of examples of the correct behaviour.
- The solution to the problem may change over time, within the bounds of the given input and output parameters (i.e., today 2 + 2 = 4 , but in the future we may find that 2 + 2 = 3.8).
- Outputs can be 'fuzzy' or non – numeric.

Learning Rate **[16]:** The amount of change to the model during each step of the process, or the step size is called the learning rate and provides perhaps the most important hyperparameter to tune for our neural network in order to achieve good performance in our problem. It has a small positive value often in the range between 0.0 and 1.0.

When it comes to choosing a database, the biggest decisions is picking a relational (SQL) or non-relational (NoSQL) databases **[15].** SQL databases use structured query language (SQL) for defining and manipulating data. A NoSQL database, on the other hand, has dynamic schema for unstructured data, and data is stored in many ways: it can be column-oriented, document-oriented, graph-based or organized as a key-value store. In this database, we can create documents without defining their structure first & the syntax may vary from database to database. SQL databases are table-based while NoSQL databases are either document-based, key-value pairs, graph databases or wide-column stores. Some examples of SQL databases include MySQL, Oracle, Microsoft SQL Server and NoSQL database examples include MongoDB, BigTable, RavenDB.

In this project, I have used MongoDB as database. It is a good choice for businesses that have rapid growth or databases with no clear schema definitions. More specifically, if we cannot define a schema for our database, or if our schema continues to change – as is often the case with mobile apps, real-time analytics, content management systems, etc. – MongoDB can be a strong choice for us.

RELATED WORK

To get more efficient and effective result of K-mean algorithm there have been a lot of research happened in previous days. All researchers worked on different view and with different idea. Krishna and Murty **[4]** proposed the genetic K-means (GKA) algorithm which integrate a genetic algorithm with K-means in order to achieve a global search and fast convergence. Jain and Dubes **[1]** recommend running the algorithm several times with random initial partitions. The clustering results on these different runs provide some insights into the quality of the ultimate clusters. Likas et al.**[5]** proposed a global K-means algorithm consisting of series of K-means clustering procedures with the number of clusters varying from 1 to K. One disadvantage of the algorithm lies in the requirement for executing K-means N times for each value of K, which causes high computational burden for large data sets. The most common initialization was proposed by Pena, Lozano et al. **[6].** This method is selecting randomly K points as centroids from the data set. The main advantage of the method is simplicity and an opportunity to cover rather well the solution space by multiple initialization of the algorithm. Ball and Hall proposed the ISODATA algorithm **[7],** which is estimating K dynamically. For selection of a proper K, a sequence of clustering structures can be obtained by running K-means several times from the possible minimum Kmin to the maximum Kmax **[12].** These structures are then evaluated based on constructed indices and the expected clustering solution is determined by choosing the one with the best index **[8].** The popular approach for evaluating the number of clusters in K-means is the Cubic Clustering Criterion **[9]** used in SAS Enterprise Miner.

PROPOSED WORK

Project Description :

This is one of the applications for predicting the future rainfall and this function is very useful to the farmers to get a hold of the good result of specific weather. The farmers may face the problem about crops, i.e., the farmers does not know which crop would be appropriate for which weather condition, for that this application will be developed which in turn will be helpful for farmers to increase their cultivation.

The main objective of this project is to predict the future rainfall condition on the application of data mining techniques.

In this project work, it is being shown that the huge data is available in meteorological field to extract information from large datasets using data analytic tool. Here in this paper, a real dataset has been taken. In this project, it is being explained that clustering is the powerful tool which is being used in various forecasting tools. Here, the research has been done on rainfall data of various regions of India. The main approach of this paper is to first apply the k-means clustering algorithm on the original database. Then compute the means of each cluster based on rainfall data in the database. ANN is also being used to foretell the behavior of non-linear and complex systems. It is commonly being used by the researchers for the purpose of rainfall prediction. The fundamental processing element of ANN is an artificial neuron. Just like the natural neuron in human brain, it can receive inputs, process them and produce the relevant output. Neural networks are successful in modeling weather or rainfall forecast system. Accurate climate forecasting performs an integral role for planning day to day activities. Neural community has been used in meteorological purposes together with climate and rainfall forecasting.

Data Collection :

In this paper, the prediction of rainfall data has been collected or performed by using different data mining methods. Monsoon data set has been received from the URL https://data.gov.in/keywords/annual-rainfall. In this dataset, around 67 years rainfall data are available starting from 1951 to 2017. Usually downloaded data are always interlinked with some challenges like missing values, high dimensional values, noise, etc. which are not efficient for all classification. Therefore, clustering is the alternate solution for data analytics. The collected data from the site in excel format is as follows:

SUBDIVISION	YEAR	january	february	march	april	may	june	july	august	september	october	november	december
Andaman & Nicobar Islands	1951	82.7	7.2	0	45.4	259	619.9	665.3	101.3	360.9	489	209.6	434.8
Andaman & Nicobar Islands	1952	0	0.8	69.7	39.4	452.9	657.7	385.5	541.3	240.3	315.6	287.5	89.2
Andaman & Nicobar Islands	1953	56	65.3	20.1	159.4	241.1	549.9	444.4	262.8	370.1	243.6	246	63.3
Andaman & Nicobar Islands	1954	83.8	2.1	34.3	58	394.7	539.4	510.8	605.4	763.8	247.2	84.8	124.7
Andaman & Nicobar Islands	1955	57.3	28.2	9.9	68.8	663.6	651.1	298.4	356.9	341.8	466.5	362.9	44.2
Andaman & Nicobar Islands	1956	49.2	77.2	33.8	130.3	491.8	343.1	468.4	467.7	325.1	458.7	164.1	70.5
Andaman & Nicobar Islands	1957	28	19.3	4.6	19	142.4	741.9	316.2	421.3	354.3	299.7	88.6	71.9
Andaman & Nicobar Islands	1958	22.9	14.3	30.4	24.6	355.1	530.7	440.7	388.4	350.6	265.2	313.8	70.7
Andaman & Nicobar Islands	1959	10.1	1.8	31.3	86.7	238.1	455.7	817.1	409.3	589.6	332.7	242.5	128.5
Andaman & Nicobar Islands	1960	104.2	72.4	12.1	53.5	535.4	541.5	308.4	270.7	442.4	400	191.7	92.8

....

Andaman & Nicobar Islands	2016	72	15.8	5.4	2.4	191.1	429.4	301.2	227.7	604.3	287.2	181.7	533.7
Andaman & Nicobar Islands	2017	228.7	5.6	33	108.3	275.8	349.1	389.4	414.7	372.8	263	205.9	243.7
Arunachal Pradesh	1951	31.8	65.3	309.9	441	344.5	757.8	554.7	278.5	383.7	101.8	46.2	39.1
Arunachal Pradesh	1952	17.2	24.5	189.6	151.5	235.1	276.8	320.6	481.1	368.2	257.9	43.9	29.7
Arunachal Pradesh	1953	56.4	106.9	236	214.9	314.9	409.9	512.5	245.3	449.8	246.2	10.4	23.4
Arunachal Pradesh	1954	52.3	89.6	81.1	180.2	326.5	253.1	738.3	411.6	366.4	75.8	6.9	15.7
Arunachal Pradesh	1955	34.9	23.6	33.8	117.6	238	446	432.7	220.8	172.9	235.1	88.9	11.5
Arunachal Pradesh	1956	56.3	56.4	47.1	121.8	331.9	494.9	377.9	227.5	212.8	213.5	87.9	21.5

The subdivisions/regions are as follows:

Indian States/ Union Territories	**Subdivision/region name according to collection data**
JAMMU & KASHMIR	JAMMU & KASHMIR
HIMACHAL PRADESH	HIMACHAL PRADESH
PUNJAB	PUNJAB
HARYANA	HARYANA DELHI & CHANDIGARH
UTTARAKHAND	UTTARAKHAND
RAJASTHAN	WEST RAJASTHAN, EAST RAJASTHAN
GUJARAT	SAURASHTRA & KUTCH, GUJARAT REGION
MAHARASHTRA	KONKAN & GOA, MADHYA MAHARASHTRA, MATATHWADA, VIDARBHA
GOA	KONKAN & GOA
KARNATAKA	COASTAL KARNATAKA, NORTH INTERIOR KARNATAKA, SOUTH INTERIOR KARNATAKA
KERALA	KERALA
TAMILNADU	TAMIL NADU
ANDHRA PRADESH	RAYALSEEMA, COASTAL ANDHRA PRADESH
TELANGANA	TELANGANA
ORISSA	ORISSA
CHHATTISGARH	CHHATTISGARH
MADHYA PRADESH	WEST MADHYA PRADESH, EAST MADHYA PRADESH
UTTAR PRADESH	EAST UTTAR PRADESH, WEST UTTAR PRADESH
BIHAR	BIHAR
JHARKHAND	JHARKHAND
WEST BENGAL	SUB HIMALAYAN WEST BENGAL & SIKKIM, GANGETIC WEST BENGAL
SIKKIM	SUB HIMALAYAN WEST BENGAL & SIKKIM
ASSAM	ASSAM & MEGHALAYA
MEGHALAYA	ASSAM & MEGHALAYA
ARUNACHAL PRADESH	ARUNACHAL PRADESH
NAGALAND	NAGA MANI MIZO TRIPURA
MANIPUR	NAGA MANI MIZO TRIPURA
MIZORAM	NAGA MANI MIZO TRIPURA
TRIPURA	NAGA MANI MIZO TRIPURA

ANDAMAN & NICOBAR ISLANDS	ANDAMAN & NICOBAR ISLANDS
CHANDIGARH	HARYANA DELHI & CHANDIGARH
DADAR AND NAGAR HAVELI	GUJARAT REGION
DAMAN AND DIU	GUJARAT REGION
DELHI	HARYANA DELHI & CHANDIGARH
LAKSHADWEEP	LAKSHADWEEP
PUDUCHERRY	TAMIL NADU

Data Processing:

We directly do not store data, which is in excel/csv format. First, we have to know how to store these data into the database. So, I decided to store data in this format and for that reason we have to group data in format of {Region, Year, Month, Rainfall-Data} form of csv/excel format collection data. After data grouping, we get a huge amount of data = no of different subdivisions x 67 Years x 12 months = 28944.

