ANALYSIS OF INDIAN RAINFALL DATA
FOR
FUTURE PREDICTION

Project Report Submitted in Partial Fulfilment of the
Requirements for the degree of
Master of Computer Application
Department of Computer Science & Engineering
Jadavpur University
May, 2019

By
BISWASINDHU MANDAL
Master of Computer Application – III
Class Roll Number: 001610503028
Examination Roll Number: MCA196024
Registration Number: 137335 of 2016 – 2017

Under the guidance of
Dr. SARMISTHA NEOGY LAHIRI
Professor

Department of Computer Science and Engineering
Faculty of Engineering and Technology
Jadavpur University
Kolkata – 700032, India
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 while or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

NAME	**:**	Biswasindhu Mandal
Class Roll Number	**:**	001610503028
Examination Roll Number	**:**	MCA196024
Registration Number	**:**	137335 of 2016 - 2017
Project Title	**:**	Analysis of Indian Rainfall Data for Future Prediction
Signature with Date	**:**	

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

TO WHOM IT MAY CONCERN

I hereby forward the project report entitled **"*Analysis of Indian Rainfall Data for Future Prediction*"** prepared by **Biswasindhu Mandal** under my supervision to be accepted in partial fulfilment for the degree of **Master of Computer Application** in the Faculty and Technology of Jadavpur University, Kolkata.

_____

(Dr.Sarmistha Neogy Lahiri)
Professor
Project Supervisor
Dept. of Computer Science and Engineering
Jadavpur University
Kolkata – 700032

Countersigned:

_____

Dr. Mahantapas Kundu
Head, Dept. of Computer Science and Engineering
Jadavpur University
Kolkata – 700032

_____

Prof.Chiranjib Bhattacharjee
Dean, Faculty of Engineering and Technology
Jadavpur University
Kolkata – 700032

Department of Computer Science and Engineering
Faculty of Engineering and Technology
Jadavpur University

CERTIFICATE OF APPROVAL

This is to certify that the project titled **"Analysis of Indian Rainfall Data for Future prediction"** is a bona-fide record of work carried out by Biswasindhu Mandal in partial fulfilments for the award of the degree of Master of Computer Application, of the Department of Computer Science & Engraining, 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.

_____

Final Examination for
evaluation of the project

_____

(Signatures of Examiners)

ACKNOWELDGEMENT

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

Date:_____ _____

Biswasindhu Mandal
Master of Computer Application – III
Class Roll No. – 001610503028
Examination Roll No. – MCA196024
Registration No. – 137335 of 2016 – 2017
Jadavpur University

TABLE OF CONTENTS

Abstract:

Due to the impact of weather forecasting on global human life, and to better reflect the current trend of weather changes, it is necessary to conduct research about the prediction of precipitation and provide timely and complete precipitation information for climate prediction and early warning decisions to avoid serious meteorological disasters. For the precipitation prediction problem in the era of climate big data, we propose a new method based on deep learning.
In this paper, I will apply Artificial Neural Networks in rainfall precipitation forecasting on Indian Rainfall Data.

Introduction:

Weather is basically the way the atmosphere is behaving, mainly with respect to its effects upon life and human activities[1]. Weather is what is happening in the atmosphere at any time or short period of time[2]. Weather conditions can change suddenly. The climate of a place may be defined as a "composite" of the long-term prevailing weather that occurs at that location[3]. In a sense, climate is "average weather"[4]. The difference between weather and climate is a measure of time. Weather is what conditions of the atmosphere are over a short period of time, and climate is how the atmosphere "behaves" over relatively long periods of time[1]. An easy way to remember the difference is that climate is what you expect, like a very hot summer, and weather is what you get, like a hot day with pop-up thunderstorms[1].

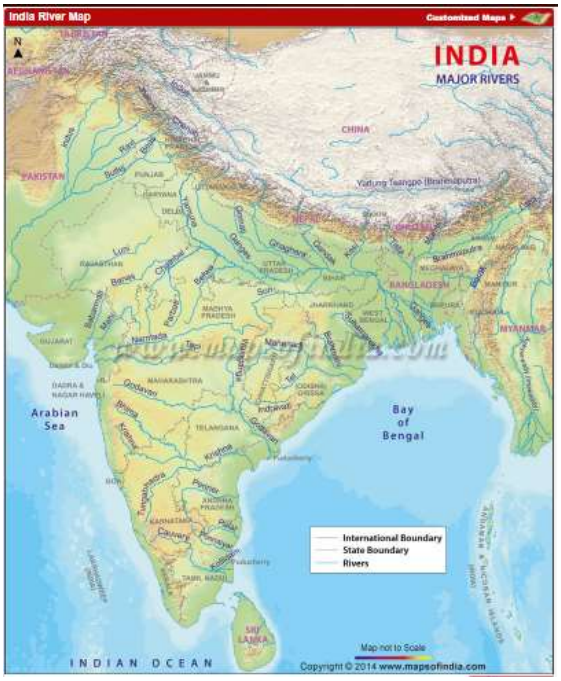
There are really a lot of components to weather. Weather includes sunshine, rain, cloud cover, winds, hail, snow, sleet, freezing rain, flooding, blizzards, ice storms, thunderstorms, steady rains from a cold front or warm front, excessive heat, heat waves and more. Some scientists define climate as the average weather for a particular region and time period, usually taken over 30-years. It's really an average pattern of weather for a particular region[1]. Climate is a critical factor in the lives and livelihoods of the people and socioeconomic development as a whole[5].

Climatology studies the spatial distribution of average values of climatic elements[6]. At the smallest scale, local climates influence areas maybe only a few miles or tens of miles across. Examples of local climatic phenomena include sea breezes and urban heating. At larger scales, climates provide a picture of particular patterns of weather within individual countries, or within climate zones that exist at different latitudes on the Earth. Climate zones include tropical, subtropical, desert, Savannah, temperate and polar climates. Different climate zones reveal variable patterns of temperature and rainfall. At the largest scale of all, climatologists study the global climate[3]. It is important to helps determine future climate expectations. It is most important factors in our environment and we need to have predictions about its behaviour to know how safe we are. It effects our food sources, our health, our homes etc. Rainfall and temperature are important climatic inputs for agricultural production, especially in the context of climate change. However, accurate analysis and simulation of the joint distribution of rainfall and temperature are difficult due to possible interdependence between them[7].

Rainfall is a product of climatic phenomena such as evaporation, condensation, vapour pressure and formation of cloud. It plays an important role in the assessment of climatic water balance of a region. It claims a first place in the practical importance as it controls humidity and aridity of a region, and consequently the agricultural efficiency. Moreover, it also plays a major role in the assessment of floods and drought events of a region. Because of its practical and climatological implications, it is vital to place interest on its characteristics - amounts, monthly and seasonal variations, percentage, intensity, variability and its areal distribution[6].

India is a riverine country. It falls under the active monsoon region of the world. Among the total area of 3,287,263 square kilometres (1,269,219 sq mi)[8] of this deltaic land, the river area alone holds 81% of the country[9]. Having an agro based economy, almost 60%-70% of the total population of this country are directly or indirectly related to agriculture. Hence, rainfall plays a crucial part in the economy of India. The erratic heavy rainfall events may affect ecosystems, agriculture, food security, urban. drainage, water availability, water quality and health and

livelihood of people of the country. Drought in India has resulted in tens of millions of deaths over the course of the 18th, 19th, and 20th centuries[10]. On the monsoon of 2018 heavy Monsoon rainfall from August 8 severe flooding affected the Indian state of Kerala resulting over 445 deaths[11]. On 2017 Gujarat state of India was affected by the severe flood resulting in more than 200 deaths[11]. Recent studies also suggest that the frequency and magnitude of heavy rainfall events have already been increased under the global warming scenario including the high altitude areas of India. So, 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. It is therefore, imperative to have an efficient rainfall estimation system for this natural disaster prone country.



In general, weather and rainfall are highly non-linear and complex phenomena, which require advanced computer modelling and recreation for their accurate prediction. Some mathematical techniques can be used to foretell the behaviour of such non-linear systems. There is a range of algorithms for prediction purpose. Hence the present investigation primarily involves data mining techniques. Firstly, through filtering we update the noisy data of huge dataset of a particular region. Secondly, to predict the future data by using different types of techniques.

This paper presents a methodology for forecasting rainfall of all regions of our country. Through clustering first we filter the data set of a specific month of a region and next using Artificial Neural Network(ANN) we predict the future rainfall. The project contains five sections or chapters. 1st section consists of Background of the project; which explain which database is suitable for this project and discuss algorithms. In 2nd section, mentioned here different related works with my project and my system configuration, database used, language versions etc. In 3rd portion, briefly explain about my proposed work, which means data collection, data processing, algorithms, code description etc. In section 4, discuss about my experiment and result. Also, in last 5th Section, discuss about conclusion and future scope.

Background

Database Choice: mongoDB

SQL - SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system.

NoSQL - A NoSQL database provides a mechanism for storage and retrieval of data that is modelled in means other than the tabular relations used in relational databases.

Difference between SQL and NoSQL database[12]

■ SQL databases are primarily called as Relational Databases (RDBMS); whereas NoSQL database are primarily called as non-relational or distributed database.

■ SQL databases are table based databases whereas NoSQL databases are document based, key-value pairs, graph databases or wide-column stores. This means that SQL databases represent data in form of tables which consists of n number of rows of data whereas NoSQL databases are the collection of key-value pair, documents, graph databases or wide-column stores which do not have standard schema definitions which it needs to adhered to.

■ SQL databases have predefined schema whereas NoSQL databases have dynamic schema for unstructured data.

■ SQL databases are vertically scalable whereas the NoSQL databases are horizontally scalable. SQL databases are scaled by increasing the horse-power of the hardware. NoSQL databases are scaled by increasing the databases servers in the pool of resources to reduce the load.

■ SQL databases uses SQL (structured query language) for defining and manipulating the data, which is very powerful. In NoSQL database, queries are focused on collection of documents. Sometimes it is also called as UnQL (Unstructured Query Language). The syntax of using UnQL varies from database to database.

■ SQL database examples: MySql, Oracle, Sqlite, Postgres and MS-SQL. NoSQL database examples: MongoDB, BigTable, Redis, RavenDb, Cassandra, Hbase, Neo4j and CouchDb

■ For complex queries: SQL databases are good fit for the complex query intensive environment whereas NoSQL databases are not good fit for complex queries. On a high-level, NoSQL don't have standard interfaces to perform complex queries, and the queries themselves in NoSQL are not as powerful as SQL query language.

■ For the type of data to be stored: SQL databases are not best fit for hierarchical data storage. But, NoSQL database fits better for the hierarchical data storage as it follows the key-value pair way of storing data similar to JSON data. NoSQL database are highly preferred for large data set (i.e for big data). Hbase is an example for this purpose.

■ For scalability: In most typical situations, SQL databases are vertically scalable. You can manage increasing load by increasing the CPU, RAM, SSD, etc, on a single server. On the other hand, NoSQL databases are horizontally scalable. You can just add few more servers easily in your NoSQL database infrastructure to handle the large traffic.

■ For high transactional based application: SQL databases are best fit for heavy duty transactional type applications, as it is more stable and promises the atomicity as well as integrity of the data. While you can use NoSQL for transactions purpose, it is still not comparable and sable enough in high load and for complex transactional applications.

■ For support: Excellent support are available for all SQL database from their vendors. There are also lot of independent consultations who can help you with SQL database for a very large scale deployments. For some NoSQL database you still have to rely on community support, and only limited outside experts are available for you to setup and deploy your large scale NoSQL deployments.
■ For properties: SQL databases emphasizes on ACID properties (Atomicity, Consistency, Isolation and Durability) whereas the NoSQL database follows the Brewers CAP theorem (Consistency, Availability and Partition tolerance)
■ For DB types: On a high-level, we can classify SQL databases as either open-source or close-sourced from commercial vendors. NoSQL databases can be classified on the basis of way of storing data as graph databases, key-value store databases, document store databases, column store database and XML databases.
■ Example of SQL database- MySQL Community Edition, MS-SQL Server Express Edition, Oracle Express Edition
Example of NoSQL database- MongoDB, CouchDB, Redis

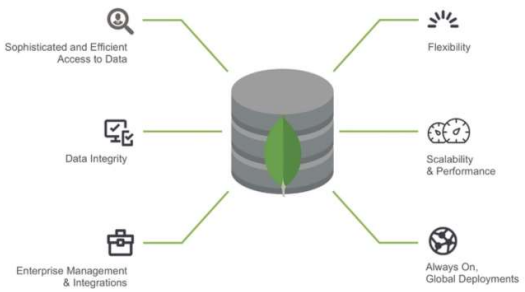
Problem of sql database[13][14]
1. Forgotten Primary Keys:
2. Poorly Managed Data Redundancy
3. Dependency on Order of Columns on ResultSet: When you use the **SELECT *** query in your application and have any dependency on order of column, which you should not, the ordering of result set will change if you add a new column or change the order of columns.
4. Forgotten NULL vs. Empty String Values
5. Copying data from one table to other: When you use **SELECT * into INSERT ... SELECT statement**, which is a common way to copy data from one table to another, you could potentially copy incorrect data into the incorrect column if the order of column is not same between two tables.
6. Conflict in JOIN Query: When you use SELECT * in JOIN query, you can introduce complications when multiple tables have columns with same name e.g. status, active, name etc.