Data insertion:

To insert data into mongodb we have to create a json file like:

```
{ “REGION” : “ORISSA”,
“YEAR” : 2009.0 ,
“MONTH” : “MAY”,
“DATA” : 68.2 }
```

After insertion we have to find out data into database. If that data already exist in database then we update that data otherwise we insert that data into the database.

Mongodb insertion documents:

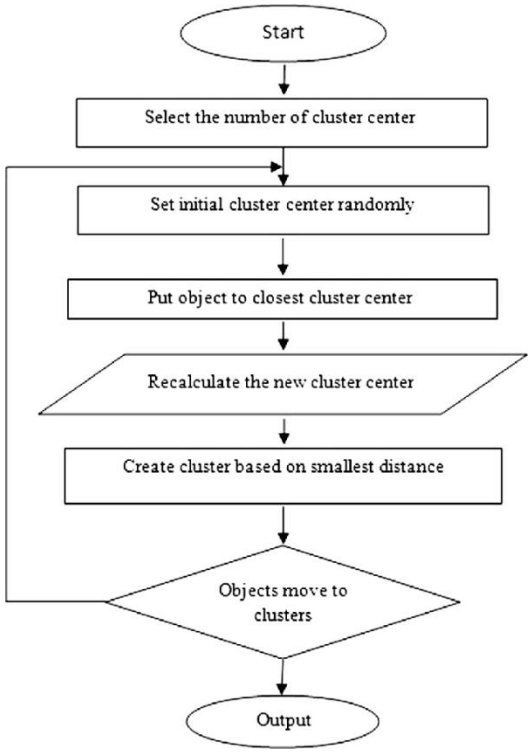
```
/* 9 */
{
    "_id" : "rainfall8",
    "REGION" : "ANDAMAN & NICOBAR ISLANDS",
    "YEAR" : 1951.0,
    "MONTH" : "SEPTEMBER",
    "DATA" : 360.9
}

/* 10 */
{
    "_id" : "rainfall9",
    "REGION" : "ANDAMAN & NICOBAR ISLANDS",
    "YEAR" : 1951.0,
    "MONTH" : "OCTOBER",
    "DATA" : 489.0
}

/* 11 */
{
    "_id" : "rainfall10",
    "REGION" : "ANDAMAN & NICOBAR ISLANDS",
    "YEAR" : 1951.0,
    "MONTH" : "NOVEMBER",
    "DATA" : 209.6
}
```

Methodology :

Clustering is a technique used to analyze data in efficient manner and generate required information. To cluster the dataset , k-mean is applied which is based on central point selection and calculation of Euclidean distance and on the basis of this distance, points are assigned to the clusters. The main weak point or the drawback of k-mean is of accuracy, as in k-mean we need to define no. of clusters, some points of dataset remain unclustered. The proposed model of K-means clustering is shown in the flowchart below:



In ANNs, each connection between 2 neurons (nodes) has a weight, an integer no. that controls signal between 2 neurons (nodes). If the network generates a 'good or desired' output, then there is no need to adjust the weights. However, if the network generates a 'poor or undesired' output or if any error occurs, then there is a requirement in the alteration of the weights in order to improve subsequent results. Proper tuning of weights ensures lower error rates, making the model reliable by increasing its generalization.

To understand Code Structure we need to focus on some topics**:**

utils : It is an utility Class, also known as Helper class, a class which contains just static functions, it is stateless and cannot be instantiated. It contains a collection of related functions, so they can be reused across the application.
StringUtils, IOUtils, FileUtils from Apache Commons; Iterables and Iterators from Guava, and Files from JDK7 are the perfect examples of utility classes.
In my project I have used some *utils* file within the **share/utils** directory.

(i) *utils.py*: used for the static functions which are being used anywhere in the project

(ii) *dbUtils.py*: used for the functions which are used within the database methods
(iii) filterUtils.py: used for the static functions which are used in filtering the data
(iv) *expectUtils.py*: these functions are used for expecting future data

config: configuration files (or config files) are files used to configure the parameters and initial settings of the application.
In my project I have used **config.json** file within the **share** directory. Here I have initialized:
(i) raw data path(RainFallData.xlsx)
(ii) database configuration(mongoDB)

Important Packages**:**

xlrd: It is a library for the developers to extract data from Excel spreadsheet files.
Xlrd is a Python module for extracting data from Excel spreadsheet files.
It can be used to extract data from new and old Excel spreadsheets on any platform.
The package itself is of pure Python with no dependencies on modules or packages outside the standard Python distribution. It has a strong support for Excel dates and is unicode-aware.

pymongo: PyMongo is a Python distribution containing tools for working with MongoDB, and is the recommended way to work with MongoDB from Python. This documentation attempts to explain everything we need to know to use PyMongo.

pip: pip is a package-management system used to install and manage software packages written in Python. pip is a recursive acronym for "Pip Installs Packages".

matplotlib: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.

numPy: For any scientific project, NumPy is the tool to know. It has been built to work with the N-dimensional array, linear algebra, random number, Fourier transform, etc. It can be integrated to
C/C++ and Fortran. NumPy is a programming language that deals with multi-dimensional arrays and matrices.

json: It is a built-in package which can be used to work with JSON data. It is a syntax for storing and exchanging data. It is a text, written with JavaScript object notation.

Mathematical Explanation :

Suppose there is a set of air pollution data which consist of 15 data. Suppose each data represents the data of each day. The set is shown by the table below **[10]** :

Air Pollutant Data			
CO_2	***RPM***	***SO_2***	***NO_x***
82	14	12	24
72	56	28	8
36	2	48	5
7	-	94	62

Here typical K-means is applied for initial data. Let us assume that initially the value of clusters is 4 i.e., K=4 & initially the means of those four clusters are C1=8, C2=56, C3=28, C4=72 and also assume that the above database contains no noisy data.

First Iteration:
We first apply typical K-means on the above data by using Manhattan distance metric (|Ai−Aj|).
Now for the first data 12(SO2),
C1=|8−12|= 4(minimum)
C2=|56−12|= 44
C3=|28−12|= 16
C4=|72−12|= 60
So, 12 ⟶ C1
Thus applying the same technique for the above all data, we have ->

		#items	***Means***
C1 = {12, 8, 5, 14, 7, 2}	=	6	8
C2 = {56, 48, 62}	=	3	55.33
C3 = {28, 24, 36}	=	3	29.33
C4 = {72, 82, 94}	=	3	82.66

Second Iteration:
Now, again we perform clustering based on the above new generated means,
C1=12 C2=48 C1=5
C2=62 C3=24 C1=8
C1=14 C1=7 C4=82
C4=72 C3=28 C1=2
C3=36 C4=94 C2=56

The result of the second iteration is same as the above.
From the above four clusters the nature of those clusters on climate change can be measured.

- From the resultant data of cluster 1 (C1), it can be said that the effect of **RPM** (14 is maximum) are more compare to the other pollutants. So, as per their effects on weather , the weather of those particular days were smogy in nature and also lots of dust, fly ash was there in the weather.
- As per the nature of Cluster 2 (C2), the weather of those particular days were hot, dry and smogy in nature due to the effect of **NOx**.
- As per the nature of Cluster 3 (C3), the weather of those particular days were hot, smogy and humid due to effect of **CO_2** ('Greenhouse effect').

- As per the nature of Cluster 4 (C4), the weather of those particular days were hot, smogy and also there may be a chance of acid rain due to the effect of **SO_2**.