Uses of NoSQL database[15]
- data sets are extremely large
- paradoxically the things you are tracking are quite small (like player stats)
- much data will never be queried or referenced
- you need nested data
- you need extremely fast in memory data
- you want globally distributed data
- your schema is flexible and changing
- you want to run different versions of your application against different evolving schema in the same database(s)
- you need 1000+ databases
- you have simple data requirements (key/value)

Introduce to MongoDB[16]
MongoDB is a non-relational database developed by MongoDB, Inc. MongoDB stores data as documents in a binary representation called BSON (Binary JSON). Related information is stored together for fast query access through the MongoDB query language. Fields can vary from document to document; there is no need to declare the structure of documents to the system –

documents are self-describing. If a new field needs to be added to a document, then the field can be created without affecting all other documents in the collection, without updating a central system catalogue, and without taking the system offline. Optionally, schema validation can be used to enforce data governance controls over each collection.

MongoDB's document data model maps naturally to objects in application code, making it simple for developers to learn and use. Documents give you the ability to represent hierarchical relationships to store arrays and other more complex structures easily.



Native, idiomatic drivers are provided for 10+ languages – and the community has built dozens more – enabling ad-hoc queries, real-time aggregation and rich indexing to provide powerful programmatic ways to access and analyze data of any structure.

Feature Comparison[16]

Like MySQL, MongoDB offers a rich set of features and functionality far beyond those offered by simple NoSQL data stores. MongoDB has a rich query language, highly functional secondary indexes (including text search and geospatial), a powerful aggregation framework for data analysis, faceted search, graph processing and more. With MongoDB you can also make use of these features across more diverse data types than a relational database, and you can do it at scale.

	MySQL	MongoDB	NoSQL Data Store
ACID Transactions	Yes	Yes	No
Flexible, rich data model	No	Yes	Partial: schema flexibility but support for only simple data structures
Schema governance	Yes	Yes	No
Expressive joins, faceted search, graphs queries, powerful aggregations	Yes	Yes	No
Idiomatic, native language drivers	No	Yes	No
Horizontal scale-out with data locality controls	No	Yes	Partial: no controls over data locality
Analytics and BI ready	Yes	Yes	No
Enterprise grade security and mature management tools	Yes	Yes	No
Database as a service on all major clouds	Yes	Yes	No

Uses of MongoDB[16]

Organizations of all sizes are adopting MongoDB because it enables them to build applications faster, handle highly diverse data types, and manage applications more efficiently at scale.

Development is simplified as MongoDB documents map naturally to modern, object-oriented programming languages. Using MongoDB removes the complex object-relational mapping (ORM) layer that translates objects in code to relational tables. MongoDB's flexible data model also means that your database schema can evolve with business requirements.

MongoDB can also be scaled within and across multiple distributed data centers, providing new levels of availability and scalability previously unachievable with relational databases like MySQL. As your deployments grow in terms of data volume and throughput, MongoDB scales easily with

no downtime, and without changing your application. In contrast, to achieve scale with MySQL often requires significant, custom engineering work.

Advantage of MongoDB[17]

1. Schema-less design
2. Scalability in managing Tera bytes of data
3. Rapid replica Set with high availability feature
4. Sharding enables linear and scale out growth w/o running out of budget
5. Support high write load
6. Use of Data locality for query processing

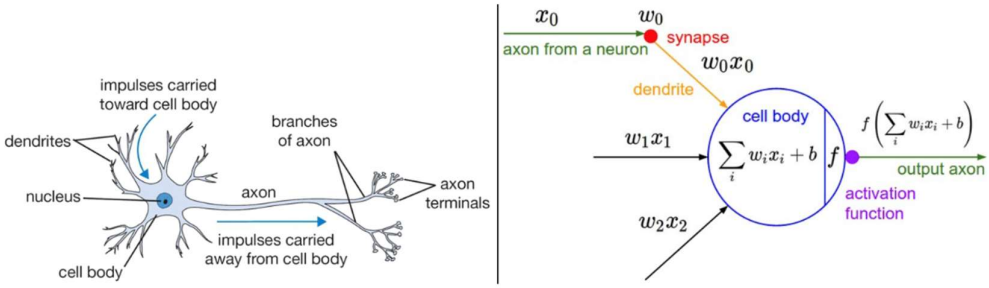
Advantage of mongodB 4.0[16]

MongoDB 4.0 added support for multi-document transactions, making it the only database to combine the ACID guarantees of traditional relational databases, the speed, flexibility, and power of the document model, with the intelligent distributed systems design to scale-out and place data where you need it. Through snapshot isolation, transactions provide a consistent view of data, and enforce all-or-nothing execution to maintain data integrity. Transactions in MongoDB feel just like transactions developers are familiar with in MySQL. They are multi-statement, with similar syntax (e.g. start_transaction and commit_transaction), and therefore easy for anyone with prior transaction experience to add to any application.

Prediction technique: Artificial Neural Network(ANN)[31][32]

To understand about ANN we have to learn about the Supervised Learning in Machine Learning. Supervised learning, in the context of artificial intelligence (AI) and machine learning, is a type of system in which both input and desired output data are provided. Input and output data are labelled for classification to provide a learning basis for future data processing. Supervised machine learning systems provide the learning algorithms with known quantities to support future judgments. Chatbots, self-driving cars, facial recognition programs, expert systems and robots are among the systems that may use either supervised or unsupervised learning[34].

Artificial Neural Network(ANN) is intended to simulate the behaviour of biological systems composed of “neurons”. ANNs are computational models inspired by an animal’s central nervous systems. It is capable of machine learning as well as pattern recognition. These presented as systems of interconnected “neurons” which can compute values from inputs[35]. Structure unit model of a neurons :--



There are 3 layers of ANN

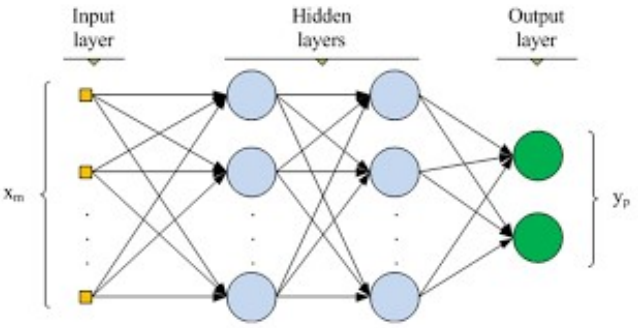
(1) Input layer, (2) Hidden layer and (3) Output layer

Input layer

The purpose of the input layer is to receive as input the values of the explanatory attributes for each observation. Usually, the number of input nodes in an input layer is equal to the number of explanatory variables. 'input layer' presents the patterns to the network, which communicates to one or more 'hidden layers'. The nodes of the input layer are passive, meaning they do not change the data. They receive a single value on their input and duplicate the value to their many outputs. From the input layer, it duplicates each value and sent to all the hidden nodes[35].

Hidden layer

The Hidden layers apply given transformations to the input values inside the network. In this, incoming arcs that go from other hidden nodes or from input nodes connected to each node. It connects with outgoing arcs to output nodes or to other hidden nodes. In hidden layer, the actual processing is done via a system of weighted 'connections'. There may be one or more hidden layers. The values entering a hidden node multiplied by weights, a set of predetermined numbers stored in the program. The weighted inputs are then added to produce a single number and add with an arbitrary number bias[35]. Actually, bias is an additional parameter in the Neural Network, which is used to adjust the output along with the weighted sum of the inputs to the neuron. Therefore, Bias is a constant, which helps the model in a way that it can fit best for the given data.

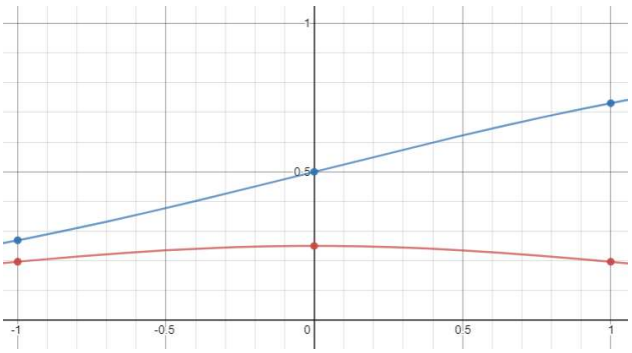


Output layer

The hidden layers then link to an 'output layer'. Output layer receives connections from hidden layers or from input layer. It returns an output value that corresponds to the prediction of the response variable. In classification problems, there is usually only one output node. The active nodes of the output layer combine and change the data to produce the output values.
The ability of the neural network to provide useful data manipulation lies in the proper selection of the weights. This is different from conventional information processing[35].

Activate Functions: Sigmoid function(σ) & derivative-sigmoid function(σ')[18]

Sigmoid functions are often used in artificial neural networks to introduce non-linearity in the model. A neural network element computes a linear combination of its input signals, and applies a sigmoid function to the result. A reason for its popularity in neural networks is because the sigmoid function satisfies a property between the derivative and itself such that it is computationally easy to perform.
Derivatives of the sigmoid function are usually employed in learning algorithms.



$\boldsymbol{Sigmoid}: \boldsymbol{\sigma(x)} = \frac{\mathbf{1}}{(\mathbf{1}+\boldsymbol{e}^{-\boldsymbol{x}})}$; $-\mathbf{1} \leq \boldsymbol{x} \leq \mathbf{1}$; $\boldsymbol{blue}$

$\boldsymbol{derv_{Sigmoid}}$: $\boldsymbol{\sigma}^{'(x)} = \frac{-\boldsymbol{e}^{-\boldsymbol{x}}}{(\mathbf{1}+\boldsymbol{e}^{-\boldsymbol{x}})^{\mathbf{2}}}$; $\mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1}$; $\boldsymbol{red}$

Learning Rate[19]

Learning rate is a hyper-parameter that controls how much we are adjusting the weights of our network with respect the loss gradient. The lower the value, the slower we travel along the downward slope. While this might be a good idea (using a low learning rate) in terms of making sure that we do not miss any local minima, it could also mean that we'll be taking a long time to converge — especially if we get stuck on a plateau region.

new_weight = existing_weight ± learning_rate * gradient

Methodology of ANN:

As ANN is a part of Supervised Learning. Therefore, there are some input and output data for learning. So, with in learning/training procedure of propagation method, there are two main part of strategy. (1) Forward Propagation and (2) Backword Propagation.

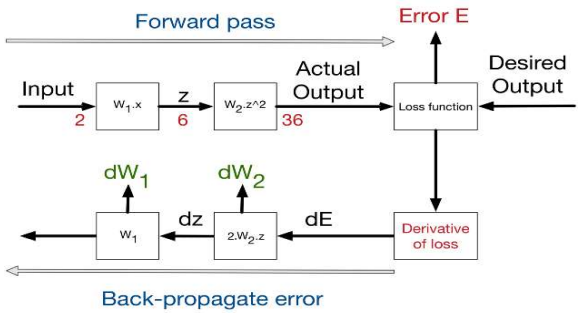
Forward Propagation

In Forward Propagation, sample weights are input to the ANN through the inputs and the respected sample outputs are recorded. Here, the inputs are fed and outputs for the inputs are received. The input X provides the initial information that then propagates to the hidden units at each layer and finally produce the output: y'. The architecture of the network entails determining its depth, width, and activation functions ($\sigma(x)$) used on each layer. Depth is the number of hidden layers. Width is the number of units (nodes) on each hidden layer since we do not control neither input layer nor output layer dimensions.