Algorithms :

<u>storing data into the database -></u>

set csv/excel data path
assume 2 arrays A[], B[]
for each row i in excel data
    A[i] = ith row data , where A[0] represents the csv header
create data structure like [region,year,month,data] from array A depending on csv/excel header store it into array B
each B[i] represents a particular region, a particular year & a particular month rainfall data
establish database connectivity and store all B[i] in database using a specific database insert query

<u>clustering, filtering and updating data -></u>

*initialize **sumData** and count to 0*
*read **arrData** as the database data*
*calculate **maxData**, **minData** as maximum and minimum of all data*
*calculate **meanSe**t as set of means of all clusters which is equal to the function **findClusteringMeans**(utils.Utils.jsonData(["clustering", "noOfClusterSet"]),maxData,minData)*
*calculate **clusterSet** as set of clusters which is equal to the function **assembleClusterSet**()*
*calculate **diff** as the difference between maxData and minData divided by no. of clusters*
calculate mean dataset which is equal to (round(minData+diff(i+.5),2))*
*for i in range(0, **len(arrData**)*
    *each data is fetched & proper cluster chosen using function **chooseProperCluster()***
*calculate mean as sum of all datapoints divided by **rowlength***
if distance between a datapoint and meanpoint is less than error value
    that datapoint is placed in the cluster of that meanpoint

*for i in range(0,len(**clusterData**))*
    low and high data are being reset for filtering
    their difference to just higher and lower data added to corresponding divided by 1.5 and 2.0 respectively round off to 2

*update data in **updateData***

<u>expecting the future rainfall data</u> ->

*read **cQuery** as client Expecting Query*
*calculate **dbQuery** to fetch data from database*
*calculate database query which depends on **cQuery** where **cQuery** may/ may not be a array of database query.*

*calculate **dbData** as database data fetch according to **dbQuery***
each dbData[i] contains a region's particular year-month's rainfall data in format [region, year, month, data]

<u>for expecting future data, ANN algorithm is being used</u> ->

*calculate **trData** (trInput, trOutput) as the ready training data from **dbData** for ANN*
*calculate **exIData** as ready expecting input data to see both **dbData** and **cQuery***
*calculate **exOData** as expecting output data which is equal to the function **ANNAlgorithm** (trInput, trOutput,exIData)*
*if (**exOData** match according to cQuery)*
print(exOData)
*include **exOData** with **dbData** of format [region, year, month, data]*
*again calculate **trData** for the training data from **dbData***

<u>ANN Algorithm -></u>

*read X as **trInput,** training input*
*read Y as **trOutput,** training output*
*read T as **exIData,** testing input*
set hh as height of each hidden layer
set ly as the no. of hidden layers
set l_r as learning rate in the interval [0.0, 1.0]
assume W as [w_1,w_2,w_3..,w_n]; set of weights of each hidden layer
assume B as [b_1,b_2,b_3...,b_n]; set of bias of each hidden layer
where, w_i is the set of random numbers between [-1, 1] of order (length(X), length(hh))
and b_i is the set of random numbers between [-1, 1] of order (1, length(hh))

for training data
for i=0, l_r :
*calculate Z = **forward**(X,W,B), which is an array*
*calculate (W,B) = **backward**(X,Y,Z)*

for testing data
*calculate result= **testing**(T,W,B)*
return the value of result

*<u>testing(T,W,B) function **->**</u>*

z_1 = sigmoid((T.w_1)+b_1)
z_2 = sigmoid((h_1.w_2)+b_2)
z_3 = sigmoid((h_2.w_3)+b_3)
. . . . .
z_n = sigmoid((h_n-1.w_n)+b_n)
return [z_1,z_2,….,z_n]

<u>forward(X, W_i, B_i) function -></u>

$z_1 = sigmoid((X.w_1)+b_1)$
$z_2 = sigmoid((z_1.w_2)+b_2)$
$z_3 = sigmoid((z_2.w_3)+b_3)$
. . . . .
$z_n = sigmoid((z_n\text{-}1.w_n)+b_n)$
return $[z_1,z_2,z_3...z_n]$

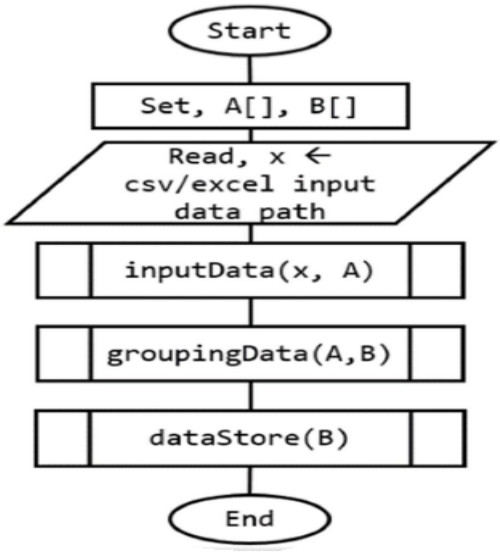
backward(X,Y, Z) function ->

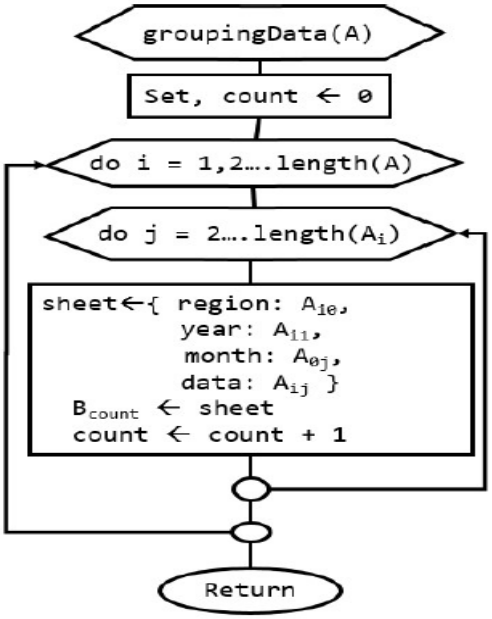
$a_1 = derv\_sigmoid((z_n * (Y-z_n))$
$a_2 = derv\_sigmoid((z_n\text{-}1 * (a_1.W_n^T))$
$a_3 = derv\_sigmoid((z_n\text{-}2 * (a_2.(W_n\text{-}1)^T))$
. . . . . .
$a_n = derv\_sigmoid((z_1 * ((a_n\text{-}1).W_1^T))$

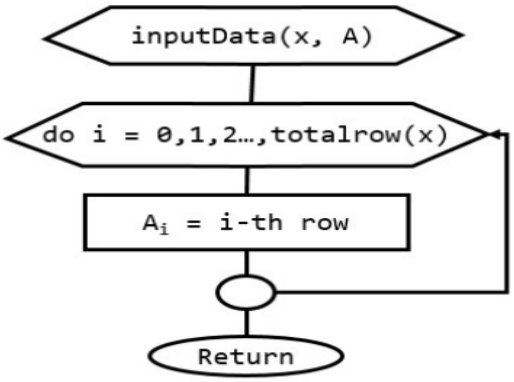
$w_1 += l_r * (X^T.a_n)$, $b_1 += sum(a_n)$
$w_2 += l_r * (z_1^T.a_n\text{-}1)$, $b_2 += sum(a_n\text{-}1)$
$w_3 += l_r * (z_2^T.a_n\text{-}2)$, $b_3 += sum(a_n\text{-}2)$
. . . .
$w_n += l_r * ((z_n\text{-}1)^T.a_1)$, $b_n += sum(a_1)$
return $([w_1,w_2,w_3...w_n], [b_1, b_2 ,b_3...b_n])$

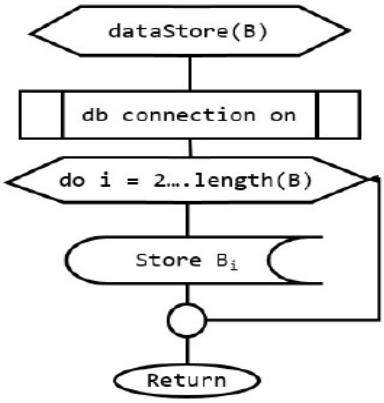
Flowcharts:

For storing data into the database->

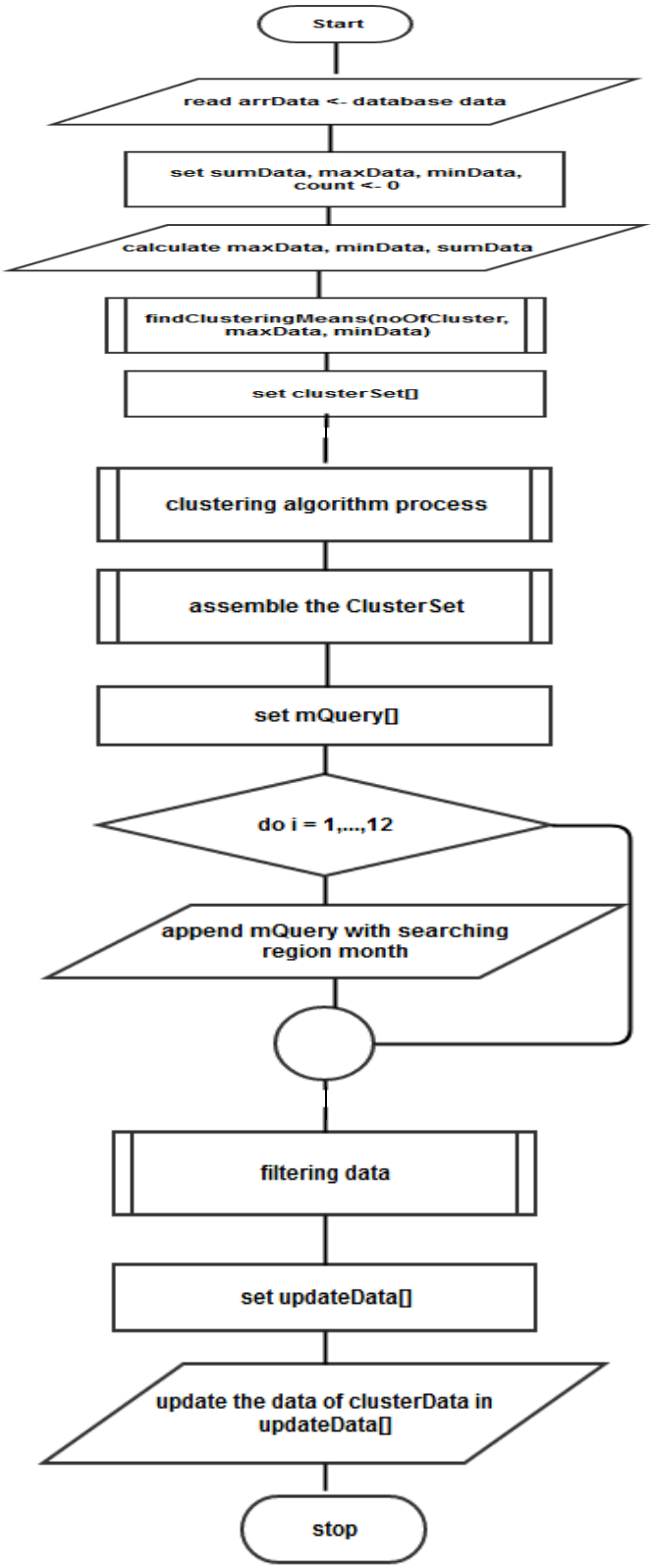




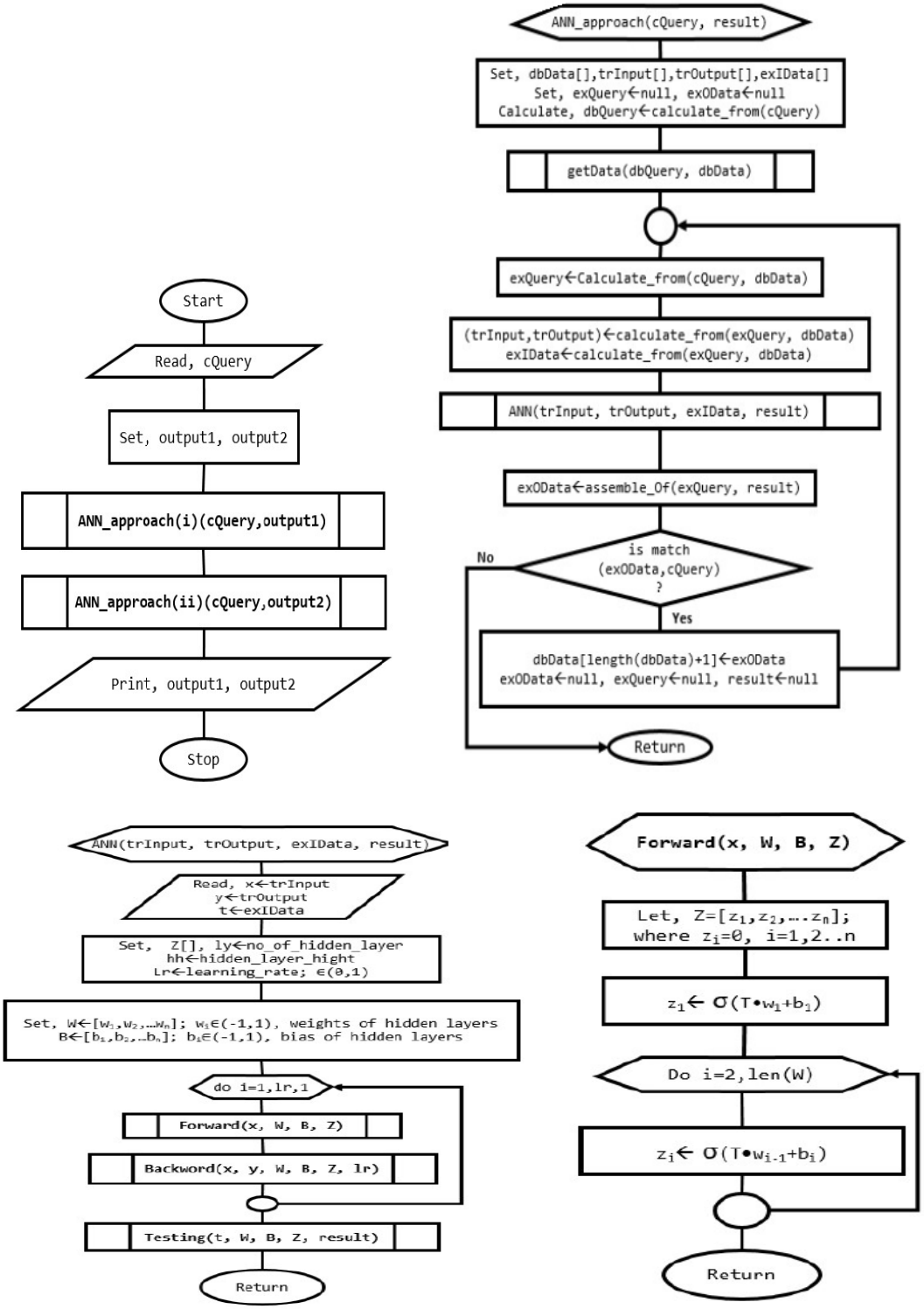


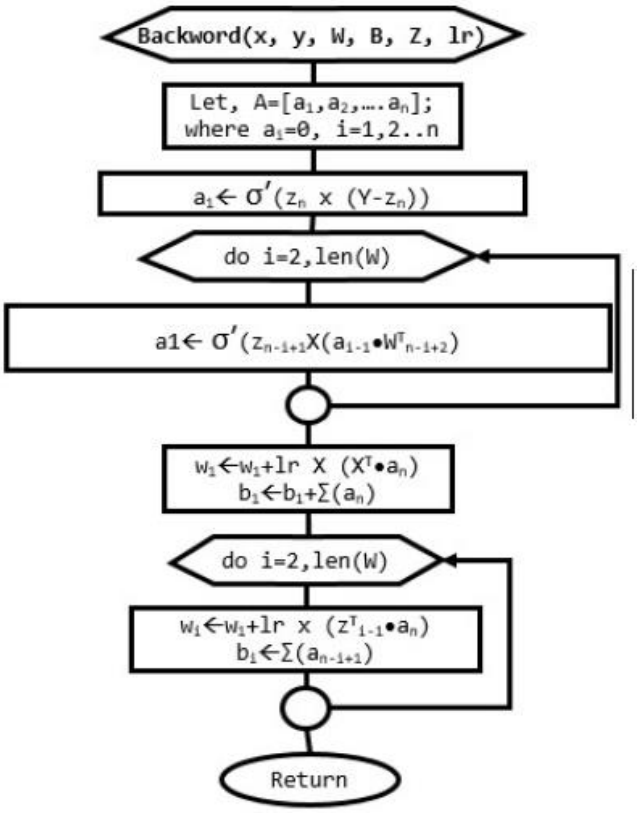


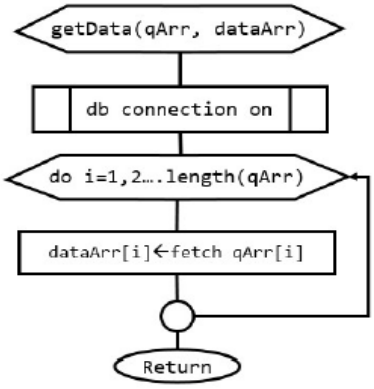
Clustering, filtering and updating data ->

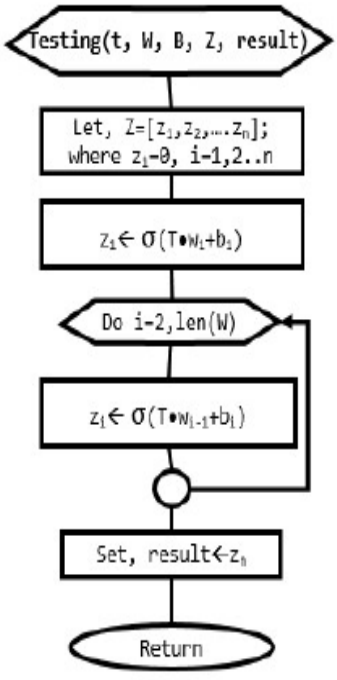


Expecting future rainfall data ->




Backword(x, y, W, B, Z, lr)
Let, A=[a1,a2,….an];
where a1=0, i=1,2..n
a1← σ′(zn x (Y-zn))
do i=2,len(W)
a1← σ′(zn-i+1X(a1-1•WTn-i+2)
w1←w1+lr X (XT•an)
b1←b1+∑(an)
do i=2,len(W)
w1←w1+lr x (zT1-1•an)
b1←∑(an-i+1)
Return


getData(qArr, dataArr)
db connection on
do i=1,2….length(qArr)
dataArr[i]←fetch qArr[i]
Return


Testing(t, W, B, Z, result)
Let, Z=[z1,z2,….zn];
where z1=0, i-1,2..n
Z1← σ(T•W1+b1)
Do i-2,len(W)
zl← σ(T•Wl-1+bl)
Set, result←zn
Return