Backword Propagation

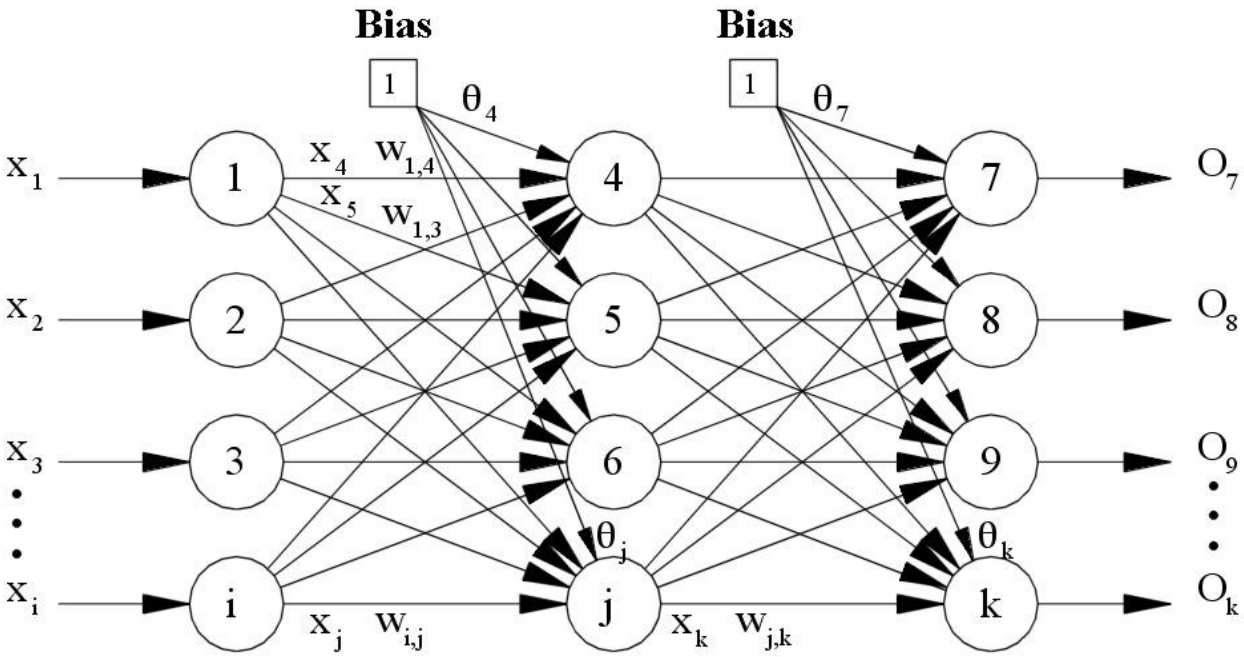
Allows the information to go back from the cost backward through the network in order to compute the gradient. Therefore, loop over the nodes starting at the final node in reverse topological order to compute the derivative of the final node output with respect to each edge's node tail. Doing so will help us know who is responsible for the most error and change the parameters in that direction. The activation function derivatives' formulas ($\sigma'^{(x)}$) will help us write the back-propagate functions.

So, training algorithm of propagation involves 4 stages :



- Initialization of weights – Initialize weight and bias 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 – 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. Using of $\sigma'^{(x)}$) we remove backword propagation error.
- Updation of weights and biases using learning rate. Where ,

$$new_{weigh} = old_{weigh} + learningRate * gradiant$$



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.

Related Work

Researchers have working on different data mining, machine learning techniques to predict climate/weather forecasting result. Fahmida tasnim prema [28], applied Data Mining technique. He used Algorithms of Decision tree, K-means clustering and regression model to estimate the rainfall. The outputs of the algorithms show a straightforward relationship between rainfall and the input parameters. In [29], authors S. Zhang, L. Lu, J. Yu and H. Zhou, performed a comparative analysis of Support Vector Machine (SVM), Artificial Neural Networks (ANN), and Adaptive Neuro Fuzzy Inference System (ANFIS) on rainfall prediction. The authors have compared the prediction models in four terms: (i) by using different lags as modelling inputs; (ii) by using training data of heavy rainfall events only; (iii) performance of forecasting for 1 hour to 6 hours and; (iv) performance analysis in peak values and all values. According to results ANN performed better when trained with dataset of heavy rainfall. Z. Ismail [30] proposed a forecasting model for prediction of gold price using linear regression. Author used factors such as inflation, money supply and concluded that MLR perform better than Naïve method of prediction.

Proposed Work

Objective

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 is the most important and challenging role in the matter of all living beings. It plays an essential position for countries' economic growth and development in the modern world. Forecasting heavy rainfall events has been a challenging job for India due to its variety of geographical land. Indian climate is the major factors behind south Asian monsoon. The heavy rainfall create flood in different area of India. In 2012 Brahmaputra floods was again due to the Brahmaputra river and its tributaries, affected Kaziranga National Park [20]. In 2014 Kashmir Floods was partially affected 1000 villages of Jammu and Kashmir state and few city areas were submerged under water[20]. Oppositely, same kind of problem have to face in drought in India. During the summer of 2002, drought conditions persisted over India from mid-June to mid-July following a later-than-average arrival of seasonal monsoon rain. By the end of that summer, the average summer Indian rainfall totalled 711 mm or 19% below normal [21]. So, for everything there are lot of economical loss of India. In 2017, the total damage due to floods / heavy rains amounted to Rs.18,279.63 crores (US$ 2.5 billion), including damage to crops, houses and public utilities, according to data from the Central Water Commission.[22]. Recent studies also suggest that the frequency and magnitude of heavy rainfall events have already been increased under the global warming scenario including the high altitude areas of India. It is therefore, imperative to have an efficient rainfall estimation system for this natural disaster prone country.

In this project work, I do to forecasting Indian rainfall data using Artificial Neural Network

Steps followed by, Data Collection, Data Processing/Grouping, Data insertion into Database, Prediction of client Query, Prediction Method (I use ANN algorithms), View Result

Data Collection:

Monsoon data set has been received from the URL `https://data.gov.in/keywords/annual-rainfall`. In this dataset, around 107 years rainfall data are available starting from 1901 to 2017. But, I take rainfall data of 67 years from 1951 to 2017. Because of before our independence there are no specific state in India and my regions values are missing, noise, etc. But, after 1951 there are stable in Indian regions/states. Therefore, I decided to take rainfall data from 1951 to 2017.

Collection data from that website in csv/excel format like as:

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

Lakshadweep	2010	18.8	0	1.2	35.6	79	318.9	336.7	335.1	161.5	155.4	201.5	81.5
Lakshadweep	2011	5.1	2.8	3.1	85.9	107.2	153.6	350.2	254	255.2	117.4	184.3	14.9
Lakshadweep	2012	19.2	0.1	1.6	76.8	21.2	327	231.5	381.2	179.8	145.9	12.4	8.8
Lakshadweep	2013	26.2	34.4	37.5	5.3	88.3	426.2	296.4	154.4	180	72.8	78.1	26.7
Lakshadweep	2014	53.2	16.1	4.4	14.9	57.4	244.1	116.1	466.1	132.2	169.2	59	62.3
Lakshadweep	2015	2.2	0.5	3.7	87.1	133.1	296.6	257.5	146.4	160.4	165.4	231	159
Lakshadweep	2016	59.6	12.1	3.2	2.6	77.4	321.1	262.6	86.2	75.6	58.6	32	74.7
Lakshadweep	2017	21.3	0.9	100.2	1.8	145.7	521.9	164.2	206.2	216	137.1	63.5	160.1

Here all subdivisions/regions are:

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

I fixed it with in code, if you search a region using its state/union territory full name/short name/first 3-4 letters, then you find out that region and show you if it has subdivision as an input region.

Data Processing:

We directly do not store data, which directly lies in, excel/csv format. First, we have to understand how to store this data into database. So, I decided to store data in the format and for that reason we have to group data in format of {Region, Year, Month, Rainfall-Data} form csv/excel format collection data. After data grouping, we get a huge amount of data = no_of_diff_subdivision X 63 Years X 12 months = 28944.

Region: LAKSHADWEEP
Year: 2017
Month: JJANUARY
Data: 21.3

```
{
    "REGION": "GANGETIC WEST BENGAL",
    "YEAR": 1993.0,
    "MONTH": "MARCH",
    "DATA": 40.8
}
```

Data insertion in mongo-DB:

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

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:

```
/* 28943 */
{
    "_id" : "rainfall28942",
    "REGION" : "LAKSHADWEEP",
    "YEAR" : 2017.0,
    "MONTH" : "NOVEMBER",
    "DATA" : 63.5
}

/* 28944 */
{
    "_id" : "rainfall28943",
    "REGION" : "LAKSHADWEEP",
    "YEAR" : 2017.0,
    "MONTH" : "DECEMBER",
    "DATA" : 160.1
}
```

... ...

... ...

```
/* 1 */
{
    "_id" : "rainfall0",
    "REGION" : "ANDAMAN & NICOBAR ISLANDS",
    "YEAR" : 1951.0,
    "MONTH" : "JANUARY",
    "DATA" : 82.7
}

/* 2 */
{
    "_id" : "rainfall1",
    "REGION" : "ANDAMAN & NICOBAR ISLANDS",
    "YEAR" : 1951.0,
    "MONTH" : "FEBRUARY",
    "DATA" : 7.2
}
```

User/Client Prediction Query:
Actually, there are two types of user/client query.
(i) Predict rainfall data in a specific region and a year
(ii) Predict rainfall data in a specific region, a year and month

1. To understand Code Structure we have to learn something:
utils: Utility Class, also known as Helper class, is a class, which contains just static methods, it is stateless and cannot be instantiated. It contains a bunch of related methods, so they can be reused across the application[23].
As an example, consider Apache StringUtils, CollectionUtils or java.lang.Math.
Within my project I use some utils file with in **share/utils directory**.
(i) Utils.py: use to the function which are used in anywhere
(ii) dbUtils.py: use the function with in database method
(iii) expectUtils.py: for expecting future data use its methods

config: configuration files (or config files) are files used to configure the parameters and initial settings of the Application[24].
With in my project I use **config,json** file with in **share directory**. Here I initialized:
(i) source data path(RainFallData.xlsx)
(ii) Database configuration(I used mongo DB)

2. Importance Packages that I use here:
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"*[wiki]*.

json: The full form of JSON is JavaScript Object Notation, is inspired by a subset of the JavaScript programming language dealing with object literal syntax. They've got a nifty website that explains the whole thing. The process of encoding JSON is usually called serialization. This term refers to the transformation of data into a series of bytes (hence serial) to be stored or transmitted across a network. You may also hear the term marshaling, but that's a whole other discussion. Naturally, deserialization is the reciprocal process of decoding data that has been stored or delivered in the JSON standard[33].
---- There is some static data use in project in format of json file. To parse them we use json package. Some example of the static data are: database connection url-port, databse name, indian all state name in short format etc.

xlrd: library for 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 pure Python with no dependencies on modules or packages outside the standard Python distribution. It has strong support for Excel dates and is Unicode-aware[25].
---- Here I use it to read collection rainfall data in excel format

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 you need to know to use PyMongo[26].
---- Here I use it to insert/update/delete/fetch data from database mongoDB

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[27].
---- Here I use it for data prediction method ANN algorithms.

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+.*[wiki]*
---- Here use to plot expect data of a region including/excluding previous year data

Algorithms:
There are two parts of My Work.
1. Store Data into Data Base
2. Expecting Rainfall Future Data

Part I: Store Data into Database
Step01: Set, **A[], B[]**
Step02: Read, **x** = read csv/excel data path,
Step03: for each row, i in **x**, csv/excel file:
A[i] = i-th row data
endfor
--- where A[0] represent the csv header
Step04: now we create data-structure like, [region, year, month, data] from A depend on csv/excel header and Store into B
So, each **B[i]**, **i = 0,1,2...n**; represent **a particular region, a particular year, a particular months rainfall data**
Step05: create Database Connectivity & store all **B[i]; i = 0,1,2...n** with in database using BD insert query
Step06: End

Part II: Expecting Future Rainfall data
Step01: read, **cQuery** = client Expecting Query
Step02: Calculate, **dbQuery** = to fetch data from database calculate database query depend on **cQuery.** Whare cQuery may or may not be a array of database query.
Step03: Calculate, **dbData** = fetch database data according to **dbQuery**
Where dbData is an array, where each dbData[i] contains **a region's particular year-months rainfall data** in format [region, year, month, data]
For expecting future data I use ANN algorithem
Step04: Calculate, **trData**(**trInput**, **trOutput**) = ready trainng data from **dbData** for ANN
Step05: Calculate, **exIData** = ready expecting input data to see both **dbData** and **cQuery**
Step06: Normalized both **trData** & **exIData**
Step07: calculate, **exOData** = **ANNAlgorithem(trData,exIData)**; get expecting output data
Step08: if(**exOData** match according to **cQuery**) goto **Step10**
Step09: include exOData with dbData of format [region, year, month, data]
Step10: goto **Step04**
Step11: print(**exOData**)