Code Structure:

```
RainFallDataAnalysis
|
|   main.py
|---src
|   |---controller
|   |       controller.py
|   |---excel
|   |       xlsxInput.py
|   |---grouping
|   |       groupingData.py
|   |---mongoo
|   |       mongoDBController.py
|   |-----clustering
|   |       kmean.py
|   |       filtering.py
|   |---expect
|   |       clientQuery.py
|   |       expectingQuery.py
|   |---ANNalgo
|   |       ann.py
|   |       annController1.py
|   |       annController2.py
|   |       processingANNdata1.py
|   |       processingANNdata2.py
|   |-----graph
|   |       dataProcess.py
|   |       plotGraphController.py
|   |---share
|       |   config.json
|       |   indianStates.json
|       |   months.json
|       |---utils
|           |   utils.py
|           |   dbUtils.py
|           |   expectUtils.py
|           |   filterUtils.py
|---data
    RainFallData.xlsx
```

Code Details :

main.py	It is being run as the entry point into the program.
controller.py	It acts as the key of the whole program. It is a basic class-based demonstration application. Under class **Controller**, constructors, destructors and functions are listed as follows: **__init__(self,dbConfig,dbName) :** Constructors of controller are basically used for the connection of database(mongoDB) **dbConfig** is used from the config.json **__del__(self) :** Destructors of controller are basically used for closing the connection of database(mongoDB) **filteringData(self) :** Mongo collection is being requested using MongoRequest function, get distinct data of region from the collection. Then, filtering process takes place and data stored in *mData*. **postData(self,dataPath):** This function is used to extract data from excel file from *xlsxInput.py* and grouping the data in a specific format {Region, Year, Month, Data} from *groupingData.py* And then update the data in the database using *mongodbController.py* **deleteData(self):** This function is used to clear the database through *mongodbController.py* **analysisData(self):** This function is used to expect future data in the following way: (i) get client query and generate dbquery according to the client query from *expectingQuery.py* (ii) get data from database corresponding to the db query (iii) send db data to *annController* for the expectation of searching data (iv) plot expect data (including previous data or not) through *drawGraph function* **drawGraph(self,pData,queryType,status) :**

	This function calls *plotGraphController.py* for plotting the data.
utils.py	It refers to the Utility class or known as helper class of the project which contains all Static functions. Under class **Utils**, the functions used are: **jsonData(searchItems):** This function is used to search ".json" format file data **menuDriven(choice):** This function is used to select a choice of posting, deleting or analyzing the data **int_or_float_str(s):** This function is used to convert string data to int or float type. In python, by default data is taken in string format. **month_number_to_string(num):** This function is used to convert month no. to its corresponding name **month_string_to_number(name):** This function is used to convert name of the moth to its corresponding no. **findEndYearEndMonth (arrData):** This function is used to find the endyear and the endmonth
config.json	It is the configuration file for initial settings of application. Here, mainly raw data path and mongodb connection are initialized and also files of searching months & states, ann details and no. of clusterset are being stated as follows: "**rawDataPath**" : {"rainfall" : "data/RainFallData.xlsx" } "**mongoConfig**" : { "url" : "localhost:27017", "dbName1" : "JUProject2019", "dbName" :"WeatherDataAnalysis", "collectionName" : {"rainfall": "RainFallData" } } "**expectData**" : { "clientQuery" : {"searchingStates" : "src/share/indianStates.json", "searchingMonths" : "src/share/months.json" }} "**ann**" : {"annLength" : 7, "hiddenSize": 5, "learningRate": 0.2, "trainingIteration" : 999} "**clustering**" : { "noOfClusterSet" : 7}

dbConnection.py	It is being used for the connection of database (here mongoDB is used) Under class **MongoConnection**, following functions are used**:** **__init__(self,dbConfig,collectionOriginalName) :** The constructor used to fetch mongourl, dbName and dbCollection name **__del__(self):** The destruction is used for closing the connection **start(self) :** It is used to start the db connection by giving url and collection name. **end(self) :** It is used to close db connection
xlsxInput.py	It is used to fetch the data from a ms-excel format using python package **xlrd.** Under class **ExcelInput**, following functions are used: **__init__(self,path) :** The constructor used to initialize data path, excel sheet and processed data is returned to the constructor using *processingData* function in *self.data* **excelInput(self) :** It takes data from excel file using xlrd package and store it into the sheet **processingData(self) :** It process n rows of sheet data and save in python list-type data within *self.data* in the constructor
groupingData.py	Processing excel return data give it a specific format to store into database. Class **DataSheet** is a Meta Class in which only one constructor is used: **__init__(self,region,year,month,data) :** It is used to create a sheet like RainFallData in {region,year,month,data) format In class **GroupingData,** **__init__(self, xlsxData):** Constructor used to initialize dataList and update dataGroup by using *processDataForGrouping* function **processDataForGrouping(self):** Using metaclass **DataSheet**, this function converts excel data in a specific format

mongodbController.py	It is used to store/fetch/update/delete data in the database. Under class **MongoRequest**, following functions are listed: **__init__(self,collection) :** Constructor used to set mongo collection **__del__(self):** Destructor used to delete the mongo collection **postData(self, data) :** It is used to store data into the database **deleteData(self, query) :** It is used to delete/clear the database **getData(self, query) :** It is used to fetch data from the database **getDistinctData(self, mongoQuery) :** It is used to get distinct data of mongo collection using *mongoQuery* **convertJson2Array(self):** It is used to convert the json format file to an array format
expectingQuery.py	It is used to get the expecting query and generate dbQuery according to it Under class **ExpectingQuery**, following constructors, functions are used: **__init__(self):** Here, *cQuery* is initialized and we get the client query from *clientQuery.py* *self.mQuery0* is initialized and if(expectQueryYr<2017) then generate the query for displaying original data which lies on database *self.mQuery1* is db query of ANN-1 approach *self.mQuery2* = db query of ANN-2 approach Then, send *cQuery,mQuery0,mQuery1,mQuery2* to *controller.py* **__del__(self):** Destructor used to delete all client and mongo queries. **generateMongoQuery_OriginalData(self):** cQuery contains 3 attributes region, year and month. Now if the year is less than 2018, then the original data is being returned from *createMongoQuery_ForOriginalData* function under *expectUtils* file and if the given year is more than 2018 then the original data of last month of 2017 is being returned **generateMongoQuery_Approach1(self):** Based on *queryType*, mongo query are generated using *generateQuery1* function. When query is of type region and year,

	then it is being stored in *arr* array for all the months. **generateQuery1(self, arr):** In this, query is being generated according to ann approach 1 and is stored in the array. **generateMongoQuery_Approach2(self,option):** Based on previous 5 months of the given year and month, expecting data of that year can be calculated **generateQuery2(self, arr):** In this, query is being generated according to ann approach 2 and is stored in the array
annController1.py, annController2.py	It is used to ready db data for ann algorithm and store expecting result from ann algorithm
ann.py	It is used to describe the algorithm of artificial neural network using forward and backward propagation. Under class **Neural**, the following constructors & functions are used: **__init__(self,x,y,hidden_size,rate) :** Constructor used to initialize sizes of input, hidden and output layers, also the bias and weights of the hidden layers and also we take 2 arrays to read ANN input and output. **__del__(self) :** Destructor used to destroy the values of sizes of different layers. **sigmoid(self,xx):** It is used to calculate the sigmoid function. **derv_sigmoid(self,tt):** It is used to calculate the derivative of the sigmoid function. **forward(self,x):** It is used to calculate the hidden layers using sigmoid function of the summation of bias and the dot product of input array with weight. Value of sigmoid is then used for calculation of next hidden layer. **backward(self,x,y):** It is used for the backward propagation of the network. It uses derivative of sigmoid function of the value found in the last hidden layer during feed forward process multiplied by the difference of output layer and the last hidden layer and so on. **learn(self,iteration):** It is used to calculate learning rate i.e., specifying no. of iterations for which the forward and backward processes take place.

	test(self,test_x): It is used for testing process to test a given array by using feed forward
clientQuery.py	It is mainly used for searching proper state name from the json file according to the query given by the client. Under class **ClientQuery**, following are used: **__init__(self) :** Constructor used to initialize clientQuery, *stateJFile* and *monthJFile* taken from jsonData written in Utils directory **__del__(self) :** Destructor used to delete *stateJFile* and *monthJFile* being initialized **clientQuery(self) :** It is used to take query according to client by *<searchByRegionYear>* or *<searchByRegionYearMonth>* and states & months are loaded by *stateJFile* and *monthJFile* **searchYear(self,sms) :** Used to search a specific year which is greater than 1952 **searchJSONData(self, statesData, hintsSMS, sms1, sms2=None) :** Used to search various data according to hints given for the regions or months. Regions are searched by: "Searching with first 3/4 letters of a State/Union Territory" or "Searching with short name of each state" Months are searched by: "Searching with first three letters/ full Name" or "Searching with Numeric Number of Months" **searchingFromJSON(self, allData,stateName,stateArea=None) :** Used to get required *statename* and *statearea* given by the client from the jsonfile
expectUtils.py	**createMongoQuery_ForOriginalData(region,year,month=None) :** Creating mongo query json object for DB actual data for ANN-0 approach **createMongoQueryANN1(region,year,month) :** Static method used to create mongo query of region, year, month for the ann 1 approach **createMongoQueryANN2(region,year,months=None) :** Static method used to create mongo query of region, year, month for the ann 2 approach **queryType (query) :** It checks and return the type of the query given, i.e., <class 'str'><class 'int'> or <class 'str'><class 'int'><class 'str'>