Step12: End

function ANNAlgorithem():
Step01: read X = **trInput**(normalized them)**,** training input
Y = **trOutput**(normalized them)**,** training output
T = **exIData**(normalized them)**,** testing input
Step02: Set, hh = each hidden layer height
ly = no of hidden layer
lr = learning rate in [0.0, 1.0]
step03: let, W = [w1,w2,w3..,wn]; set of weights of each hidden layer
B = [b1,b2,b3...,bn]; set of bias of each hidden layer
where,
wi = set of random numbers between (-1, 1) of order (length(X), length(hh))
bi = set of random numbers between (-1, 1) of order (1, length(hh))
step04: Set, Z[]
step05: training data
for i=0, lr :
Calculate, Z = **forward**(X,W,B); an array
Calculate, (W,B) = **backword**(X,Y,Z,lr)
Endfor
step06: testing data
result= **testing**(T,W,B)
return **result**

function testing(T,W,B) :
$z_1 = sigmoid((T.w_1)+b_1)$
$z_2 = sigmoid((h_1.w_2)+b_2)$
$z_3 = sigmoid((h_2.w_3)+b_3)$
. . . . .
$z_n = sigmoid((h_n\text{-}1.w_n)+b_n)$
return $\mathbf{z_n}$

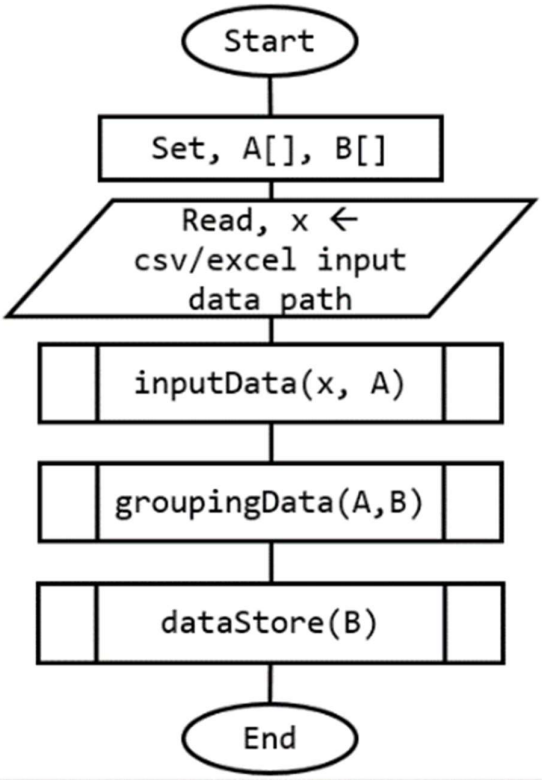
function forward(X, Wi, Bi):
$z_1 = sigmoid((X.w_1)+b_1)$
$z_2 = sigmoid((z_1.w_2)+b_2)$
$z_3 = sigmoid((z_2.w_3)+b_3)$
. . . . .
$z_n = sigmoid((zn\text{-}1.w_n)+b_n)$
return $[z_1,z_2,z_3...z_n]$

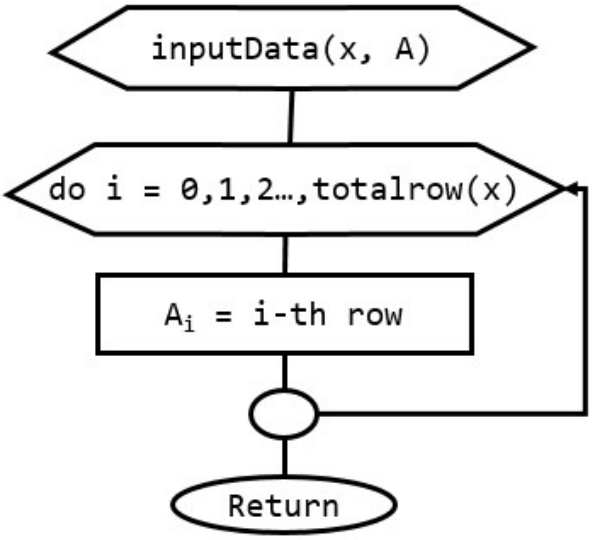
function backword(X,Y, Z,lr):
$a_1 = derv\_sigmoid((z_n * (Y-z_n))$; use it for calculate error
$a_2 = derv\_sigmoid((z_{n-1} * (a_1.W_n^T))$
$a_3 = derv\_sigmoid((z_{n-2} * (a_2.W_{n-1}^T))$
. . . . . .
$a_n = derv\_sigmoid((z_1 * (a_{n-1}.W1^T))$

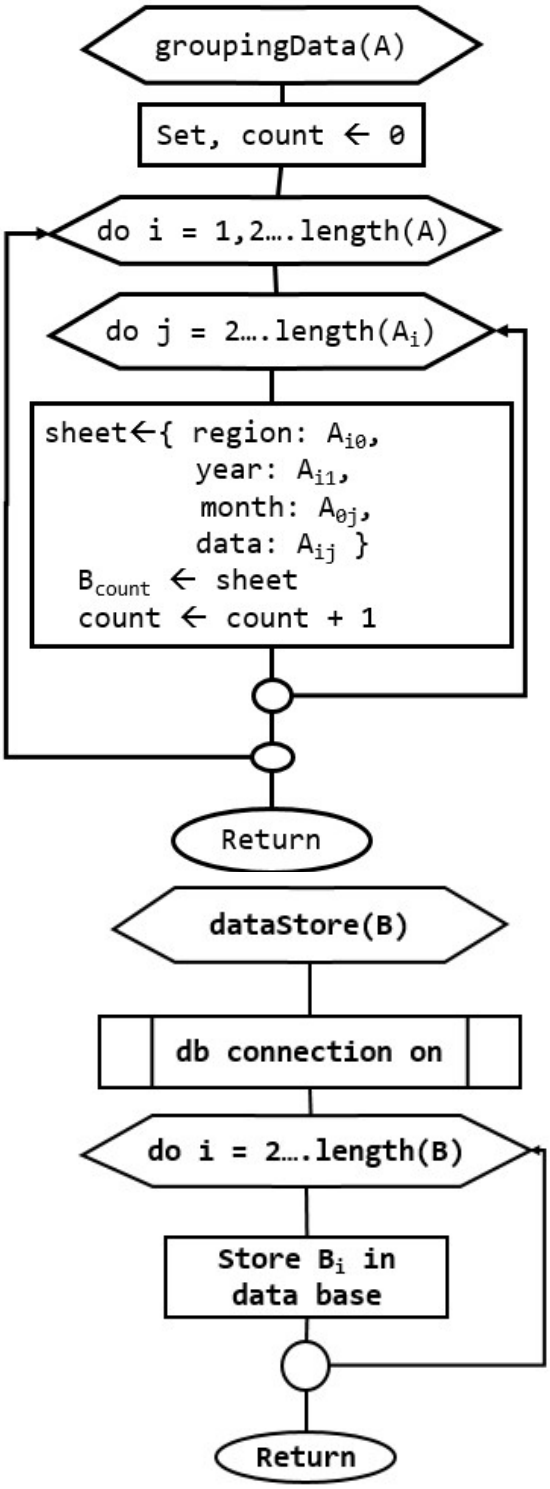
w_1 += lr * (X^T.an), b_1 += sum(a_n)
w_2 += lr * ($z1^T$.a_{n-1}), b_2 += sum(a_{n-1})
w_3 += lr * ($z2^T$.a_{n-2}), b_3 += sum(a_{n-2})
. . . .
w_n += lr * (z_{n-1}^T.a1), B_n += sum(a_1)
return([w_1,w_2,w_3...w_n], [b_1,b_2,b_3...b_n])

Flowcharts:

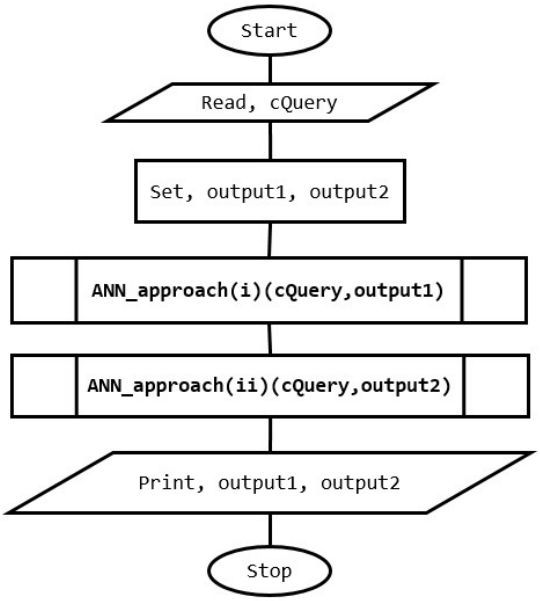
Part I: Store Data into Database

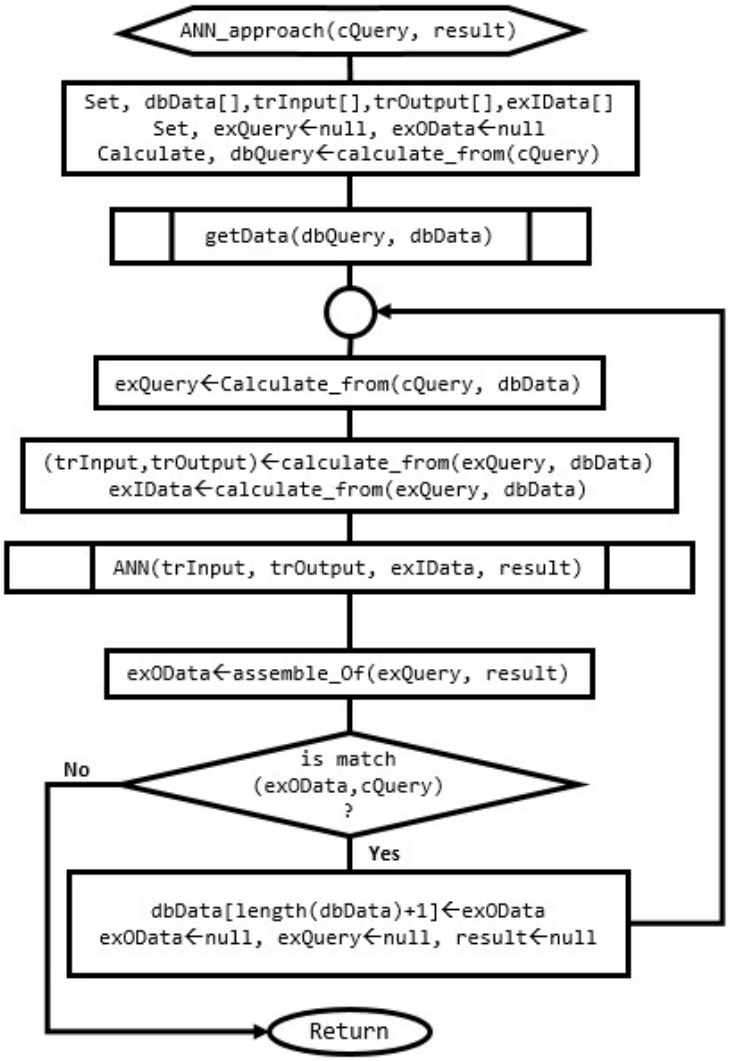


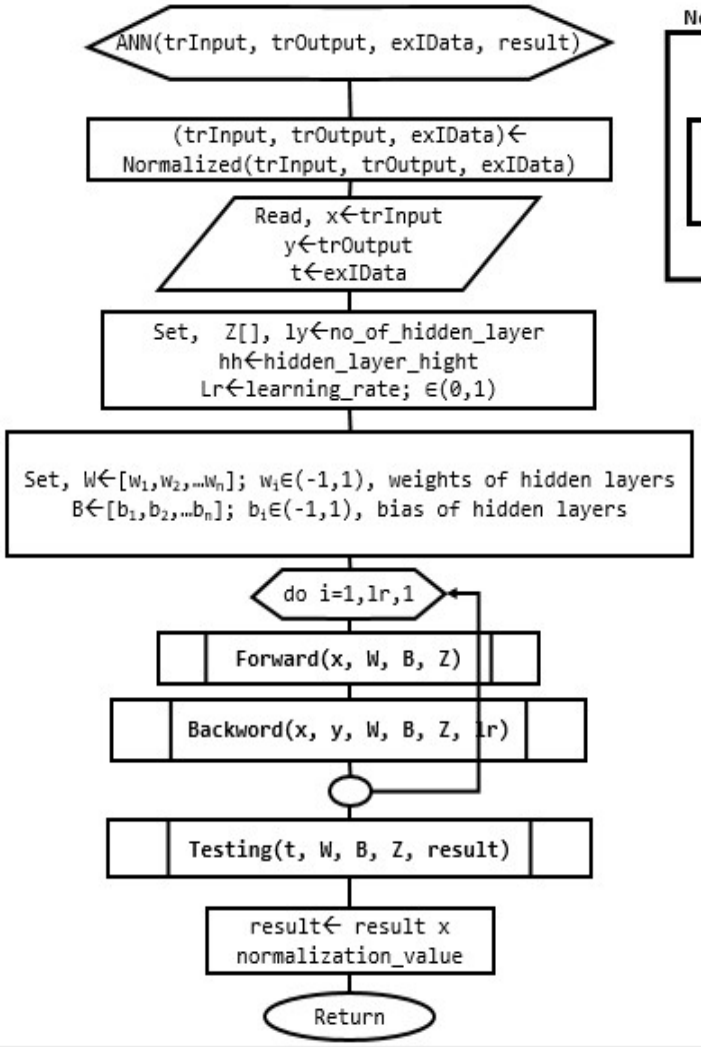


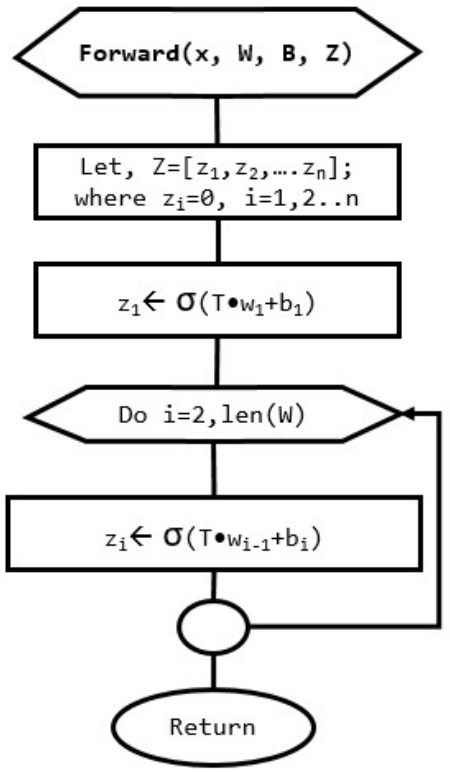


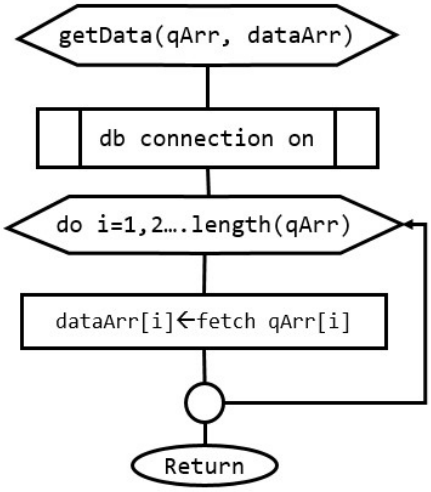
Part II: Expecting Future Rainfall data

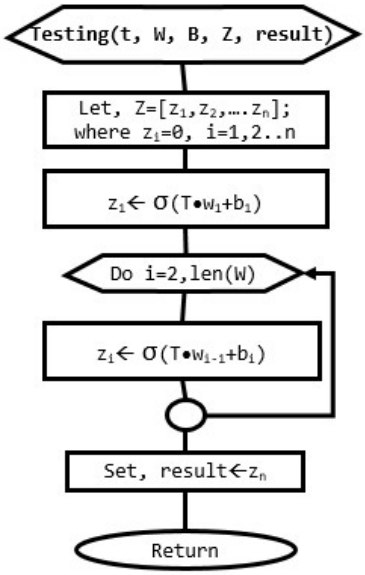


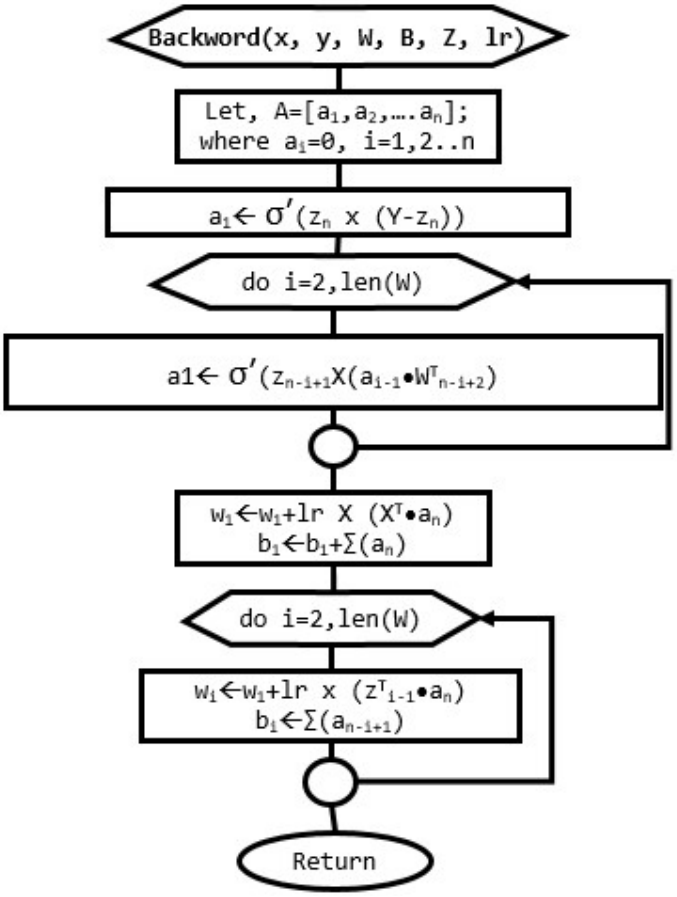












Code Structure:

My full code structure is here:

main.py is the starting point of the code. Within **src** we write rest of code by creating a directory, which show its functionality. Within data directory store input data in 'excel' format. 'main.py' call 'src/controller/controller.py' every time. From 'controller.py' we control rest of code.

```
RainFallDataAnalysis
|
|    main.py
|---src
|    |---controller
|        |       controller.py
|    |---excel
|        |       xlsxInput.py
|    |---grouping
|        |       groupingData.py
|    |---mongoo
|        |       mongoDBController.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
|---data
        RainFallData.xlsx
```

Code details

Name	Description
main.py	Entry point into the application.

cofig.json	Configuration file for initial settings of Application.
	code details
	{
	"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" : 5,
	"hiddenSize": 5,
	"learningRate": 0.2,
	"trainingIteration": 1000
	}
	}

utils.py	Utility class or known as helper class of the project. Which contain all Static Method code details import **json :** to fetch config.json data class **Utils:** • **jsonData(searchItems):** use to search ".json" file data • **menuDriven(choice):** • **int_or_float_str(s):** • **month_number_to_string(num):** • **month_string_to_number(num):**
controller.py	It is the key/heart of the project. code details class **Controller** • **__init__(self,dbConfig,dbName) :** Constructors of Controller, basically used to connect the Database(mongo DB) used dbConfig from config.json and

	• **__del__(self) :** Destructors of controller, basically used to close connection Database(mongo DB) • **postData(self,dataPath):** use to (i) take data from excel/csv file from **xlsxInput.py** (ii) grouping data use of csv data in the specific format from **groupingData.py** { Region, Year, Month, Data } (iii) save/update data into Database(I used mongo DB) through **mongodbController.py** • **deleteData(self):** use to clear Database through **mongodbController.py** • **analysisData(self):** to expect future data follow the way: get client query and generate db query acc. to client query from **expectingQuery.py** get data from database w.r.t. to db query send db data to **annController** to expect searching data plot expect data (including previous data or not) through **Self.drawGraph()** • **drawGraph(self,pData,queryType,status) :** calling **plotGraphController.py** to plot data
dbConnection.py	Using for connect db (mong DB) Import **pymongo** class **MongoConnection:** • **__init__(self,dbConfig,collectionOriginalName) :** Constructor to fetch mongourl, dbName, dbCollection • **Start(self) :** Use to start db connection on • **End(self) :** Use to closed db connection
xlsxInput.py	Used to fetch data from a ms-excel format using python package **xlrd.** code structure: Import **xlrd** class **ExcelInput** : • **__init__(self,path) :** Self.path = path Self.sheet = self.excelInput() Self.data = gropingData() • **excelInput(self) :** take data from excel file using xlrd package and store into self.sheet • **processingData(self) :** processing self.sheet.nrows data and save python list-type data with in self.data --- use **self.data** with in controller.py
gropingData.py	Processing excel return data give it a specific format to store into database. Code structure:

	Import **utils.py** : to use **int_or_float_str() method** class **DataSheet: (**which is a metaClass) • **__init__(self,region,year,month,data) :** "REGION" : "GANGETIC WEST BENGAL", "YEAR" : 1993.0, "MONTH" : "MARCH", "DATA" : 40.8 Use to create a sheet like : (RainFallData) class **GroupingData:** • **__init__(self, xlsxData):** Self.dataList = xlsxData Self.dataGroup = self.processDataForGrouping() • **processDataForGrouping(self):** using metaclass **DataSheet** convert excel data to a beautiful format --- use **self.dataGroup** with in controller.py
mongodbController.py	Use to store/fetch/update/delete data in database class **MongoRequest :** • **def __init__(self,collection) :** use to set mongo collection • **def postData(self, data) :** use to store data into database. • **def deleteData(self, query) :** use to delete/clear database • **def getData(self, query) :** use to fetch data from database --- all of three method return a specific sms/data
expectingQuery.py	Use to get expectingQuery and generate dbQuery according to class **ExpectingQuery :** • **def __init__:** self.cQuery = get clientQuery from **clientQuery.py** self.mQuery0 = if(expectQueryYr<2017) then generate query for display original data which lies on db self.mQuery1 = db query of ANN-1 approach self.mQuery2 = db query of ANN-2 approach --- send **cQuery,mQuery0,mQuery1,mQuery2** to **controller.py**

annController.py
(annController1.py,
annController2.py)
	def main(cQuery,dbData) :
	while(true) :
	expectQuery = generate expected query to see (cQuery, dbData)
	send `expectQuery,dbData` to **processingANNdata.py** to find out **annInput, annOutput, annQInput**
	now normalized 'annInput, annOutput, annQInput' and calling **ANNalgo.py** to get **annQOutput** rainfall prediction
	merge expectQuery, annQOutput with dbData
	if(cQuery match with exQuery) break
	--- send dbData to controller.py

processingANNdata.py	Here, use **class ProcessOnlyData_of_ANN** process ANN (Artificial Neural Network) data. I use two approach to predict future rainfall data.

	Here I use two approach **ANN-approach-1, ANN-approach-2** to ready data for ANN algorithm. Both all them discuss after code-details. According to both approach here ready **annInput, annOutput, annQInput** data --- send annInput, annOutput, annQInput to annController.py
annAlgo.py	Here we write code of Artificial Neural Network algorithms using annInput, annOutput, annQInput data and find out annQOutput --- send annQOutput to annController.py
plotgraphController.py	Here I plot the data graphically. Actually here I plot both approach data. If your query year of any region below 2018, then in this case we plot with original data. With in graph, dataProcess.py help to data processing of ploting data, like a rainfall data separate, graph both axis's sms etc.

ANN-approach-1: If we predict rainfall of a region 'reg', year 'yr' of month 'x'. Then in this approach we pick up all data of that region all year of x-month. According to my data base there is 1950 to 2017 data. So, if we predict west-Bengal, 2019, march data, then pick up all west-Bengal, march data from 1950 to 2017, as follow below,--

62.1, 23.3, 4.8, 0.3, 11.4, 50.7, 8.7, 13.3, 17.3, 30.9, 6.8, 6.0, 5.7, 18.0, 46.5, 5.1, 44.9, 7.7, 20.9, 30.8, 4.8, 4.1, 67.6, 83.1, 10.6, 20.7, 7.3, 50.5, 8.4, 60.9, 64.8, 87.2, 47.0, 0.6, 8.8, 7.1, 18.9, 53.5, 6.1, 119.0, 41.7, 0.1, 40.8, 7.3, 8.7, 10.4, 44.2, 128.8, 1.6, 3.8, 32.8, 26.6, 70.0, 14.9, 82.8, 10.6, , 11.5, 17.9, 8.0, 40.5, 4.4, 4.8, 19.9, 19.3, 14.7, 35.2

As we have 1950 to 2017 data. So, my first query is west-bengal,2018,march and using of 2018 march result my next query is west-bengal,2019,march

Now to create ANN input/output data I decided a input length for ANN algorithms. Let ANN input length = 5. So, I take annInput, annOutput as first 5 month is annInput and next month is annOutput, now again start from 2nd month and upto 5 month data is take annInput and next month is annOutput…continue.
So, the result is,

annInput	annOutput
[62.1, 23.3, 4.8, 0.3, 11.4]	[50.7]
[23.3, 4.8, 0.3, 11.4, 50.7]	[8.7]
[4.8, 0.3, 11.4, 50.7, 8.7]	[13.3]
[0.3, 11.4, 50.7, 8.7, 13.3]	[17.3]
... ...	...
... ...	...
[10.6, 39.3, 11.5, 17.9, 8.0]	[40.5]
[39.3, 11.5, 17.9, 8.0, 40.5]	[4.4]
[11.5, 17.9, 8.0, 40.5, 4.4]	[4.8]
[17.9, 8.0, 40.5, 4.4, 4.8]	[19.9]
[8.0, 40.5, 4.4, 4.8, 19.9]	[19.3]
[40.5, 4.4, 4.8, 19.9, 19.3]	[14.7]
[4.4, 4.8, 19.9, 19.3, 14.7]	[35.2]

Now we take similarly create annQInput, annQoutput as:

annQInput: [4.8, 19.9, 19.3, 14.7, 35.2]

annQoutput: [?]; predict result of west-bengal,2018,march

Calling through '**annAlgo.py**' we get result **y**. So, including west-bengal, 2018,march next adding another annInput, annOutput :

add annInput: [4.8, 19.9, 19.3, 14.7, 35.2]

add annOutput: [y]

Now create annQInput, annQoutput as:

annQInput: [19.9, 19.3, 14.7, 35.2, y]

annQoutput: [?]; predict result of west-bengal,2018,march

There is an another problem to predict the total year rainfall result of any region/state. So, take all data from 1950-2017 of query region. If our query is **Bihar, 2020**. Then our first query is to predict Bihar, 2018 result. We apply above method on all month from January-December. Then adding result of Bihar, 2018 result predict Bihar, 2019 result and then in same way predict Bihar, 2020.

ANN-approach-2: This is another approach of create ANN data set. According to above if ANN input length = 5. Here we take a month previously 5 month data to create ANN input-output data. For example if query is **Kerala, 2019, October**. Then in this approach, we have to pick up all data of region Kerala from 1950-2017 in month of 'October' and its previous 5 months data i.e. 'April, May, June, July, September'. a.e.,

Year	annInput Data	annOutput Data
1951	[6.5, 41.6, 175.9, 148.5, 774.1]	[544.6]
1952	[22.6, 18.5, 132.4, 55.4, 340.5]	[1027.6]
1953	[6.3, 28.2, 125.9, 544.2, 782.4]	[392.8]
1954	[16.0, 25.7, 70.2, 381.2, 872.0]	[835.3]
…	… … …	…
…	… … …	…
2015	[11.7, 15.1, 151.6, 351.3, 755.4]	[466.8]
2016	[2.8, 90.6, 136.9, 179.5, 798.3]	[640.5]
2017	[48.2, 20.8, 112.2, 214.6, 576.7]	[430.0]

There is a drought; we does not directly find first Kerala, 2018, October data according as above. Because we does not have any data of 2018. So, to solve this problem, we have first predict January, 2018 data according this method, then using of January, 2018 data predict February, 2018 data and continue same way till October, 2019.

If query is for a year, then we start from January 2018 and continue up to December, searching year. Then show that year January to December data

Experiments & Results:

Project Related Tools:

SYSTEM CONFIGURATION: RAM 4GB (min), Processor 64bit
OPERATING SYSTEMS: WINDOWS XP/7/8/8.1/10, UNIX, LINUX
LANGUAGES: Python 3.6/3.7
RDBMS/BACK END: MongoDB 4.0
RELATED PACKAGES/TOOLS: xlrd, json, pymongo, numpy, matplotlib

Part1: data insert/update into database
Excel data:

SUBDIVISION	YEAR	january	february	march	april	may	june	july	august	september	october	november	december
Orissa	1951	2.2	5.1	89.7	37.1	48	172.5	339.1	349.5	165.5	155.6	37.1	0
Orissa	1952	1.7	9.5	26.2	53.2	44.4	187.1	356.4	352	307.9	176.6	2.2	1.1
Orissa	1953	39.9	5.9	0.2	13.8	23.7	182.8	318.9	512.1	231.5	59.2	49.8	0
Orissa	1954	0.6	6.2	8.6	19.2	50.2	163.2	211.7	271.9	344	118.2	0	14.2
Orissa	1955	0.1	0.2	8.4	29.6	76.5	201.7	238	355.9	371.4	262.7	52.3	0.3
Orissa	1956	1.4	37.9	18.2	11.9	118.1	345.2	470	434.5	249.8	191.7	6.7	0
Orissa	1957	12.3	26.6	30.4	12.4	33.6	153.5	349.2	415.4	194.5	49.5	0.1	0
Orissa	1958	11.5	42.8	16	32.8	39	125.2	404.1	319.1	364.2	217.4	51.8	0.1
Orissa	1959	28.5	2.4	2.7	22.2	47.1	202.7	357.4	341.6	316.5	170.5	1.5	2.7
Orissa	1960	5.4	0.8	51.6	12.4	42.7	211.4	444.1	461.7	214.6	130.8	3.6	13.9
Orissa	1961	15.9	16.2	3.2	15.9	59.7	324.8	406.2	327.9	483.2	178.2	11.3	2.7

Choose 1, to insert/update data into mongodb

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     established mongo connection
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

0. Exit

1. Insert/Update Data from Excel Format to MongoDB

2. Delete/Clear Data

3. Predict Future Data with Both Approach

        Your choice:    1
Total data:  28944
....data insert/update start....
...1000 data inserted/updated......
...2000 data inserted/updated......
...3000 data inserted/updated......
...4000 data inserted/updated......
...5000 data inserted/updated......
...6000 data inserted/updated......
```

insertion:

After complete data insertion show sms:

```
----Insertd/updated 28944 Data Successfully----
```

mongodb data after

```
/* 4825 */
{
    "_id" : "rainfall4824",
    "REGION" : "ORISSA",
    "YEAR" : 1951.0,
    "MONTH" : "JANUARY",
    "DATA" : 2.2
}

/* 4826 */
{
    "_id" : "rainfall4825",
    "REGION" : "ORISSA",
    "YEAR" : 1951.0,
    "MONTH" : "FEBRUARY",
    "DATA" : 5.1
}

/* 4827 */
{
    "_id" : "rainfall4826",
    "REGION" : "ORISSA",
    "YEAR" : 1951.0,
    "MONTH" : "MARCH",
    "DATA" : 89.7
}
```

Part2: predict future data
To predict Future data we choose 3.
Also, user search in 2-way.

```
3. Predict Future Data with Both Approach

        Your choice:    3
---------going for fetch mongoSave data-----------

1. expect by Region & Year
2. expect by Region, Year & Month

        Enter your choice:
```

Actually in our data-base there is not state-wise data, there is data according of regions of India based on geographical structure states portion. But problem is how to know all regions name in India. To fixed it in code I classify which region is part of which state, that's are lies in '**indianStates.json**' file**.** Also this file help to searching states in short form/using its first 3-4 letters. Like as:

```
"KARNATAKA": {
    "COASTAL": "COASTAL KARNATAKA",
    "NORTH": "NORTH INTERIOR KARNATAKA",
    "SOUTH": "SOUTH INTERIOR KARNATAKA"
},
```

```
"KAR": {
    "COASTAL": "COASTAL KARNATAKA",
    "NORTH": "NORTH INTERIOR KARNATAKA",
    "SOUTH": "SOUTH INTERIOR KARNATAKA"
},
```

```
1. expect by Region & Year
2. expect by Region, Year & Month

        Enter your choice: 2
Searching State: kar
Choose State: KARNATAKA
Searching Area: {
        'COASTAL': 'COASTAL KARNATAKA',
        'NORTH': 'NORTH INTERIOR KARNATAKA',
        'SOUTH': 'SOUTH INTERIOR KARNATAKA'
    }
Searching State's Area: north
Choose Area: NORTH INTERIOR KARNATAKA
Searching Month: _
```

So, when we choose a state name then show me which region you what to choose in this state.
Similarly, '**months.json**' help us to search month in short name or numeric value.

First, take about query: `expect by Region, Year & Month`

To predict `Searching Query: [ORISSA, 2020, FEBRUARY]` in approach-1
First predict [Orissa, 2018, February] using 1950-2017 Orissa, February data
Next predict [Orissa, 2019, February] using 1950-2018 Orissa, February data
At least predict [Orissa, 2020, February] using 1950-2019 Orissa, February data

```
['ORISSA', 2018.0, 'FEBRUARY', 20.06]
['ORISSA', 2019.0, 'FEBRUARY', 10.69]
['ORISSA', 2020.0, 'FEBRUARY', 9.85]
```

To predict `Searching Query: [ORISSA, 2020, FEBRUARY]` in approach-2
First predict [Orissa, 2018, January] using 1951-2017 Orissa, January & its previous 5 months data. Second predict [Orissa, 2018, February] using 1951-2017 Orissa, February & its previous 5 months data and 2018 January and its previous 4 months data. Third predict [Orissa, 2018, March] using 1951-2017 Orissa, February & its previous 5 months data and 2018 February & its previous 4 months data and continue… after complete the 2018 full year prediction I go result:

```
[['ORISSA', 2018.0, 'JANUARY', 11.48], ['ORISSA', 2018.0, 'FEBRUARY', 14.88], ['ORISSA', 2018.0, 'MARCH',
 19.97], ['ORISSA', 2018.0, 'APRIL', 45.87], ['ORISSA', 2018.0, 'MAY', 101.34], ['ORISSA', 2018.0, 'JUNE'
, 56.08], ['ORISSA', 2018.0, 'JULY', 558.39], ['ORISSA', 2018.0, 'AUGUST', 83.59], ['ORISSA', 2018.0, 'SE
PTEMBER', 25.38], ['ORISSA', 2018.0, 'OCTOBER', 44.98], ['ORISSA', 2018.0, 'NOVEMBER', 22.46], ['ORISSA',
 2018.0, 'DECEMBER', 4.88]]
```

Similarly, predict 2019 full year prediction using 1950-2018 data and get result:

```
[['ORISSA', 2019.0, 'JANUARY', 11.0], ['ORISSA', 2019.0, 'FEBRUARY', 15.11], ['ORISSA', 2019.0, 'MARCH',
20.98], ['ORISSA', 2019.0, 'APRIL', 14.63], ['ORISSA', 2019.0, 'MAY', 134.64], ['ORISSA', 2019.0, 'JUNE',
 395.76], ['ORISSA', 2019.0, 'JULY', 208.98], ['ORISSA', 2019.0, 'AUGUST', 692.27], ['ORISSA', 2019.0, 'S
EPTEMBER', 376.49], ['ORISSA', 2019.0, 'OCTOBER', 183.79], ['ORISSA', 2019.0, 'NOVEMBER', 22.11], ['ORISS
A', 2019.0, 'DECEMBER', 4.93]]
```

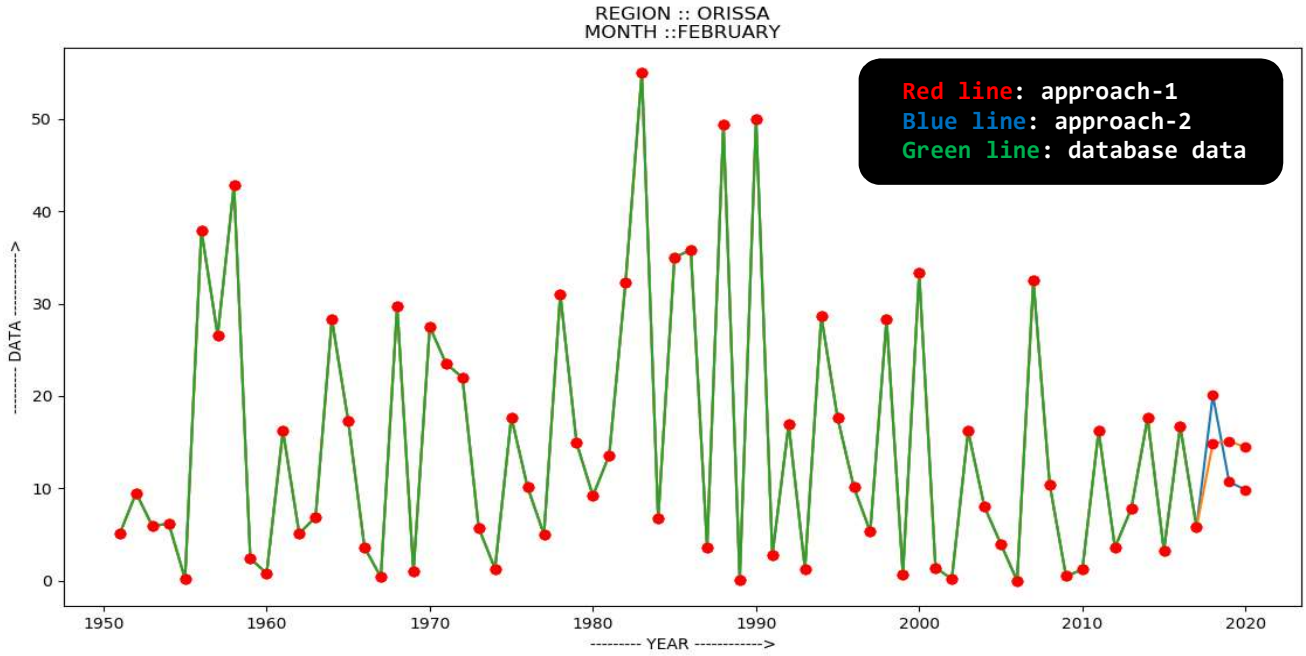
As query is to find [Orissa, 2020, February] result. Therefore, we predict 2020 January & February (include of January 2020 data) result and get result:

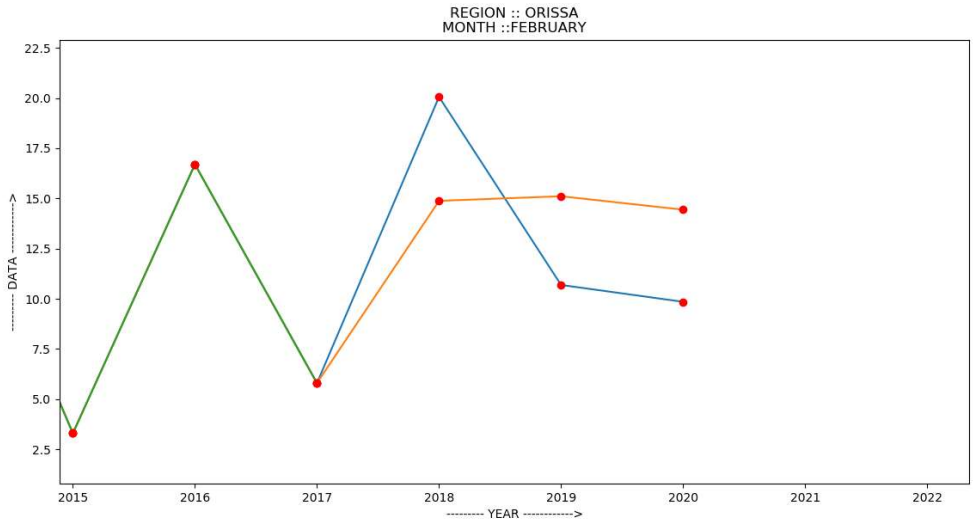
```
[['ORISSA', 2020.0, 'JANUARY', 10.92], ['ORISSA', 2020.0, 'FEBRUARY', 14.44]]
```

If we see about all prediction year February data then we see:

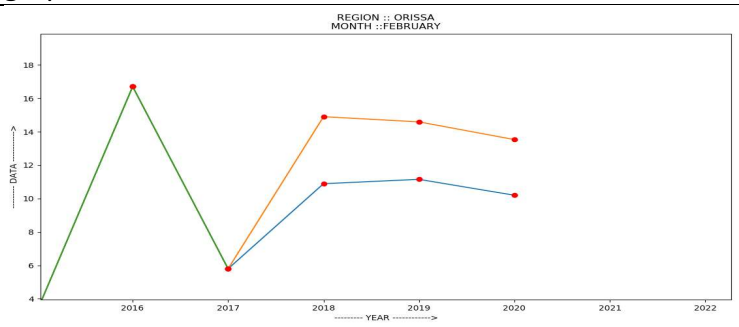
```
['ORISSA', 2018.0, 'FEBRUARY', 14.88]
['ORISSA', 2019.0, 'FEBRUARY', 15.11]
['ORISSA', 2020.0, 'FEBRUARY', 14.44]
```

We draw the graph using Orissa, 1950-2020 February data and see:





If predict again then see:

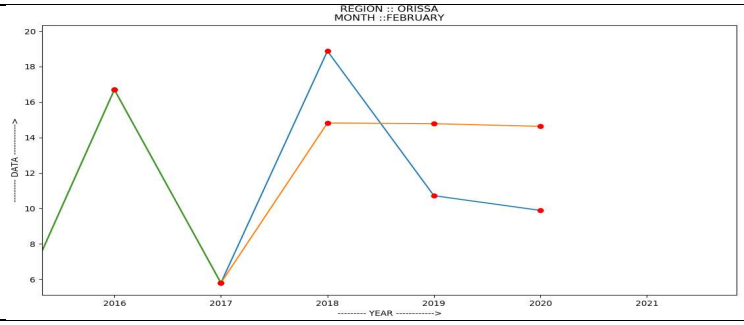
Prediction data	graph
Approach-1 ['ORISSA', 2018.0, 'FEBRUARY', 10.89] ['ORISSA', 2019.0, 'FEBRUARY', 11.15] ['ORISSA', 2020.0, 'FEBRUARY', 10.2]	
Approach-2 ['ORISSA', 2018.0, 'FEBRUARY', 14.9] ['ORISSA', 2019.0, 'FEBRUARY', 14.59] ['ORISSA', 2020.0, 'FEBRUARY', 13.53]	

Approach-1

```
['ORISSA', 2018.0, 'FEBRUARY', 18.88]
['ORISSA', 2019.0, 'FEBRUARY', 10.72]
['ORISSA', 2020.0, 'FEBRUARY', 9.89]
```

Approach-2

```
['ORISSA', 2018.0, 'FEBRUARY', 14.82]
['ORISSA', 2019.0, 'FEBRUARY', 14.78]
['ORISSA', 2020.0, 'FEBRUARY', 14.63]
```



So, it is clear that between two approaches approach-2 give us a best result.

Second, take about query: `expect by Region & Year`

To predict `Searching Query: [WEST UTTAR PRADESH, 2019]` in approach-1
Where 'West Uttar Pradesh' is a area/region of state Uttar Pradesh
First predict [West Uttar Pradesh, 2018, January] using 1950-2017 West Uttar Pradesh January data
Next predict [West Uttar Pradesh, 2019, January] using 1950-2018 West U.P. January data and complete January prediction.
Next, apply same procedure for February, March, April. . . December
Therefore, we got result:

```
[['WEST UTTAR PRADESH', 2018.0, 'JANUARY', 15.48], ['WEST UTTAR PRADESH', 2019.0, 'JANUARY', 15.31]]
[['WEST UTTAR PRADESH', 2018.0, 'FEBRUARY', 15.53], ['WEST UTTAR PRADESH', 2019.0, 'FEBRUARY', 15.43]]
[['WEST UTTAR PRADESH', 2018.0, 'MARCH', 11.55], ['WEST UTTAR PRADESH', 2019.0, 'MARCH', 11.63]]
[['WEST UTTAR PRADESH', 2018.0, 'APRIL', 5.76], ['WEST UTTAR PRADESH', 2019.0, 'APRIL', 5.75]]
[['WEST UTTAR PRADESH', 2018.0, 'MAY', 14.09], ['WEST UTTAR PRADESH', 2019.0, 'MAY', 14.11]]
[['WEST UTTAR PRADESH', 2018.0, 'JUNE', 75.78], ['WEST UTTAR PRADESH', 2019.0, 'JUNE', 76.08]]
[['WEST UTTAR PRADESH', 2018.0, 'JULY', 243.08], ['WEST UTTAR PRADESH', 2019.0, 'JULY', 244.22]]
[['WEST UTTAR PRADESH', 2018.0, 'AUGUST', 249.21], ['WEST UTTAR PRADESH', 2019.0, 'AUGUST', 249.89]]
[['WEST UTTAR PRADESH', 2018.0, 'SEPTEMBER', 139.71], ['WEST UTTAR PRADESH', 2019.0, 'SEPTEMBER', 139.83]]
[['WEST UTTAR PRADESH', 2018.0, 'OCTOBER', 28.94], ['WEST UTTAR PRADESH', 2019.0, 'OCTOBER', 28.96]]
[['WEST UTTAR PRADESH', 2018.0, 'NOVEMBER', 3.35], ['WEST UTTAR PRADESH', 2019.0, 'NOVEMBER', 3.36]]
[['WEST UTTAR PRADESH', 2018.0, 'DECEMBER', 6.58], ['WEST UTTAR PRADESH', 2019.0, 'DECEMBER', 6.56]]
```

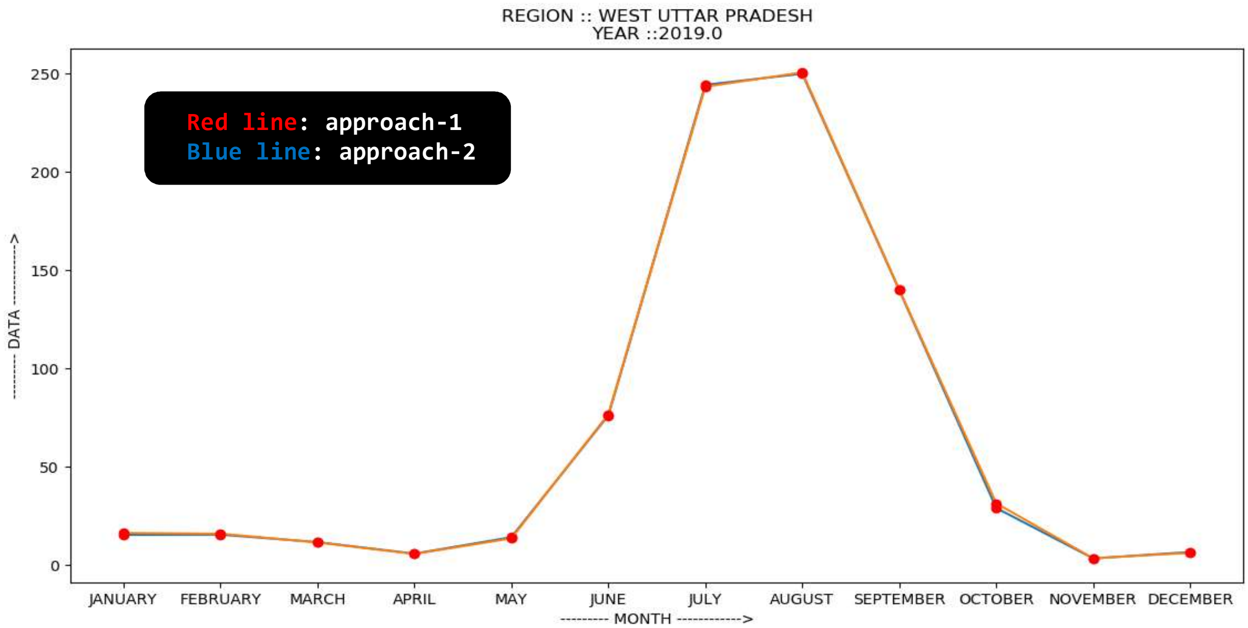
To predict `Searching Query: [WEST UTTAR PRADESH, 2019]` in approach-2
Where 'West Uttar Pradesh' is a area/region of state Uttar Pradesh
First predict [West Uttar Pradesh, 2018, January] using 1950-2017 West Uttar Pradesh data
Next predict [West Uttar Pradesh, 2018, February] using 1950-2017 and January 2018 data and continue…
Therefore we got result of 2018:

```
[['WEST UTTAR PRADESH', 2018.0, 'JANUARY', 16.2], ['WEST UTTAR PRADESH', 2018.0, 'FEBRUARY', 15.86], [
'WEST UTTAR PRADESH', 2018.0, 'MARCH', 11.48], ['WEST UTTAR PRADESH', 2018.0, 'APRIL', 5.59], ['WEST U
TTAR PRADESH', 2018.0, 'MAY', 13.51], ['WEST UTTAR PRADESH', 2018.0, 'JUNE', 76.62], ['WEST UTTAR PRAD
ESH', 2018.0, 'JULY', 243.27], ['WEST UTTAR PRADESH', 2018.0, 'AUGUST', 250.56], ['WEST UTTAR PRADESH'
, 2018.0, 'SEPTEMBER', 139.93], ['WEST UTTAR PRADESH', 2018.0, 'OCTOBER', 31.34], ['WEST UTTAR PRADESH
', 2018.0, 'NOVEMBER', 3.39], ['WEST UTTAR PRADESH', 2018.0, 'DECEMBER', 6.18]]
```

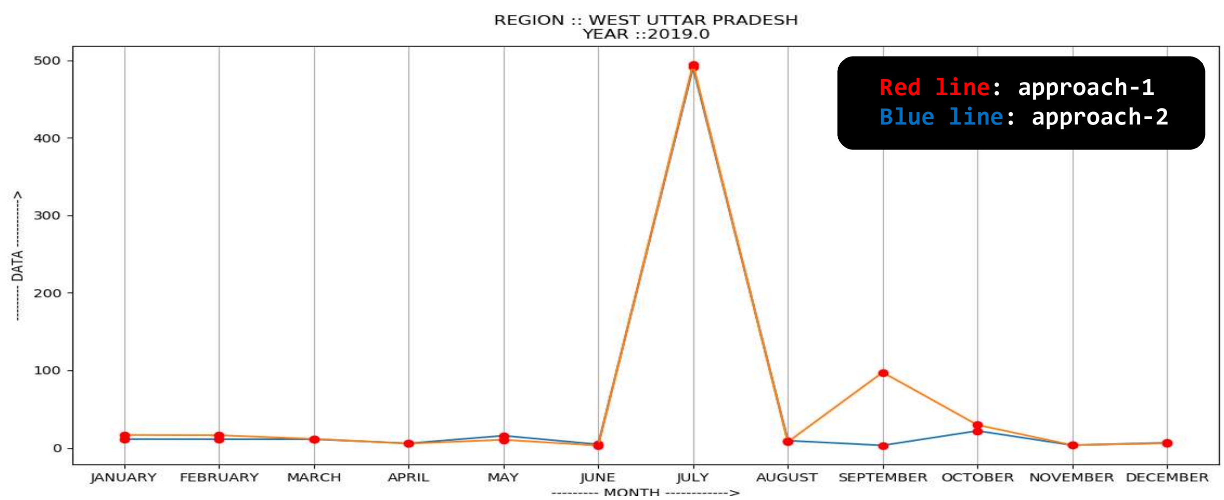
After complete 2018 prediction. Using of 1950-2018 result in this approach we predict 2019 result.
So, 2019 prediction result of West U.P.:

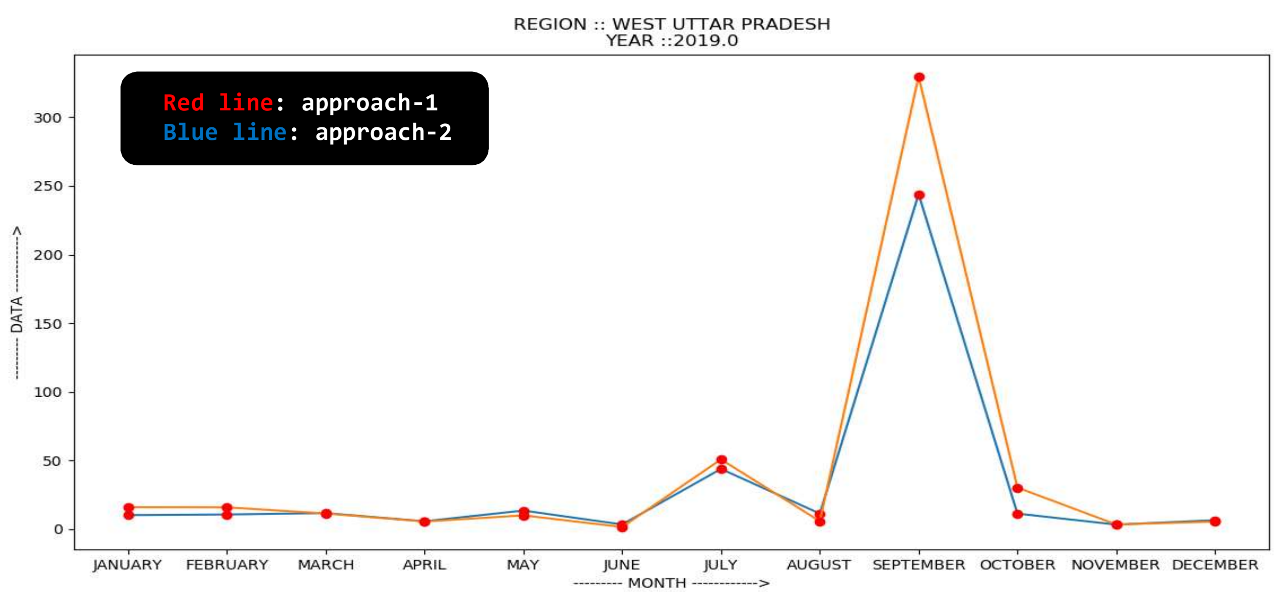
```
[['WEST UTTAR PRADESH', 2019.0, 'JANUARY', 16.27], ['WEST UTTAR PRADESH', 2019.0, 'FEBRUARY', 15.91],
['WEST UTTAR PRADESH', 2019.0, 'MARCH', 11.45], ['WEST UTTAR PRADESH', 2019.0, 'APRIL', 5.59], ['WEST
UTTAR PRADESH', 2019.0, 'MAY', 13.52], ['WEST UTTAR PRADESH', 2019.0, 'JUNE', 76.62], ['WEST UTTAR PRA
DESH', 2019.0, 'JULY', 243.26], ['WEST UTTAR PRADESH', 2019.0, 'AUGUST', 250.56], ['WEST UTTAR PRADESH
', 2019.0, 'SEPTEMBER', 139.93], ['WEST UTTAR PRADESH', 2019.0, 'OCTOBER', 31.34], ['WEST UTTAR PRADES
H', 2019.0, 'NOVEMBER', 3.39], ['WEST UTTAR PRADESH', 2019.0, 'DECEMBER', 6.18]]
```

We draw the graph of West U.P., 2019 prdiction result of both approach:



If we predict again we see results:





Conclusion and Future Work:

Rainfall prediction is a beneficial but challenging task. Data mining techniques have the ability to predict the rainfall by extracting and using the hidden knowledge from past weather data. In the last decade, many researchers have worked to increase the accuracy of rainfall prediction by optimizing and integrating data mining techniques. Various models and techniques are available today for effective rainfall prediction but still there was a lack of a compact literature review and systematic mapping study, which could reflect the current problems, proposed solutions and the latest trends in this domain.

I am here using ANN (Artificial Neural Network) to predict the future data. There are another option to predict the rainfall data are SVM (Support Vector Machine), RNN (Recurrent Neural Networks), ANFIS (Adaptive Neuro Fuzzy Inference System) etc. and after that if we comparison all of things then we get a better solution to predict Rainfall data.

I start to project with the goal of weather prediction. But, with only rainfall prediction we did not predict the weather data. Therefore, if we add another criterion of the weather data prediction, like temperature, humidity, air-presser etc. then we predict a perfect weather condition.

References:

[01]. Administrator, N. C. (2017, august 07). *NASA - What's the Difference Between Weather and Climate?* Retrieved from nasa.gov: https://www.nasa.gov/mission_pages/noaa-n/climate/climate_weather.html

[02]. (2003, april 26). *Introduction to Weather*. Retrieved from ecoca.ro: http://www.ecoca.ro/meteo/tutorial/Weather/Younger/Weather_Introduction.html

[03]. (2003, april 26). *Introduction to Climate*. Retrieved from ecoca.ro: http://www.ecoca.ro/meteo/tutorial/Climate/Older/Climate_Introduction.html

[04]. (2003, april 26). *Average Weather*. Retrieved from ecoca.ro: http://www.ecoca.ro/meteo/tutorial/Climate/Older/Average_Weather.html

[05]. L S Rathore, S. D. (2013). *STATE LEVEL CLIMATE CHANGE TRENDS IN INDIA.* Lodi Road, New Delhi- 3 (India): India Meteorological Department. Retrieved from http://www.imd.gov.in/pages/services_climate.php?adta=PDF&adtc=../section/climate/StateLevelClimateChangeMonoFinal

[06]. Chakraborty, B. (2008, August). ANALYSIS OF RAINFALL DATA SILCHAR, ASSAM. *ANALYSIS OF RAINFALL DATA SILCHAR, ASSAM*, p. 30.

[07]. Rong-Gang Cong, M. B. (2012, August 17). Copula Analyses. (G. O. Varotsos, Ed.) *The Interdependence between Rainfall and Temperature*, 11. Retrieved from http://dx.doi.org/10.1100/2012/405675

[08]. (2019, April). *Geography of India*. Retrieved from wikipedia.org: https://en.wikipedia.org/wiki/Geography_of_India

[09]. (n.d.). Retrieved from india-wris.nrsc.gov.in: http://www.india-wris.nrsc.gov.in/wrpinfo/index.php?title=Basins

[10]. (2019, march). *Drought in India*. Retrieved from wikipedia.org: https://en.wikipedia.org/wiki/Drought_in_India

[11]. (2019). *Floods_in_India*. Retrieved from wikipedia.org: https://en.wikipedia.org/wiki/Floods_in_India

[12]. & ISSAC, L. P. (2014, January). *SQL vs NoSQL Database Differences Explained with few Example DB.* Retrieved from thegeekstuff.com: https://www.thegeekstuff.com/2014/01/sql-vs-nosql-db

[13]. (2019). *10 Common SQL Programming Mistakes and How to Avoid Them.* Retrieved from upwork.com: https://www.upwork.com/hiring/data/common-sql-programming-mistakes/

[14]. j. p. (n.d.). *Why you should not use SELECT * in SQL Query.* Retrieved from java67.com: http://www.java67.com/2018/02/why-you-should-not-use-select-in-sql.html

[15]. J. J. (2015, february). *Why and when should I use NoSQL instead of SQL?* Retrieved from quora.com: https://www.quora.com/Why-and-when-should-I-use-NoSQL-instead-of-SQL

[16]. (n.d.). *MongoDB and MySQL Compared*. Retrieved from .mongodb.com: https://www.mongodb.com/compare/mongodb-mysql

[17]. M. K. (2014, may). *When to Use MongoDB Rather than MySQL (or Other RDBMS): The Billing Example*. Retrieved from dzone.com: https://dzone.com/articles/when-use-mongodb-rather-mysql

[18]. P. T. (2007). *Sigmoid Functions and Their Usage in.* Bahçeşehir University,, Faculty of Engineering and Natural Sciences, Computer Engineering Department. Beşiktaş, İstanbul / TÜRKİYE: -. Retrieved from excel.ucf.edu: https://excel.ucf.edu/classes/2007/Spring/appsII/Chapter1.pdf

[19]. H. Z. (2018, january 22). *How It Improves Performance in Deep Learning.* Retrieved from towardsdatascience.com: https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10

[20]. (2018). *15-worst-ever-floods-in-india-over-last-decade.* Retrieved from walkthroughindia.com: http://www.walkthroughindia.com/walkthroughs/15-worst-ever-floods-in-india-over-last-decade

[21]. P. R. (2004). *PREDICTION OF MONSOON RAINFALL AND RIVER DISCHARGE ON 15-30-DAY TIME SCALES .* Atlanta , Georgi a : American Meteorological Society. Retrieved from https://journals.ametsoc.org/doi/pdf/10.1175/BAMS-85-11-1745

[22]. (2018, December). *From drought to flood - how climate change-fuelled.* Retrieved from indiaspend.com: https://www.indiaspend.com/wp-content/uploads/2018/12/Climate_impacts_hurting_poor.pdf

[23]. V. R. (2017, March). *Avoid Utility Classes*. Retrieved from vojtechruzicka.com: https://www.vojtechruzicka.com/avoid-utility-classes/

[24]. (2019, May). *Configuration file*. Retrieved from wikipedia.org: https://en.wikipedia.org/wiki/Configuration_file

[25]. (n.d.). *xlrd*. Retrieved from directory.fsf.org: https://directory.fsf.org/wiki/Xlrd, http://www.python-excel.org/

[26]. (n.d.). *PyMongo 3.8.0 Documentation*. Retrieved from mongodb.com: https://api.mongodb.com/python/current/
[27]. (2019). *Python NumPy Tutorial: Learn with Example*. Retrieved from guru99.com: https://www.guru99.com/numpy-tutorial.html
[28]. F. T. (2017). PREDICTION OF RAINFALL USING DATA MINING TECHNIQUES. 38. Retrieved from http://dspace.bracu.ac.bd/xmlui/bitstream/handle/10361/9489/12201069%2C17141001%2C13110015_CSE.pdf?sequence=1&isAllowed=y
[29]. S. Z. (2016). Short-term water level predictionusing different artificial intelligent models. Retrieved from https://ieeexplore.ieee.org/document/7577678/authors#authors
[30]. O. T. (2012). *Monthly Rainfall Estimation Using Data-Mining.* New York, United State: Hindawi Publishing Corporation. Retrieved from https://dl.acm.org/citation.cfm?id=2433946
[31]. A. J. (n.d.). *Multi-Layer Perceptron: Artificial Nural Network.* Retrieved from cse.unsw.edu.au: http://www.cse.unsw.edu.au/~cs9417ml/MLP2/
[32]. J. R. (2019, April 30). Equalizing Seasonal Time Series Using Artificial Neural Networks in Predicting the Euro–Yuan Exchange Rate. 76. Retrieved from mdpi.com: https://doi.org/10.3390/jrfm12020076
[33]. L. L. (2019). *Working With JSON Data in Python.* Retrieved from realpython.com: https://realpython.com/python-json
[34]. M. R. (2016, December). *supervised learning.* Retrieved from techtarget.com: https://searchenterpriseai.techtarget.com/definition/supervised-learning
[35]. S. S. (2017, August). *Artificial Neural Network (ANN) in Machine Learning.* Retrieved from datasciencecentral.com: https://www.datasciencecentral.com/profiles/blogs/artificial-neural-network-ann-in-machine-learning