dbUtils.py	It is used for the functions which are being used within the database methods. Under class **MongoDBUtils**, following functions are used: **postQuery(id,reg,yr,mon,data) :** It finds, set and inserts query of region, year, month, data alongwith an id given to each set **deleteQuery():** It deletes all data from the collection in mongodb
indianStates.json	Here, names of all regions are being listed alongwith their ids (possible hints) in the json format
months.json	Here, months are being listed in json format alongwith the ids (possible hints)
dataProcess.py	It is used for processing the data for plotting the graph. Under the class **dataProcessing**, following functions are used: **processDataForPlotting(arrayData) :** Static method used to plot a graph for the specific region-year-month's data, an array *arrData* is initialized and then values of region, year, month and data are appended on it **checkColor(i) :** It is also a static method used for stating a colour along with its id and status i.e., '0' is for blue for showing ann 1 result, '1' is for red for showing ann 2 result and '2' is for green for showing the original data in database
plotGraphController.py	Under class **PlotGraph**, following constructors and functions are used: **__init__(self)** constructor and **__del__(self)** destructor are used for initializing sms or searching type for plotting data **DrawGraph(self,xlabel_sms,ylabel_sms,title_sms) :** Used to plot the graph with x_label, y_label and title according to *xlabel_sms* , *ylabel_sms* and *title_sms* respectively **plotGraph(self, region, month, year, data, rsms, ysms, msms, dsms) :** According to the region and month given , graph is plotted for the range of length of data corresponding to each year Now under class **PlotController**, we have: **plotExpectData(self,arrayData, queryType,status=None):** Here, plotting of the expecting data is done according to the *queryType* given and the data to be processed is stored in the *arrayData* array
processingANNdata1.py	It is used to process ANN data using approach 1. Under class **ProcessOnlyData_of_ANN1**, following functions are used: **__init__(self, cQuery, allData, lenANN):** Constructor used to initialize ANNlength, ann input, ann output and maxData & *readyIOdata* function is called to ready ann data

	__del__(self) : Destructor used to delete *lenANN*, *annInput*, *annOutput*, *queryInput* and *maxData* **fetchingOnlyDataPart(self,allData):** Used to fetch only the data part of the array *allData* and store it *arr* **createANNInput(self) :** Used to create the input of ann. Data from *onlyData* array is being appended to an array. If length of onlyData array by subtracting no. of count is less than length of ANN, then return array *arr.* **createANNOutput(self) :** Used to create output array for displaying output of ANN for the range from length of ANN to length of *onlyData* array and return the resultant array **getMaxData(self) :** It returns max value of *onlyData* array **readyIOdata(self) :** Used to ready input output data. Here, *queryInput* array is taken. Last term of *annInput* is *queryInput* and the remaining first (n-1) data are *annInput*
processingANNdata2.py	It is used for processing ANN data using approach 2. Under class **ProcessOnlyData_of_ANN2**, following functions are used: **__init__(self, cQuery, arrData, lenANN) :** Constructor used to initialize *lenANN*, *maxData* & *annInput*, *annOutput*, *annQInput* arrays. **__del__(self) :** Destructor used to destroy or delete *lenANN*, *maxData* & *annInput*, *annOutput*, *annQInput* arrays. **createANNIO(self,cQuery,arrData) :** Firstly, first_end details are taken i.e., 1st year,1st month & last year, last month. *cQuery* is loaded with *qyear*, *qmonth* at positions 1 & 2 respectively. *ckLast* & *ckFirst* are used to take data from year 2017 to 1951 & for detecting which one is *annQueryInput.* Process of fetching data and expecting takes place for the range 0 to ANNlength. The data of a month to be expected is given by l2 i.e., column no. and for that previous 3 months' data are fetched and stored in array *filterIData*, max value is found out and then it is reversed. If value of *ckFirst* is 0, then *annQInput* is appended with *filterIData* otherwise *annInput* is appended with *filterIData* and *annOutput* is appended with *filterOData* which is the *arrData* with row k-1 and column of

	month to be expected. **maxValue(self, b) :** It is used to return the max value of maxData. If max(b) is greater than maxData, then return max(b) else return maxData. **find_First_End_Details(self,arrData) :** It is used to find first and last year and month from the *arrData* and return their values. **display(self) :** It displays *annQInput*, *maxData* and each *annInput[i]* and *annOutput[i]* for the range 0 to length of *annInput*.
kmean.py	It is used for clustering the data. Under class **Clustering_kMean**, *arrData*, *clusterSet* and *meanSet* are initialized. following functions are used: **__init__(self, arrData) :** Here in this constructor, *maxData* and *minData* arrays are taken, *sumData* and *count* are initialized to zero. For the range 0 to length of the *arrData*; *maxData* and *minData* are calculated according to the data part of the *arrData*. Then sum of all the data is calculated and then average data *avgData* is found out for all data. Next *meanSet* is created using *findClusteringMeans* function. By calling *clusteringAlgo* function, *clusterSet* is created and is assembled using *assembleClusterSet* function. **assembleClusterSet(self) :** It is used to sort or assemble clustering mean in the ascending order. For the range of the length of *clusterSet*, sorting of means of clusters is being done in ascending order and then the resultant *clusterSet* is returned. **findClusteringMeans(self, noOfCluster, maxData, minData):** Here, means of each cluster is calculated. Now, diff is calculated as the difference of *maxData* and *minData* divided by total no. of clusters. Mean dataset is stored in array *arr* which is calculated as *minData*+*diff**(i+.5) rounded off to 2 points. **clusteringAlgo(self) :** Clustering process is performed. Mean is stored at 0^{th} position of *clusterSet*. Now for range of the length of *arrData*, each data is fetched and a proper cluster is chosen using *chooseProperCluster* function. *meanArr* is calculated using *findMean* function. Also *count* counts the no. of iterations. **compareMeanAfterExecution(self, meanArr):** Used to find a point to be stored in which cluster. If the absolute

	value of difference of *meanArr* data and *clusterSet* data is greater than the value of error, then mean is set in 0^{th} position of *clusterSet*. **findMean(self):** Used to find the mean of each cluster. Here, *rowlength* is taken as length of *clusterSet*. For the range 0 to *rowlength*, sum of all datapoints of a cluster is calculated and then divided by *rowlength* round off to 2 gives the value of *meanArr.* **chooseProperCluster(self, value):** It is used to choose proper cluster. For the range of length of *clusterSet*, absolute difference between a data point value with the *clusterSet* point is calculated and stored in the array *diff*. *position* is returned which is the index of min value of *diff*.
filtering.py	Used for filtering the data after clustering. Under the class filtering, following are used: **__init__(self, allRegions):** Constructor used to initialize *mQuery* which searches mongo query of specific region & month using *selectedRegionMonthQuery function. allRegion* is initialized for fetching all the regions. **selectedRegionMonthQuery(self) :** For a specific region, *mQuery* is appended with the specific month using *query_SearchRegionMonth* function and return *mQuery* array. **query_SearchRegionMonth(self, reg, mon) :** Query generated for region & month with ids as 'REGION' and 'MONTH'. **filteringData(self, arrData) :** Firstly, the header of *arrData* is deleted and an object of the class **Clustering_kMean** is created. Object ob calls the array *clusterSet* and stores value in *clusterData.* Now for the range of length of *clusterData*, data of low and high *clusterData* is being reset and difference to the just higher and lower data of clusterData is added to the corresponding clusterSet divided by 1.5 and 2.0 respectively round off to 2. Now update the data of *clusterData* in *updateData* and deleting the 0^{th} row i.e., header of *clusterData*.
filterUtils.py	In this, methods are used to search all regions. **searchAllRegion():** All regions are searched and returned given by client or user. **queryForRegionMonth(self,reg,mon) :** Query generated for region and month.

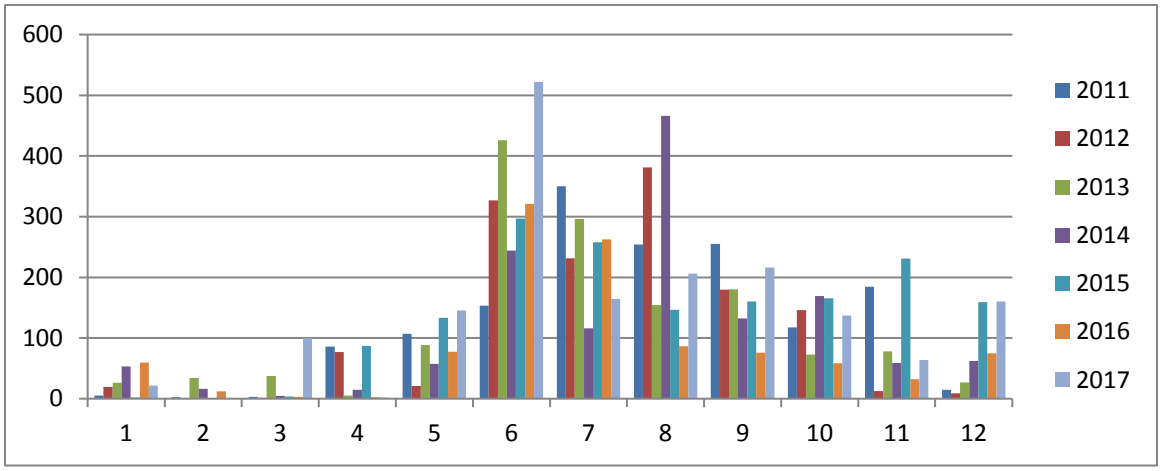
EXPERIMENTS AND RESULTS

We have evaluated our k-means algorithm using randomly generated dataset and compared with various proposed methods. We have used Python (version 3.7) language to implement our algorithm. Experiments were conducted on a machine consisting of operating system windows 7 on Intel i3-2350M CPU , 4 GB RAM, RDBMS/Back end used is MongoDB 4.0. Related packages of python used are: xlrd, pymongo, numpy, matplotlib.

The following table illustrates the comparison with other methods:

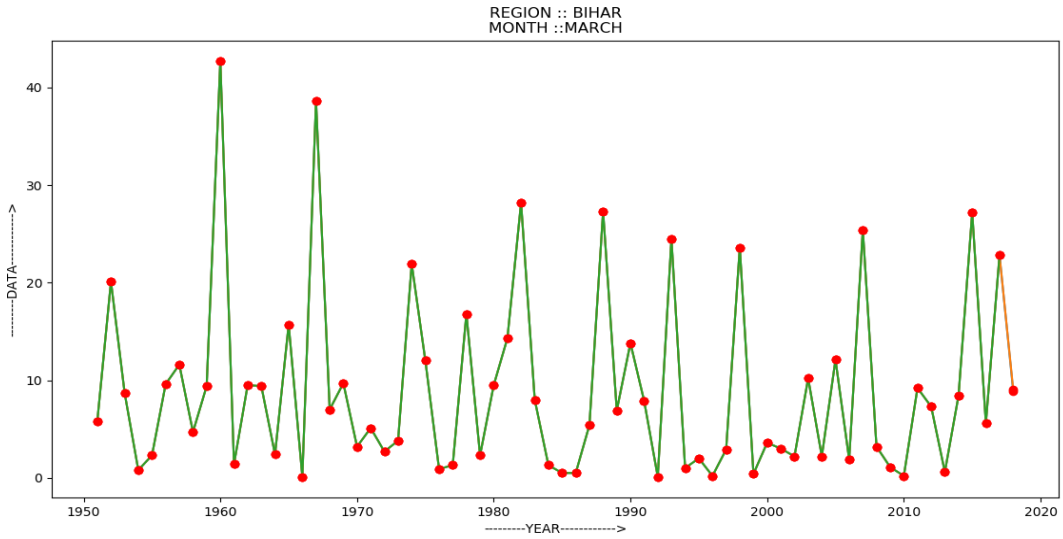
Algorithm	**Data points**	**Number of cluster**	**Initial cluster**
Standard K-means algorithm	Input by user	Input by user	Input by user or random
Improved K-means algorithm	Input by user	Input by user	Calculate by algorithm
Optimized K-means algorithm	Input by user	Input by user	Input by user or random

Result for the rainfall in Lakshadweep for the 7 years interval 2011-2017:



Here, numbers 1 – 12 represents the 12 months of a year.

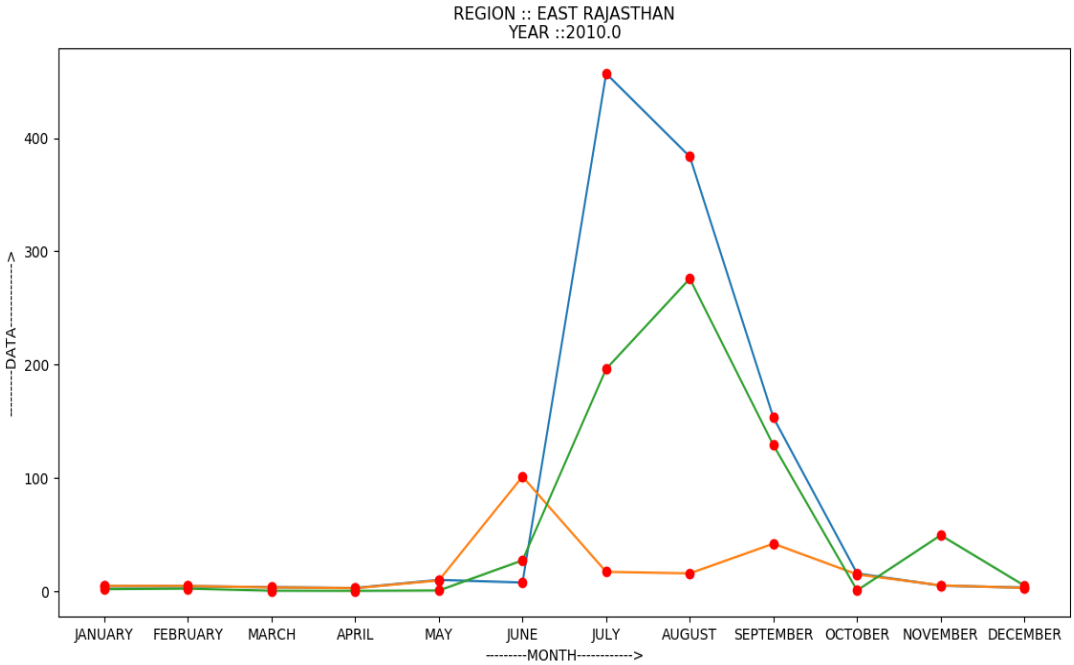
Result for the ‘expectation by region, year & month’ of Bihar March 2018 is as follows:



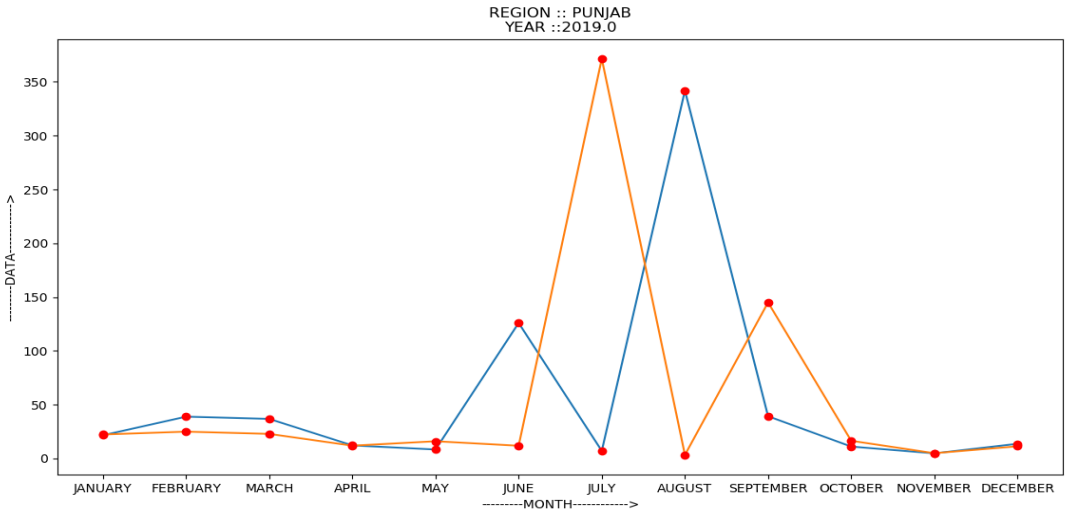
where Red line represents data predicted using ANN approach 2, Green line represents original data of the database.

Result for the 'expectation by region & year' of East Rajasthan 2010 is as follows:

Here, Blue line represents data predicted using ANN approach 1, Red line represents data predicted using ANN approach 2 and Green line represents the original data of the database.

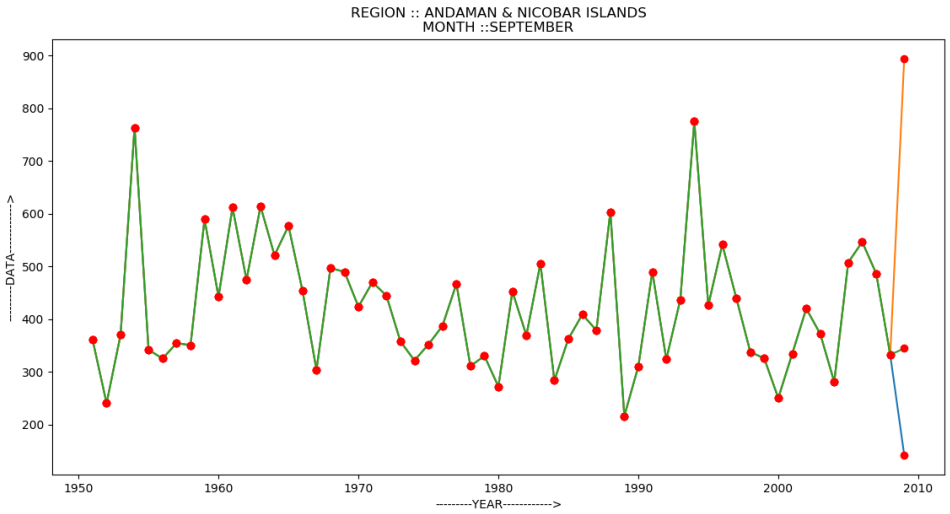


Result for the 'expectation by region & year' of Punjab 2019 is as follows:



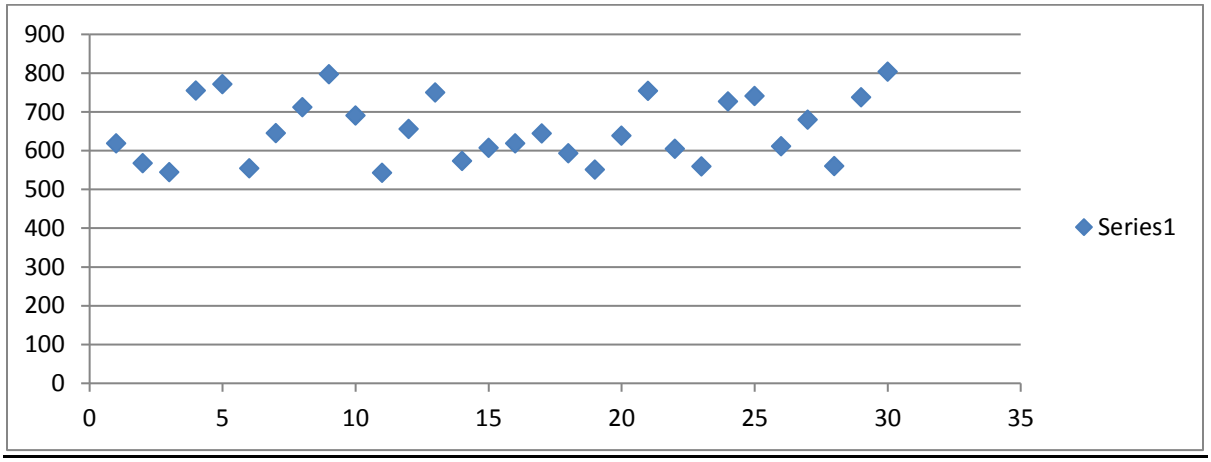
where Blue line represents data predicted using ANN approach 1 and Red line represents data predicted using ANN approach 2.

Result for the 'expectation by region, year & month' of Andaman & Nicobar Islands, September, 2009 is as follows:

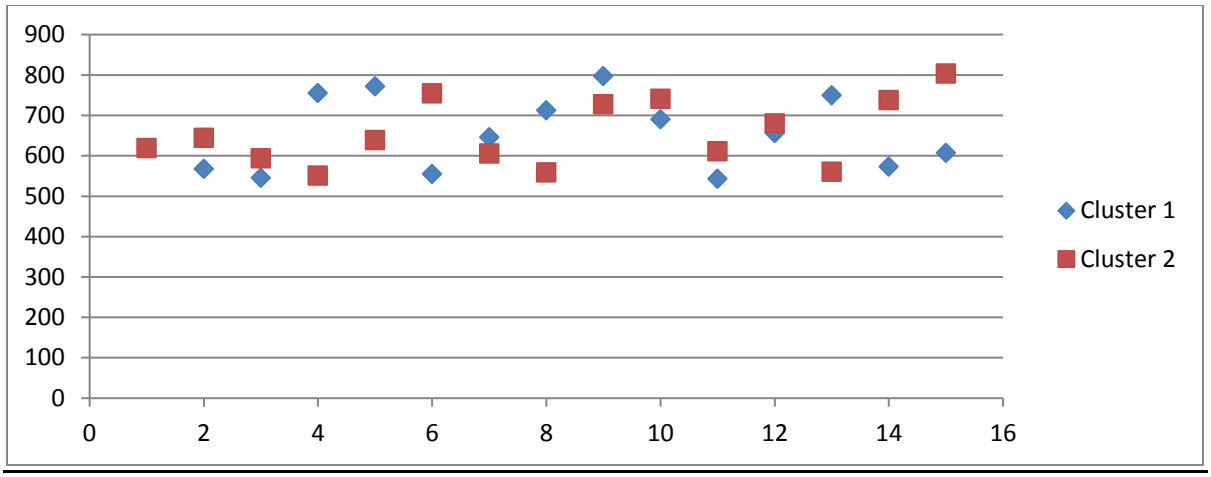


where Green line represents original data of the database, Blue line represents data predicted using ANN approach 1 and Red line represents data predicted using ANN approach 2.

Scatter plot of 30 data points (unclustered data) is shown below:



After clustering, scattered points are grouped into clusters as follows:


900
800
700
600
500
400
300
200
100
0
0
2
4
6
8
10
12
14
16
Cluster 1
Cluster 2

CONCLUSION & FUTURE SCOPE

In the proposed system, the study of general data mining algorithms is being presented for a rainfall forecast. This lookup or the work done will help the farmers to recognize how plenty production will be there in the forthcoming seasons and how a great deal quantity of rainfall will appear so that farmers can get awareness and they can control themselves from their heavy loss. The overall goal of data mining process is to extract information from a large data set and transfer it into an understandable form for future use. Clustering is important in data analysis and data mining applications. Clustering is a division of data into group of similar objects. Clustering can be done by different algorithms such as hierarchical-based, partitioning-based, grid-based or density-based algorithms. In this paper, techniques used to predict the rainfall of forthcoming years with the help of K-means clustering and ANN algorithms. Data Mining deploys techniques that are based on the machine learning. More importantly, these techniques were used generate the decision or prediction models. In future, other clustering algorithms can be used to predict the rainfall and weather and can compare them with each other to detect which algorithm among them provide better accuracy. In this paper, it is concluded that clustering is a technique by which large datasets are divide into small data collections that are called clusters. The cluster is a collection of the data that are turned into information. The data clusters are different from each others as they are possessing some different values from each other. There are a number of algorithms that work well for clustering the data that can divide a dataset into clusters.

REFERENCES

[1] Anil K. Jain and Richard C. Dubes, Michigan State University; *Algorithms for Clustering Data:* Prentice Hall, Englewood Cliffs, New Jersey 07632. ISBN: 0-13-0222278-X

[2] N. Sundaravalli , Dr.A.Geetha (2016) International Research Journal of Engineering and Technology(IRJET) ; *A Study & Survey on Rainfall Prediction And Production of Crops Using Data Mining Techniques 12(3),* ISSN: 2395 -0056 https://www.irjet.net/archives/V3/i12/IRJET-V3I12284.pdf

[3] Aswini.R, Kamali.D, Jayalakshmi.S, R.Rajesh (2018); IFET College of Engineering and Technology, Villupuram, India*; Predicting Rainfall And Forecast Weather Sensitivity Using Data Mining Techniques* 14(119)*;* pp 843-847 https://acadpubl.eu/hub/2018-119-14/articles/2/104.pdf

[4] Krishna K, Murty M (1999) *Generic K-Means algorithm.* IEEE Transactions on systems, man, and cybernetics- part B: Cybernetics, 29(3): pp 433-439

[5] Likas A, Vlassis N, Verbeek J (2003) *The global K-means clustering algorithm*. Pattern recognition, 36(2), pp 451-461

[6] Pena JM, Lozano JA, Larranaga P (1999) *An empirical comparison of four initialization methods for K-means algorithm.* Pattern recognition letters 20: pp 1027-1040

[7] Ball G, Hall D (1967) *A clustering technique for summarizing multivariate data.* Behavioral science, 12: pp 153-155

[8] Milligan G, Cooper M (1985) *An examination of procedures for determining the number of clusters in a data set*. Psychometrika, 50: pp 150-179

[9] SAS Institute Inc., *SAS technical report A-108 (1983) Cubic clustering criterion.* Cary, NC: SAS Institute Inc., 56 pp

[10] Sanjay Chakraborty, Prof. N.K.Nagwani, Lopamudra Dey ; *Weather Forecasting using Incremental K-means Clustering* https://arxiv.org/ftp/arxiv/papers/1406/1406.4756.pdf

[11] *Multi-Layer Perceptron – Back Propagation*
https://www.cse.unsw.edu.au/~cs9417ml/MLP2/BackPropagation

[12] Rui X, Wunsch DC II (2009*) Clustering*. IEEE Press series on computational intelligence, John Wiley & Sons

[13] Arpit Bansal, Mayur Sharma, Shalini Goel (2017); International Journal of Computer Applications; *Improved K-mean Clustering Algorithm for Prediction Analysis using Classification*

Technique in Data Mining 6(157)*; pp* 0975 –8887 https://pdfs.semanticscholar.org/f0c5/a423fcdbe7d5faa52d58b26a9bfd2ca36587.pdf

[14] Himesh Parmar, Swarndeep Saket (2017); IJARIIE*: Overview of Clustering Algorithm for Weather Data* 6(3); ISSN(O)-2395-4396 http://ijariie.com/AdminUploadPdf/Overview_of_Clustering_Algorithm_for_Weather_Data_ijariie7083.pdf

[15]*The SQL vs NoSQL Difference: MySQL vs MongoDB – Xplenty Blog - Medium* http://medium.com/xplenty-blog/the-sql-vs-nosql-difference-mysql-vs-mongodb-32c9980e67b2

[16] *How to Configure The Learning Rate Hyperparameter When Training Deep Learning Neural Networks* https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks