A

MINOR PROJECT REPORT

ON

**RAINFALL PREDICTION**

MACHINE LEARNING PROJECT FOR EMOTION DETECTION

SUBMITTED TOWARDS PARTIAL FULFILMENT OF THE ACADEMIC

REQUIREMENT FOR THE AWARD OF DEGREE OF

**MASTER OF COMPUTER APPLICATION**

**Under the esteemed guidance of**

**Dr. Jibitesh Mishra**

**Associate Professor**

**Department of CSA**



**Submitted**

**By**

**Jagdish Sahoo (1905106010)**

**DEPARTMENT OF COMPUTER SCIENCE & APPLICATION**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Techno Campus ,PO- Ghatikia, Mahalaxmi Vihar, Bhubaneswar-751029)**

**2020-2021**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Techno Campus, PO- Ghatikia, Mahalaxmi Vihar,Bhubaneswar-751029)**

**DEPARTMENT OF COMPUTER SCIENCE & APPLICATION**



**CERTIFICATE**

This is to certify that the minor project entitled **RAINFALL PREDICTION** is being submitted by **Jagdish Sahoo** with reg.no **1905106010** in partial fulfillment of the requirement for the award “Master of Computer Application” degree is an authentic record of the work done by him under my supervision and guidance. The matter embodied in the project has not been submitted to any other University for the award of any degree or diploma to the best of my knowledge. The work carried out by him during the project period is original and performance during the compilation of project was appreciable.

**HOD & INTERNAL GUIDE**

**Dr. Jibitesh Mishra**

**(Associate Professor)**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Techno Campus, PO- Ghatikia, Mahalaxmi Vihar,Bhubaneswar-751029)**

**DEPARTMENT OF COMPUTER SCIENCE & APPLICATION**



**DECLARATION**

We hereby declare that the Minor project work entitled “**RAINFALL PREDICTION**“ submitted to the College of Engineering and Technology, Bhubaneswar is a record of original work done by us under the guidance of

**Dr. Jibitesh Mishra**, Associate professor of Computer Science and Application and this project work is submitted in the partial fulfillment of the award of the degree of Master of Computer Applications. The results embodied in this report have not been submitted to any other University or Institutes for the award of any degree.

**Date: Student Name**

**Jagdish Sahoo**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Techno Campus, PO- Ghatikia, Mahalaxmi Vihar,Bhubaneswar-751029)**

**DEPARTMENT OF COMPUTER SCIENCE & APPLICATION**



**ACKNOWLEDGMENT**

What we write or mention in this sheet will hardly be adequate in return for the amount of help and cooperation we have received from all the people who have contributed to make this project a reality. This project owes its existence to the help, support and inspiration of these people.

We would like to express our sincere gratitude to our adviser, **Dr. Jibitesh Mishra**, Associate Professor, Computer Science & Engineering Department, whose knowledge and guidance has motivated us to achieve goals we never thought possible. He has consistently been a source of motivation, encouragement, and inspiration. The time we have spent working under her supervision has truly been a pleasure.

We would also like to convey our deep regards to all other faculty members of Dept. of CSA, who have bestowed their great effort and guidance at appropriate times without which it would have been very difficult on our part to finish this work. Finally, we would also like to thank our friends for their advice and pointing out our mistakes.

**Date: Jagdish Sahoo**

**Place: CET,BBSR (1905106010)**

**ABSTRACT**

Rainfall prediction is important as heavy rainfall can lead to many disasters. The prediction helps people to take preventive measures and moreover the prediction should be accurate. There are two types of prediction short term rainfall prediction and long term rainfall. Prediction mostly short term prediction can gives us the accurate result. The main challenge is to build a model for long term rainfall prediction. Heavy precipitation prediction could be a major drawback for earth science department because it is closely associated with the economy and lifetime of human. It’s a cause for natural disasters like flood and drought that square measure encountered by individuals across the world each year.

Accuracy of rainfall statement has nice importance for countries like India whose economy is basically dependent on agriculture. The dynamic nature of atmosphere, applied mathematics techniques fail to provide sensible accuracy for precipitation statement. The prediction of precipitation using machine learning techniques may use regression. Intention of this project is to offer non-experts easy access to the techniques, approaches utilized in the sector of precipitation prediction and provide a comparative study among the various machine learning techniques.

**Table of Contents**

1. Introduction…………………………………………………..…1
2. Software and Hardware Requirements…………………….……9
   1. Software Requirements
   2. Hardware Requirements
3. Literature Survey…………………………………………...….11
4. Software Requirement Analysis………………………….........14
   1. Problem Statement
   2. Proposed Method For Solution
5. Software Design……………………………………………….17

5.1 Data Flow Diagram

1. Code Templates and Results…………………………………..19
2. Observation & Conclusion………………………………….....52
3. Reference / Bibliography…………………………………........54

**Chapter 1**

**INTRODUCTION**

**1 Introduction**

Rainfall forecasting is very important because heavy and irregular rainfall can have many impacts like destruction of crops and farms, damage of property so a better forecasting model is essential for an early warning that can minimize risks to life and property and also managing the agricultural farms in better way. This prediction mainly helps farmers and also water resources can be utilized efficiently. Rainfall prediction is a challenging task and the results should be accurate. There are many hardware devices for predicting rainfall by using the weather conditions like temperature, humidity, pressure. These traditional methods cannot work in an efficient way so by using machine learning techniques we can produce accurate results. We can just do it by having the historical data analysis of rainfall and can predict the rainfall for future seasons.

Two types of rainfall predictions can be done; they are - Long term predictions: Predict rainfall over few weeks/months in advance. Short term predictions: Predict rainfall a few days in advance in specific locations. Indian meteorological department provides forecasting data required for project. In this project we are planning to work on long term predictions of rainfall. The main motive of the project is to predict the amount of rainfall in a particular division or state well in advance. We can apply many techniques like classification, regression according to the requirements and also we can calculate the error between the actual and prediction and also the accuracy. Different techniques produce different accuracies so it is important to choose the right algorithm and model it according to the requirements.

**Regression analysis:**

Regression analysis deals with the dependence of one variable (called as dependent variable) on one or more other variables, (called as independent variables) which is useful for estimating and/ or predicting the mean or average value of the former in terms of known or fixed values of the latter. For example, the salary of a person is based on his/her experience here, the experience attribute is independent variable salary is dependent variable. Simple linear regression defines the relationship between a single dependent variable and a single independent variable. The below equation is the general form of regression.

y = β0 + β1x + ε where β0 and β1 are parameters, and ε is a probabilistic error term. Regression analysis is a vital tool for modeling and analyzing information. It is used for predictive analysis that is forecasting of rainfall or weather, predicting trends in business, finance, and marketing. It can also be used for correcting errors and also provide quantitative support.

**The advantages of regression analysis are:**

1. It is a powerful technique for testing relationship between one dependent variable and many independent variables.

2. It allows researchers to control extraneous factors.

3. Regression asses the cumulative effect of multiple factors.

4. It also helps to attain the measure of error using the regression line as a base for estimations.

**Chapter 2**

**SOFTWARE & HARDWARE REQUIREMENTS**

# 2 Software & Hardware Requirement

## 2.1 Software Requirement

### 2.1.1 Anaconda

2.1.2 Jupyter NoteBook

2.1.3 Python

2.1.4 Python Libraries

2.1.4.1 Pandas

2.1.4.2 Seaborn

2.1.4.3 Keras

2.1.4.4 SciPy

2.1.4.5 SkLearn

2.1.4.6 NumPy

**2.2 Hardware Requirement**

2.2.1 Windows 10 – 64 Bit

2.2.2 AMD Ryzen 5 Gen 3rd

2.2.3 RAM 8 GB

**Chapter 3**

**LITERATURE SURVEY**

# 3. Literature Survey

Thirumalai, Chandrasegar, et al. [1] discusses the amount of rainfall in past years according to the crop seasons and predicts the rainfall for future years. The crop seasons are Rabi, Kharif and Zaid. Linear regression method is applied for early prediction. Here, Rabi and kharif were taken as variables if one variable was given then other can be predicted using linear regression. Standard deviation and Mean was also calculated for future prediction of crop seasons. This implementation will be used for farmers to have an idea of which crop to harvest according to crop seasons. Geetha, A., and G. M. Nasira. [2] implements a model which predicts the weather conditions like rainfall, fog, thunderstorms and cyclones which will be helpful to the people to take preventive measures.

Data mining techniques were used and a data mining tool named Rapid miner was used to model the decision trees. The data set of Trivandrum with attributes like day, temperature, dew point, pressure etc. The dataset is divided into training set and testing set and decision tree algorithm is applied. The accuracy is calculated, actual and predicted values are compared. The accuracy is 80.67 and to achieve high value it can be extended by applying soft computing techniques like fuzzy logic and genetic algorithms. Parmar, Aakash, Kinjal Mistree, and Mithila Sompura [3] discusses the different methods used for rainfall prediction for weather forecasting with their limitations. Various neural networks algorithm which are used for prediction are discussed with their steps in detail categorizes various approaches and algorithms used for rainfall prediction by various researchers in today’s era. Finally, presents conclusion of paper. Done the background work about some models of machine learning ARIMA Model, Artificial neural network and types like Back- Propagation Neural Network - Cascade Forward Back Propagation Network Layer Recurrent Network, Self-Organizing Map and Support Vector Machine, Collected, surveyed and table presents categorization of different approaches of rainfall prediction.

Dash, Yajnaseni, Saroj K. Mishra, and Bijaya K. Panigrahi [4] has used artificial intelligence techniques like Artificial Neural Network (ANN), Extreme Learning Machine (ELM), K nearest neighbor (KNN) are applied for prediction of summer monsoon and post monsoon rainfall. The dataset used is the time series data of Kerala from 1871 to 2016 taken from Indian Institute of Tropical Meteorology (IITM).The data is pre-processed and normalization was performed on the data next, the data is divided into training and testing the data up to 2010 was taken as training set and the data from 2011- 2016 taken as test set. The above mentioned algorithms were applied and its performance was calculated by using MAE, RMSE, and MASE. The ELM algorithm has given accurate results compared to the others. Singh, Gurpreet, and Deepak Kumar[5] states that there are many machine learning algorithms applied for the prediction of rainfall and in this, they have used a hybrid approach that is combining two techniques, Random forest and Gradient boosting with many machine learning techniques like ada boost, K-Nearest Neighbor(KNN), Support vector machine(SVM), and Neural Network(NN).These have been applied on the rainfall data of North Carolina from 2007 – 2017 and also the performance is calculated by applying different metrics F-score, precision, accuracy, recall.

Finally, eight hybrid models have been proposed and Gradient boosting-Ada boost has been the superior which exhibited good results. Kar, Kaveri, Neelima Thakur, and Prerika Sanghvi [6] has used the fuzzy logic approach for the prediction of rainfall on the data of temperature in a geographic location. The fuzzy model has been applied Due to other climatic factors the prediction is not accurate so they have considered other influencing factors like humidity also analyzed the advantages of fuzzy system over other techniques. Sardeshpande, Kaushik D., and Vijaya R. Thool [7] has used the artificial neural networks, back propagation (BPNN), radial basis function (RBFNN) and generalized regression (GRNN) on the rainfall data of India mainly Nanded district, Maharashtra was considered and the data is normalized between 0 to 1 and the algorithms are applied and the performance of those was calculated and compared. BPNN and RBFNN has given good results compared to GRNN. Chen, Binghong, et al. [8] focuses on the non-linear machine learning approaches like gradient boosting decision tree model and deep neural networks for a short term prediction of rainfall and these algorithms were built on Alibaba cloud and data was collected from different sites and effectiveness is calculated by using classification metrics AUC, F1 score, precision and accuracy and by Regression metric RMSE, correlation. It has been observed that DNN showed better result than ECData. Moon, Seung-Hyun, et al [9] implements an early warning system (EWS) that produces a signal when it reaches a threshold limit that givesWarning before 3 hrs. This was done by using machine learning techniques. South Korea data from 2007 to 2012 was taken and performance is measured by some criteria and a confusion matrix was produced. The logistic regression with feature selection and PCA was proposed. F-measure is calculated for estimating the efficiency of model.

**Chapter 4**

**SOFTWARE REQUIREMENT ANALYSIS**

**4 Software Requirement Analysis**

**4.1 Problem Statement**

Rainfall prediction remains a serious concern and has attracted the attention of governments, industries, risk management entities, as well as the scientific community. Rainfall is a climatic factor that affects many human activities like agricultural production, construction, power generation, forestry and tourism, among others. As we know heavy and irregular rainfall can have many impacts like destruction of crops and farms, damage of property so a better forecasting model is essential for an early warning that can minimize risks to life and property and also managing the agricultural farms in better way. Therefore, having an appropriate approach for rainfall prediction is also required to counter these problems. This project is made focussing these problems.

**4.2 Proposed Method for Solution**

The predictive model is used to prediction of the precipitation. The first step is converting data in to the correct format to conduct experiments then make a good analysis of data and observe variation in the patterns of rainfall. We predict the rainfall by separating the dataset into training set and testing set then we apply different machine learning approaches (MLR, SVR, etc.) and statistical techniques and compare and draw analysis over various approaches used. With the help of numerous approaches we attempt to minimize the error.

**Dataset Description:**

The dataset [10] consists of the measurement of rainfall from year 1901-2015 for each state.

• Data consists of 19 attributes (individual months, annual, and combinations of 3 consecutive months) for 36 sub divisions.

• The data is available only from 1950 to 2015 for some of the subdivisions

• The attributes are the amount of rainfall measured in mm

As the dataset is very large, feature reduction is done so that it improves the accuracy, reduces the computation time and also storage. Principal Component Analysis (PCA) is a technique of extracting necessary variables from a huge set of variables. It extracts low dimensional set with a motive to capture the maximum amount of information. With few variables, visualization becomes more significant. It is done by using covariance matrix and by obtaining Eigen values from it. In our dataset by using PCA it has reduced the attributes by considering only the rainfall data of combination of three consecutive months and annual data from every subdivision. Techniques used: Multiple Linear Regression: Multiple regression tries to model the connection between two or additional variables and a response by fitting an equation to determined information. Clearly, it's nothing however an extension of straight forward regression toward the mean. The general form of multivariable linear regression model is: y=α+β1x1+ β2x2+…+ βkxk+ε where y = dependent variable and x1, x2… xk are independent variables,α,β are coefficients. Multiple regression will model additional complicated relationship that comes from numerous options along they should to be employed in cases wherever one explicit variable isn't evident enough to map the link between the independent and also the variable quantity.

**Support Vector Regression:**

Support Vector regression machine learning and data science with the term SVM or support vector machine but SVR that is support vector regression is a bit different from SVM that is support vector machine as the name suggests that is integration algorithm so we can use SVR for working with continuous value instead of classification which is SVM Support Vector Machines support linear and nonlinear regression that we can refer to as Support Vector Regression. Instead of trying to fit the largest possible street between two classes while limiting margin violations, Support Vector Regression tries to fit as many instances as possible on the street while limiting margin violations. The size of the lane is measured by a hyper parameter Epsilon.

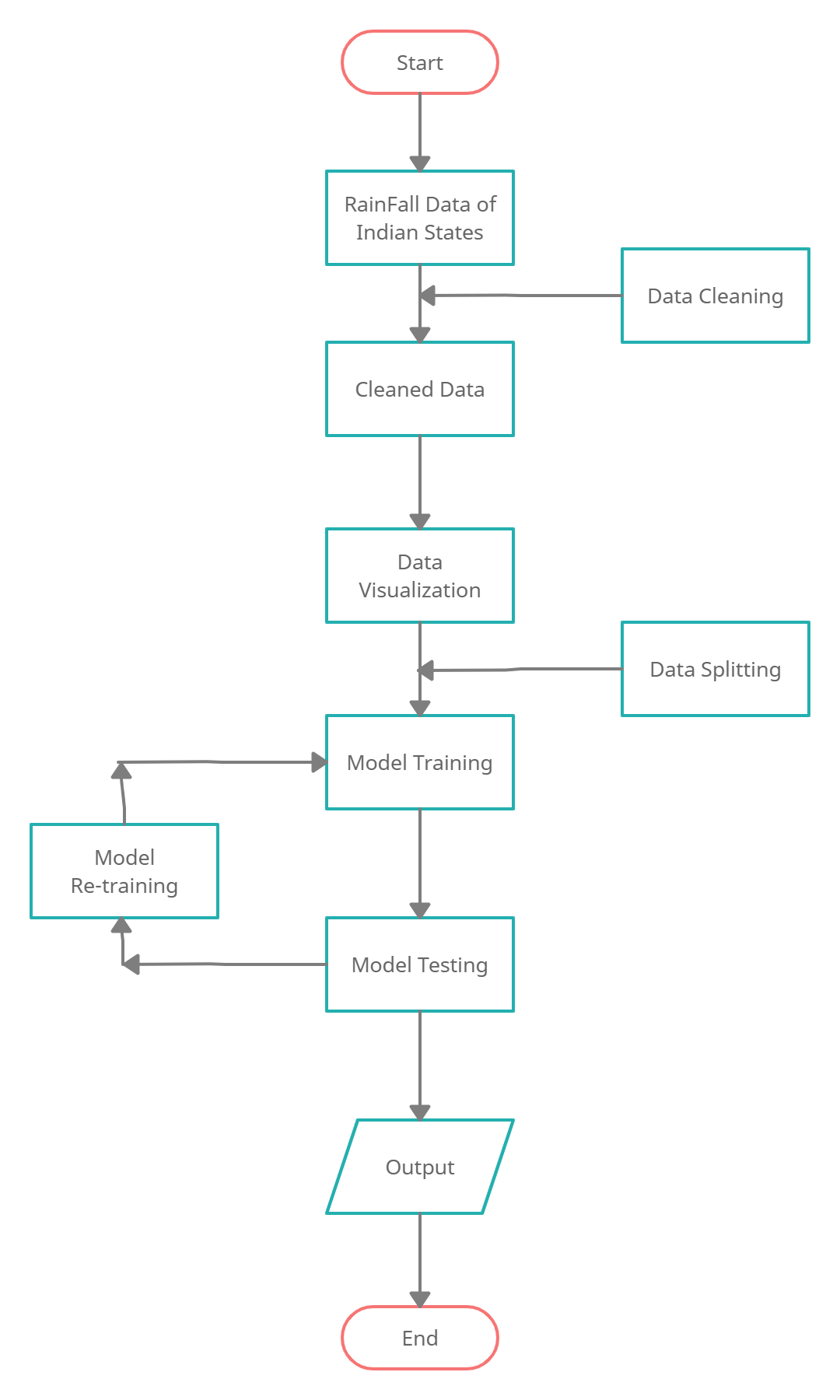
**Kernel- The function used to map a low dimensional data into higher dimensional data.**

Hyper plane- in SVM this is a basically The Separation line between the data classes also in SVR we are going to define it is as the line that will that will help us to predict the continuous value or target value. Boundary line - the SVM plane which creates imagine the support vector can be on boundary lines or outside the boundary line separates two classes in the concept same. Vectors-these are the data points which are closest to the boundary the distance of the point is minimum. SVR performs linear regression in higher dimensional space. We can think of SVR as if each data point in the training represents its own dimension. When we evaluate kernel between a test point and a point in the training set the resulting value gives you the coordinate of your test point in that dimension. The vector we get when we evaluate the test point for all points in the training set, k is the representation of the test point in the higher dimensional space. The equation of the hyper plane is wx+b=0 and the two equations of boundary lines is Wx+b=+e, Wx+b=-e Equation that satisfy our SVR is e<=y-Wx-b<=+e SVR has a different regression goal compared to linear regression in linear regression, we are trying to minimize the error between the prediction and data whereas in SVR a goal is to make sure that error do not exceed the threshold.

**Chapter 5**

**SOFTWARE DESIGN**

**5.1 Data Flow Diagram**



**Chapter 6**

**CODE TEMPLATES AND RESULTS**

**6 Code Templates and Outputs**

# Dataset

* + - Dataset1([dataset1](https://data.gov.in/resources/district-rainfall-normal-mm-monthly-seasonal-and-annual-data-period-1951-2000)) This dataset has average rainfall from 1951-2000 for each district, for every month.
    - Dataset2([dataset2](https://data.gov.in/resources/subdivision-wise-rainfall-and-its-departure-1901-2015)) This dataset has average rainfall for every year from 1901-2015 for each state.

# Methodology

* + - Converting data into the correct format to conduct experiments.
    - Make a good analysis of data and observe variation in the patterns of rainfall.
    - Finally, we try to predict the average rainfall by separating data into training and testing. We apply various statistical and machine learning approaches(*SVM*, etc) in prediction and make analysis over various approaches. By using various approaches we try to minimize the error.

In [1]: import numpy as np *# linear algebra*

import pandas as pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

import matplotlib.pyplot as plt import seaborn as sns

# Types of graphs

* + - Bar graphs showing distribution of amount of rainfall.
    - Distribution of amount of rainfall yearly, monthly, groups of months.
    - Distribution of rainfall in subdivisions, districts form each month, groups of months.
    - Heat maps showing correlation between amount of rainfall between months.

In [2]: data = pd.read\_csv("../data/rainfall\_in\_india\_1901-2015.csv",sep=",") data = data.fillna(data.mean())

data.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 4116 entries, 0 to 4115

|  |  |  |
| --- | --- | --- |
| Data columns | (total | 19 columns): |
| SUBDIVISION | 4116 | non-null object |
| YEAR | 4116 | non-null int64 |
| JAN | 4116 | non-null float64 |
| FEB | 4116 | non-null float64 |
| MAR | 4116 | non-null float64 |
| APR | 4116 | non-null float64 |
| MAY | 4116 | non-null float64 |
| JUN | 4116 | non-null float64 |
| JUL | 4116 | non-null float64 |
| AUG | 4116 | non-null float64 |
| SEP | 4116 | non-null float64 |
| OCT | 4116 | non-null float64 |
| NOV | 4116 | non-null float64 |
| DEC | 4116 | non-null float64 |
| ANNUAL | 4116 | non-null float64 |
| Jan-Feb | 4116 | non-null float64 |
| Mar-May | 4116 | non-null float64 |
| Jun-Sep | 4116 | non-null float64 |
| Oct-Dec | 4116 | non-null float64 |

dtypes: float64(17), int64(1), object(1)

memory usage: 611.0+ KB

# Dataset-1 Description

* + - Data has 36 sub divisions and 19 attributes (individual months, annual, combinations of 3 consecutive months).
    - For some of the subdivisions data is from 1950 to 2015.
    - All the attributes has the sum of amount of rainfall in mm.

In [3]: data.head()

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Out[3]: | SUBDIVISION | YEAR | JAN | FEB | | MAR | APR | MAY | JUN \ |
|  | 0 ANDAMAN & NICOBAR ISLANDS | 1901 | 49.2 | 87.1 | | 29.2 | 2.3 | 528.8 | 517.5 |
|  | 1 ANDAMAN & NICOBAR ISLANDS | 1902 | 0.0 | 159.8 | | 12.2 | 0.0 | 446.1 | 537.1 |
|  | 2 ANDAMAN & NICOBAR ISLANDS | 1903 | 12.7 | 144.0 | | 0.0 | 1.0 | 235.1 | 479.9 |
|  | 3 ANDAMAN & NICOBAR ISLANDS | 1904 | 9.4 | 14.7 | | 0.0 | 202.4 | 304.5 | 495.1 |
|  | 4 ANDAMAN & NICOBAR ISLANDS | 1905 | 1.3 | 0.0 | | 3.3 | 26.9 | 279.5 | 628.7 |
| JUL AUG SEP OCT | | NOV | DEC | | ANNUAL | | Jan-Feb | Mar-May \ | |
| 0 365.1 481.1 332.6 388.5 | | 558.2 | 33.6 | | 3373.2 | | 136.3 | 560.3 | |
| 1 228.9 753.7 666.2 197.2 | | 359.0 | 160.5 | | 3520.7 | | 159.8 | 458.3 | |
| 2 728.4 326.7 339.0 181.2 | | 284.4 | 225.0 | | 2957.4 | | 156.7 | 236.1 | |
| 3 502.0 160.1 820.4 222.2 | | 308.7 | 40.1 | | 3079.6 | | 24.1 | 506.9 | |
| 4 368.7 330.5 297.0 260.7 | | 25.4 | 344.7 | | 2566.7 | | 1.3 | 309.7 | |
| Jun-Sep Oct-Dec | |  |  | |  | |  |  | |

0 1696.3 980.3

1 2185.9 716.7

2 1874.0 690.6

3 1977.6 571.0

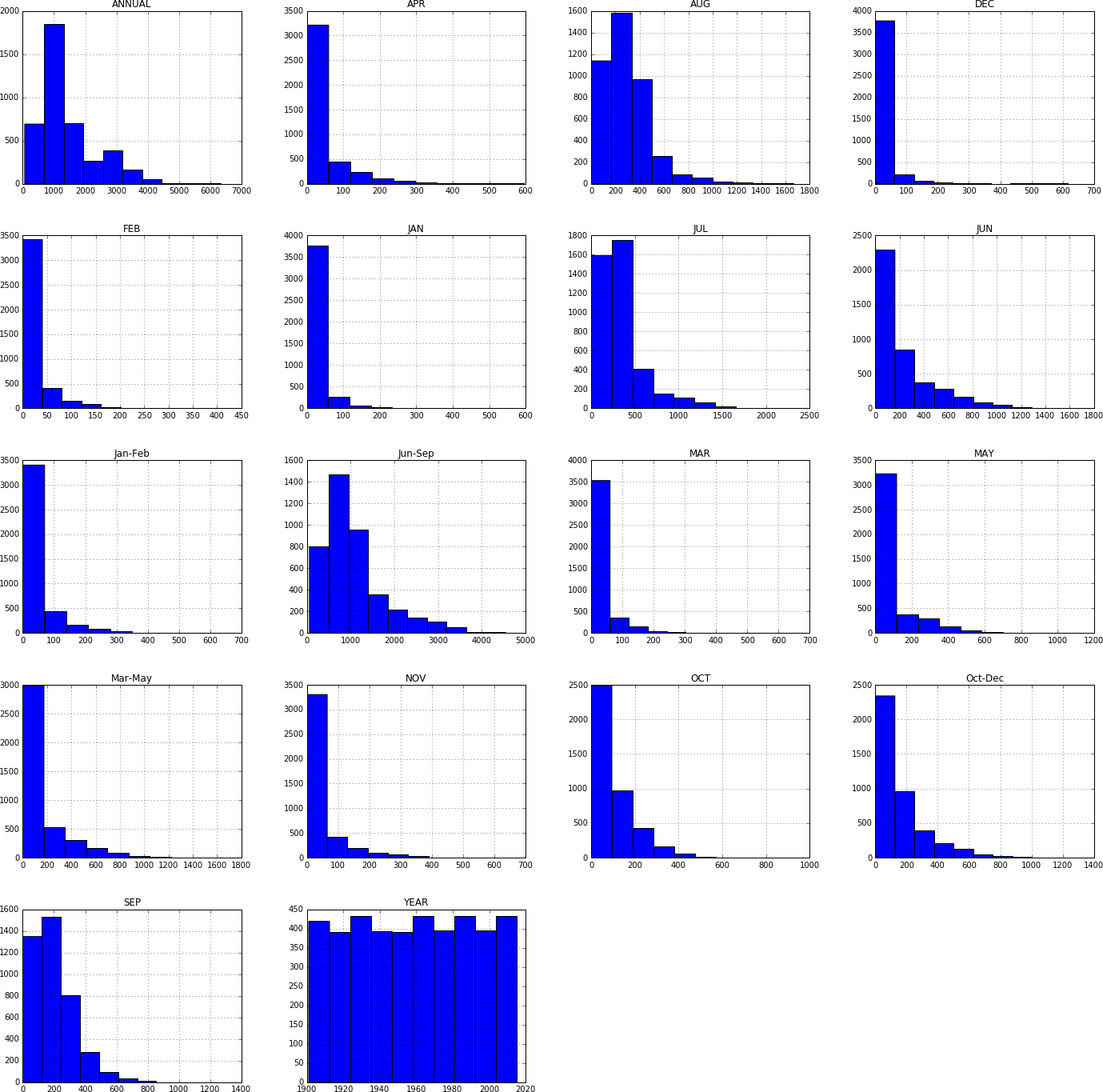
4 1624.9 630.8

In[4]: data.describle()

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Out[4]: | YEAR | JAN | FEB | MAR | APR | \ |
| count | 4116.000000 | 4116.000000 | 4116.000000 | 4116.000000 | 4116.000000 |  |
| mean | 1958.218659 | 18.957320 | 21.805325 | 27.359197 | 43.127432 |  |
| std | 33.140898 | 33.569044 | 35.896396 | 46.925176 | 67.798192 |  |
| min | 1901.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |  |
| 25 | 1930.000000 | 0.600000 | 0.600000 | 1.000000 | 3.000000 |  |
| 50 | 1958.000000 | 6.000000 | 6.700000 | 7.900000 | 15.700000 |  |
| 75 | 1987.000000 | 22.125000 | 26.800000 | 31.225000 | 49.825000 |  |
| max | 2015.000000 | 583.700000 | 403.500000 | 605.600000 | 595.100000 |  |
|  | MAY | JUN | JUL | AUG | SEP | \ |
| count | 4116.000000 | 4116.000000 | 4116.000000 | 4116.000000 | 4116.000000 |  |
| mean | 85.745417 | 230.234444 | 347.214334 | 290.263497 | 197.361922 |  |
| std | 123.189974 | 234.568120 | 269.310313 | 188.678707 | 135.309591 |  |
| min | 0.000000 | 0.400000 | 0.000000 | 0.000000 | 0.100000 |  |
| 25 | 8.600000 | 70.475000 | 175.900000 | 156.150000 | 100.600000 |  |
| 50 | 36.700000 | 138.900000 | 284.900000 | 259.500000 | 174.100000 |  |
| 75 | 96.825000 | 304.950000 | 418.225000 | 377.725000 | 265.725000 |  |
| max | 1168.600000 | 1609.900000 | 2362.800000 | 1664.600000 | 1222.000000 |  |
|  | OCT | NOV | DEC | ANNUAL | Jan-Feb | \ |
| count | 4116.000000 | 4116.000000 | 4116.000000 | 4116.000000 | 4116.000000 |  |
| mean | 95.507009 | 39.866163 | 18.870580 | 1411.008900 | 40.747786 |  |
| std | 99.434452 | 68.593545 | 42.318098 | 900.986632 | 59.265023 |  |
| min | 0.000000 | 0.000000 | 0.000000 | 62.300000 | 0.000000 |  |
| 25 | 14.600000 | 0.700000 | 0.100000 | 806.450000 | 4.100000 |  |
| 50 | 65.750000 | 9.700000 | 3.100000 | 1125.450000 | 19.300000 |  |
| 75 | 148.300000 | 45.825000 | 17.700000 | 1635.100000 | 50.300000 |  |
| max | 948.300000 | 648.900000 | 617.500000 | 6331.100000 | 699.500000 |  |
|  | Mar-May | Jun-Sep | Oct-Dec |  |  |  |
| count | 4116.000000 | 4116.000000 | 4116.000000 |  |  |  |
| mean | 155.901753 | 1064.724769 | 154.100487 |  |  |  |
| std | 201.096692 | 706.881054 | 166.678751 |  |  |  |
| min | 0.000000 | 57.400000 | 0.000000 |  |  |  |
| 25 | 24.200000 | 574.375000 | 34.200000 |  |  |  |
| 50 | 75.200000 | 882.250000 | 98.800000 |  |  |  |
| 75 | 196.900000 | 1287.550000 | 212.600000 |  |  |  |
| max | 1745.800000 | 4536.900000 | 1252.500000 |  |  |  |



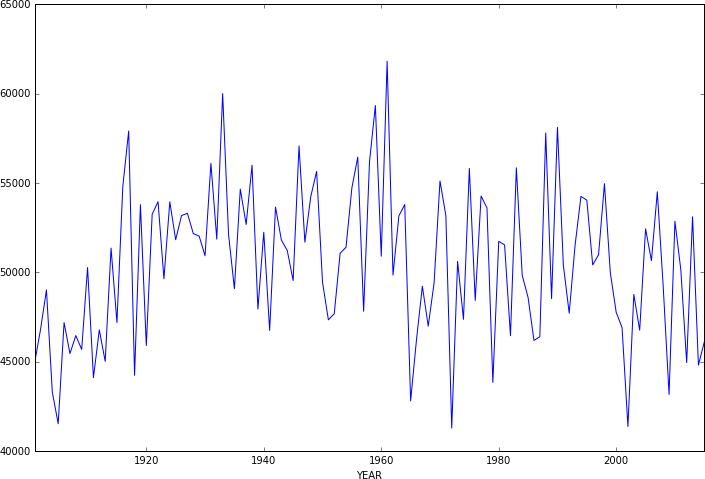
In [5]: data.hist(figsize=(24,24));



# Observations

* + - Above histograms show the distribution of rainfall over months.
    - Observed increase in amount of rainfall over months July, August, September.

In [6]: data.groupby("YEAR").sum()['ANNUAL'].plot(figsize=(12,8));

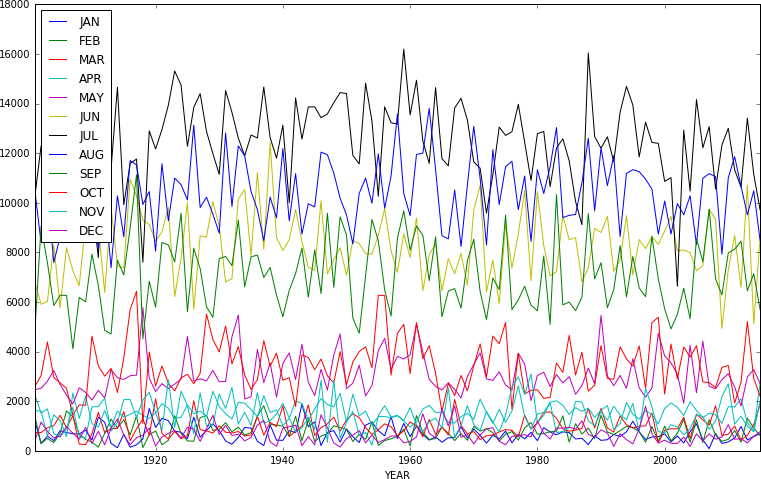


# Observations

* + - Shows distribution of rainfall over years.
    - Observed high amount of rainfall in 1950s.

In [7]: data[['YEAR', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

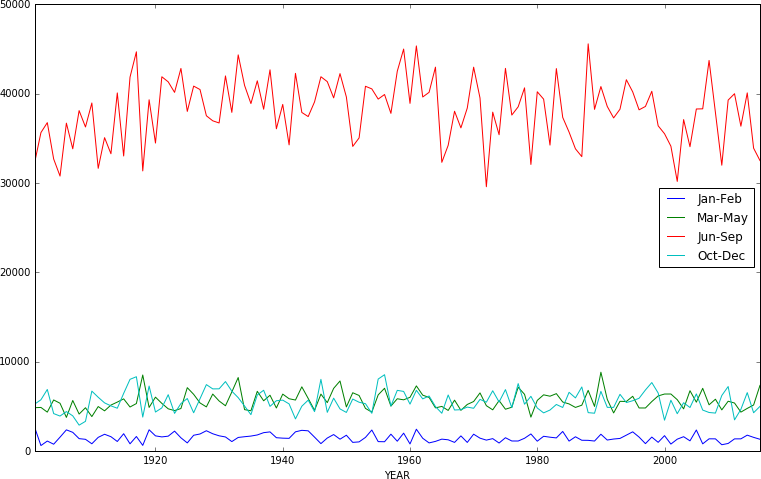
'AUG', 'SEP', 'OCT', 'NOV', 'DEC’]]



C']].groupby("YEAR").sum().plot(figsize=(13,8));

In [8]: data[['YEAR','Jan-Feb', 'Mar-May',

'Jun-Sep', 'Oct-Dec']].groupby("YEAR").sum().plot(figsize=(13,8));

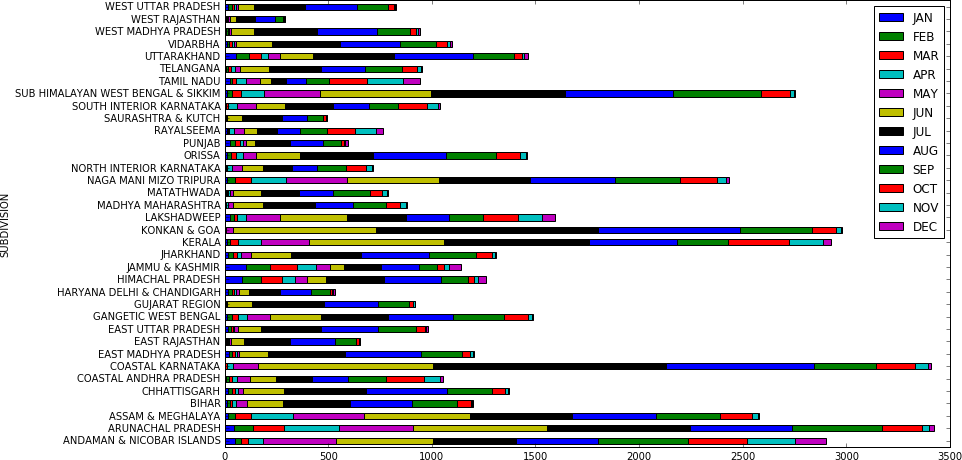


# Observations

* + - The above two graphs show the distribution of rainfall over months.
    - The graphs clearly shows that amount of rainfall in high in the months july, aug, sep which is monsoon season in India.

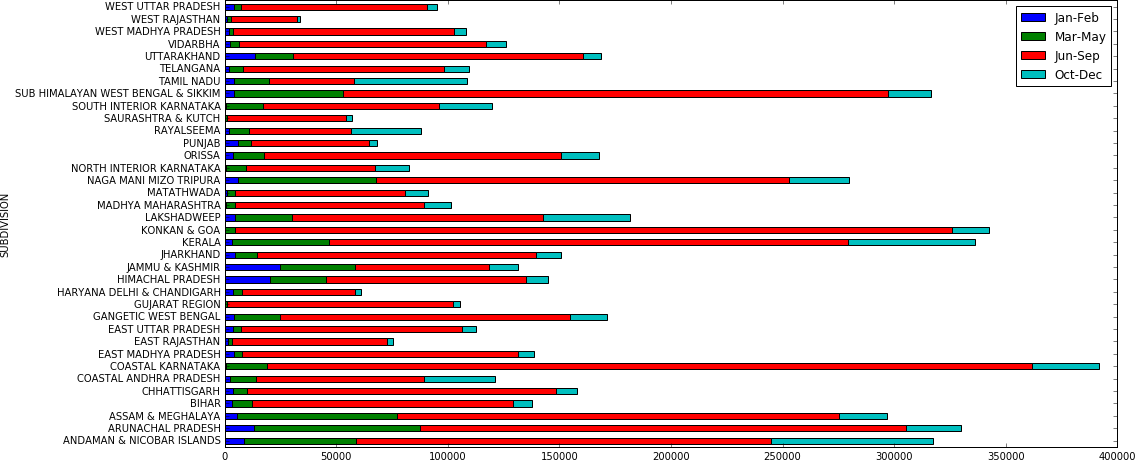
In [9]: data[['SUBDIVISION', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].groupby("SUBDIVISION").mean().plot.barh(stacked=True



In [10]: data[['SUBDIVISION', 'Jan-Feb', 'Mar-May',

'Jun-Sep', 'Oct-Dec']].groupby("SUBDIVISION").sum().plot.barh(stacked=True,figsize=(16,



# Observations

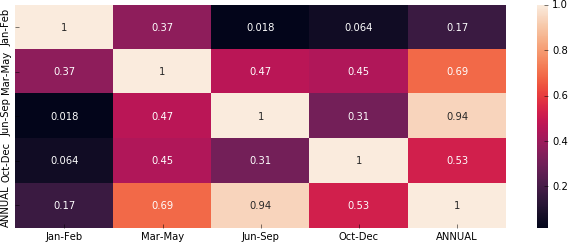
* + - Above two graphs shows that the amount of rainfall is reasonably good in the months of march, april, may in eastern India.

In [11]: plt.figure(figsize=(11,4))

sns.heatmap(data[['Jan-Feb','Mar-May','Jun-Sep','Oct-Dec','ANNUAL']].corr(),annot=True) plt.show()

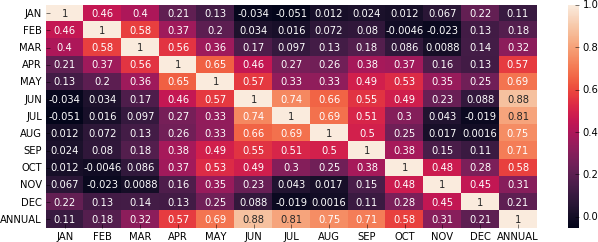
/home/sudheer.achary/.local/lib/python2.7/site-packages/pandas/core/computation/check.py:17: UserWarnin The minimum supported version is 2.4.6

ver=ver, min\_ver=\_MIN\_NUMEXPR\_VERSION), UserWarning)



In [12]: plt.figure(figsize=(11,4)) sns.heatmap(data[['JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OCT','NOV','DEC','ANN

plt.show()



# Observations

* + - **Heat Map** shows the co-relation(dependency) between the amounts of rainfall over months.
    - From above it is clear that if amount of rainfall is high in the months of july, august, september then the amount of rainfall will be high annually.
    - It is also obwserved that if amount of rainfall in good in the months of october, november, december then the rainfall is going to b good in the overall year.

In [13]: *#Function to plot the graphs*

def plot\_graphs(groundtruth,prediction,title):

N = 9

ind = np.arange(N) *# the x locations for the groups*

width = 0.27 *# the width of the bars*

fig = plt.figure() fig.suptitle(title, fontsize=12) ax = fig.add\_subplot(111)

rects1 = ax.bar(ind, groundtruth, width, color='r') rects2 = ax.bar(ind+width, prediction, width, color='g')

ax.set\_ylabel("Amount of rainfall")

ax.set\_xticks(ind+width)

ax.set\_xticklabels( ('APR', 'MAY', 'JUN', 'JUL','AUG', 'SEP', 'OCT', 'NOV', 'DEC') )

ax.legend( (rects1[0], rects2[0]), ('Ground truth', 'Prediction') )

*# autolabel(rects1)*

for rect in rects1:

h = rect.get\_height() ax.text(rect.get\_x()+rect.get\_width()/2., 1.05\*h, ' d' int(h),

ha='center', va='bottom') for rect in rects2:

h = rect.get\_height() ax.text(rect.get\_x()+rect.get\_width()/2., 1.05\*h, ' d' int(h),

ha='center', va='bottom')

*# autolabel(rects2)*

plt.show()

# Predictions

* + - For prediction we formatted data in the way, given the rainfall in the last three months we try to predict the rainfall in the next consecutive month.
    - For all the experiments we used 80:20 training and test ratio.
      * Linear regression
      * SVR
      * Artificial neural nets
    - Tersting metrics: We used Mean absolute error to train the models.
    - We also shown the amount of rainfall actually and predicted with the histogram plots.
    - We did two types of trainings once training on complete dataset and other with training with only telangana data
    - All means are standard deviation observations are written, first one represents ground truth, second one represents predictions.

In [14]: *# seperation of training and testing data*

from sklearn.model\_selection import train\_test\_split from sklearn.metrics import mean\_absolute\_error

division\_data = np.asarray(data[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL', 'AUG', 'SEP', 'OCT', 'NOV', 'DEC']])

X = None; y = None

for i in range(division\_data.shape[1]-3): if X is None:

X = division\_data[:, i:i+3] y = division\_data[:, i+3]

else:

X = np.concatenate((X, division\_data[:, i:i+3]), axis=0) y = np.concatenate((y, division\_data[:, i+3]), axis=0)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=42)

In [15]: *#test 2010*

temp = data[['SUBDIVISION','JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[data['YEAR'] == 2010]

data\_2010 = np.asarray(temp[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[temp['SUBDIVISION'] == 'TELANGANA'])

X\_year\_2010 = None; y\_year\_2010 = None for i in range(data\_2010.shape[1]-3):

if X\_year\_2010 is None:

X\_year\_2010 = data\_2010[:, i:i+3] y\_year\_2010 = data\_2010[:, i+3]

else:

X\_year\_2010 = np.concatenate((X\_year\_2010, data\_2010[:, i:i+3]), axis=0) y\_year\_2010 = np.concatenate((y\_year\_2010, data\_2010[:, i+3]), axis=0)

In [16]: *#test 2005*

temp = data[['SUBDIVISION','JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[data['YEAR'] == 2005]

data\_2005 = np.asarray(temp[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[temp['SUBDIVISION'] == 'TELANGANA'])

X\_year\_2005 = None; y\_year\_2005 = None for i in range(data\_2005.shape[1]-3):

if X\_year\_2005 is None:

X\_year\_2005 = data\_2005[:, i:i+3] y\_year\_2005 = data\_2005[:, i+3]

else:

X\_year\_2005 = np.concatenate((X\_year\_2005, data\_2005[:, i:i+3]), axis=0) y\_year\_2005 = np.concatenate((y\_year\_2005, data\_2005[:, i+3]), axis=0)

In [17]: *#terst 2015*

temp = data[['SUBDIVISION','JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[data['YEAR'] == 2015]

data\_2015 = np.asarray(temp[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[temp['SUBDIVISION'] == 'TELANGANA'])

X\_year\_2015 = None; y\_year\_2015 = None for i in range(data\_2015.shape[1]-3):

if X\_year\_2015 is None:

X\_year\_2015 = data\_2015[:, i:i+3] y\_year\_2015 = data\_2015[:, i+3]

else:

X\_year\_2015 = np.concatenate((X\_year\_2015, data\_2015[:, i:i+3]), axis=0) y\_year\_2015 = np.concatenate((y\_year\_2015, data\_2015[:, i+3]), axis=0)

In [18]: from sklearn import linear\_model

*# linear model*

reg = linear\_model.ElasticNet(alpha=0.5) reg.fit(X\_train, y\_train)

y\_pred = reg.predict(X\_test)

print mean\_absolute\_error(y\_test, y\_pred) 96.32435229744095

In [19]: *#2005*

y\_year\_pred\_2005 = reg.predict(X\_year\_2005)

*#2010*

y\_year\_pred\_2010 = reg.predict(X\_year\_2010) y\_year\_pred\_2015 = reg.predict(X\_year\_2015)

print "MEAN 2005"

print np.mean(y\_year\_2005),np.mean(y\_year\_pred\_2005) print "Standard deviation 2005"

print np.sqrt(np.var(y\_year\_2005)),np.sqrt(np.var(y\_year\_pred\_2005))

print "MEAN 2010"

print np.mean(y\_year\_2010),np.mean(y\_year\_pred\_2010) print "Standard deviation 2010"

print np.sqrt(np.var(y\_year\_2010)),np.sqrt(np.var(y\_year\_pred\_2010))

print "MEAN 2015"

print np.mean(y\_year\_2015),np.mean(y\_year\_pred\_2015) print "Standard deviation 2015"

print np.sqrt(np.var(y\_year\_2015)),np.sqrt(np.var(y\_year\_pred\_2015))

plot\_graphs(y\_year\_2005,y\_year\_pred\_2005,"Year-2005") plot\_graphs(y\_year\_2010,y\_year\_pred\_2010,"Year-2010") plot\_graphs(y\_year\_2015,y\_year\_pred\_2015,"Year-2015")

MEAN 2005

121.2111111111111 134.68699821349824

Standard deviation 2005

123.77066107608005 90.86310230416397

MEAN 2010

139.93333333333334 144.8050132651592

Standard deviation 2010

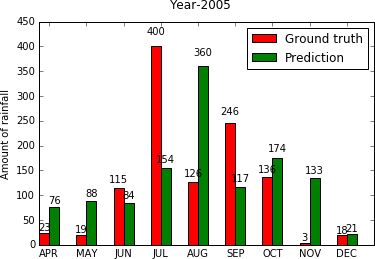
135.71320250194282 95.94931363601675

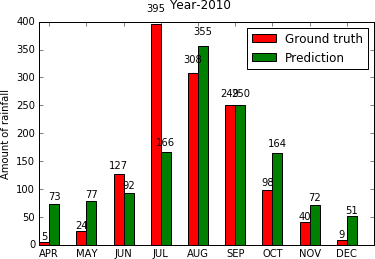
MEAN 2015

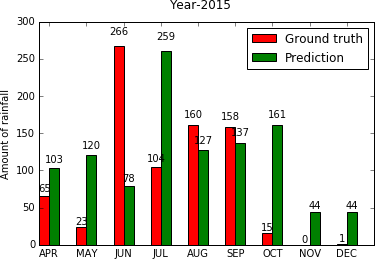
88.52222222222223 119.64752006738864

Standard deviation 2015

86.62446123324875 62.36355370163346







In [20]: from sklearn.svm import SVR

*# SVM model*

clf = SVR(gamma='auto', C=0.1, epsilon=0.2) clf.fit(X\_train, y\_train)

y\_pred = clf.predict(X\_test)

print mean\_absolute\_error(y\_test, y\_pred) 127.1600615632603

In [21]: *#2005*

y\_year\_pred\_2005 = reg.predict(X\_year\_2005)

*#2010*

y\_year\_pred\_2010 = reg.predict(X\_year\_2010)

*#2015*

y\_year\_pred\_2015 = reg.predict(X\_year\_2015) print "MEAN 2005"

print np.mean(y\_year\_2005),np.mean(y\_year\_pred\_2005) print "Standard deviation 2005"

print np.sqrt(np.var(y\_year\_2005)),np.sqrt(np.var(y\_year\_pred\_2005))

print "MEAN 2010"

print np.mean(y\_year\_2010),np.mean(y\_year\_pred\_2010) print "Standard deviation 2010"

print np.sqrt(np.var(y\_year\_2010)),np.sqrt(np.var(y\_year\_pred\_2010))

print "MEAN 2015"

print np.mean(y\_year\_2015),np.mean(y\_year\_pred\_2015) print "Standard deviation 2015"

print np.sqrt(np.var(y\_year\_2015)),np.sqrt(np.var(y\_year\_pred\_2015))

plot\_graphs(y\_year\_2005,y\_year\_pred\_2005,"Year-2005") plot\_graphs(y\_year\_2010,y\_year\_pred\_2010,"Year-2010") plot\_graphs(y\_year\_2015,y\_year\_pred\_2015,"Year-2015")

MEAN 2005

121.2111111111111 134.68699821349824

Standard deviation 2005

123.77066107608005 90.86310230416397

MEAN 2010

139.93333333333334 144.8050132651592

Standard deviation 2010

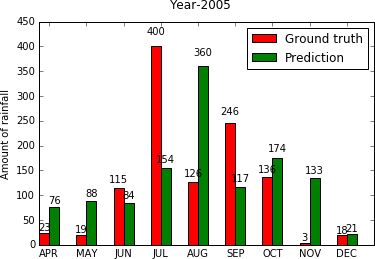
135.71320250194282 95.94931363601675

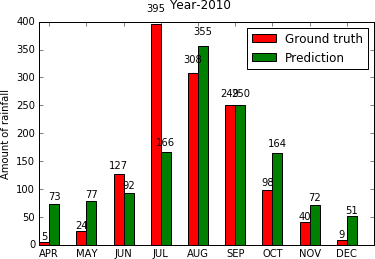
MEAN 2015

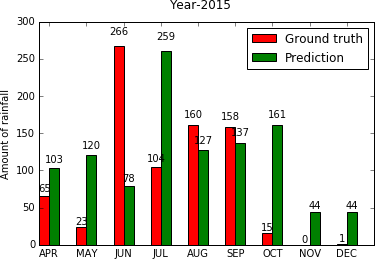
88.52222222222223 119.64752006738864

Standard deviation 2015

86.62446123324875 62.36355370163346







In [22]: from keras.models import Model

from keras.layers import Dense, Input, Conv1D, Flatten

*# NN model*

inputs = Input(shape=(3,1))

x = Conv1D(64, 2, padding='same', activation='elu')(inputs) x = Conv1D(128, 2, padding='same', activation='elu')(x)

x = Flatten()(x)

x = Dense(128, activation='elu')(x) x = Dense(64, activation='elu')(x) x = Dense(32, activation='elu')(x)

x = Dense(1, activation='linear')(x)

model = Model(inputs=[inputs], outputs=[x]) model.compile(loss='mean\_squared\_error', optimizer='adamax', metrics=['mae']) model.summary()

/home/sudheer.achary/.local/lib/python2.7/site-packages/h5py/ init .py:36: FutureWarning: Conversion from .\_conv import register\_converters as \_register\_converters

Using TensorFlow backend.



Layer (type) Output Shape Param #

=================================================================

|  |  |  |
| --- | --- | --- |
| input\_1 (InputLayer) | (None, 3, 1) | 0 |

|  |  |  |
| --- | --- | --- |
| conv1d\_1 (Conv1D) | (None, 3, 64) | 192 |

|  |  |  |  |
| --- | --- | --- | --- |
| conv1d\_2 (Conv1D) | (None, | 3, 128) | 16512 |
| flatten\_1 (Flatten) | (None, | 384) | 0 |
| dense\_1 (Dense) | (None, | 128) | 49280 |
| dense\_2 (Dense) | (None, | 64) | 8256 |
| dense\_3 (Dense) | (None, | 32) | 2080 |
| dense\_4 (Dense) | (None, | 1) | 33 |

=================================================================

Total params: 76,353

Trainable params: 76,353

Non-trainable params: 0



In [23]: model.fit(x=np.expand\_dims(X\_train, axis=2), y=y\_train, batch\_size=64, epochs=10, verbose=1, v y\_pred = model.predict(np.expand\_dims(X\_test, axis=2))

print mean\_absolute\_error(y\_test, y\_pred) Train on 30005 samples, validate on 3334 samples

92.28250624049363

In [24]: *#2005*

y\_year\_pred\_2005 = reg.predict(X\_year\_2005)

*#2010*

y\_year\_pred\_2010 = reg.predict(X\_year\_2010)

*#2015*

y\_year\_pred\_2015 = reg.predict(X\_year\_2015)

print "MEAN 2005"

print np.mean(y\_year\_2005),np.mean(y\_year\_pred\_2005) print "Standard deviation 2005"

print np.sqrt(np.var(y\_year\_2005)),np.sqrt(np.var(y\_year\_pred\_2005))

print "MEAN 2010"

print np.mean(y\_year\_2010),np.mean(y\_year\_pred\_2010) print "Standard deviation 2010"

print np.sqrt(np.var(y\_year\_2010)),np.sqrt(np.var(y\_year\_pred\_2010))

print "MEAN 2015"

print np.mean(y\_year\_2015),np.mean(y\_year\_pred\_2015) print "Standard deviation 2015"

print np.sqrt(np.var(y\_year\_2015)),np.sqrt(np.var(y\_year\_pred\_2015))

plot\_graphs(y\_year\_2005,y\_year\_pred\_2005,"Year-2005") plot\_graphs(y\_year\_2010,y\_year\_pred\_2010,"Year-2010") plot\_graphs(y\_year\_2015,y\_year\_pred\_2015,"Year-2015")

MEAN 2005

121.2111111111111 134.68699821349824

Standard deviation 2005

123.77066107608005 90.86310230416397

MEAN 2010

139.93333333333334 144.8050132651592

Standard deviation 2010

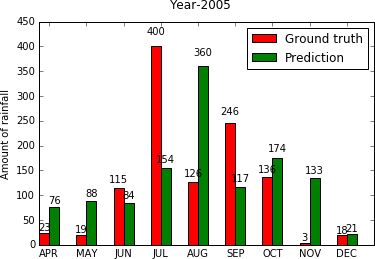
135.71320250194282 95.94931363601675

MEAN 2015

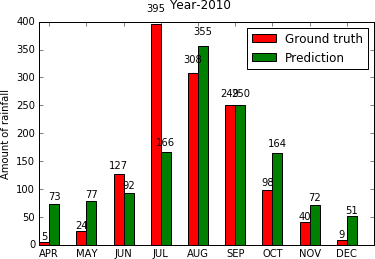
88.52222222222223 119.64752006738864

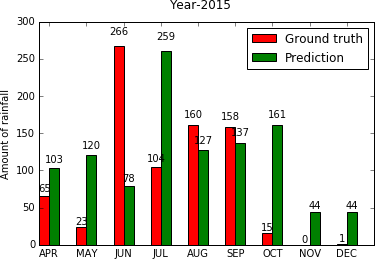
Standard deviation 2015

86.62446123324875 62.36355370

163346

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Epoch 1/10  30005/30005  Epoch 2/10 | [==============================] | - | 3s | 111us/step - loss: 19589.7591 - mean\_absolute\_error: | | | |
| 30005/30005  Epoch 3/10 30005/30005 | [==============================]  [==============================] | * 2s * 2s | | 53us/step  54us/step | * loss: * loss: | 18582.2211  18466.6604 | * mean\_absolute\_error: * mean\_absolute\_error: |
| Epoch 4/10  30005/30005 | [==============================] | - 2s | | 54us/step | - loss: | 18482.5326 | - mean\_absolute\_error: |
| Epoch 5/10  30005/30005 | [==============================] | - 2s | | 54us/step | - loss: | 18358.7726 | - mean\_absolute\_error: |
| Epoch 6/10  30005/30005 | [==============================] | - 2s | | 53us/step | - loss: | 18312.5666 | - mean\_absolute\_error: |
| Epoch 7/10  30005/30005 | [==============================] | - 2s | | 54us/step | - loss: | 18236.9615 | - mean\_absolute\_error: |
| Epoch 8/10  30005/30005 | [==============================] | - 2s | | 54us/step | - loss: | 18118.3601 | - mean\_absolute\_error: |
| Epoch 9/10  30005/30005 | [==============================] | - 2s | | 54us/step | - loss: | 18193.9362 | - mean\_absolute\_error: |
| Epoch 10/10  30005/30005 | [==============================] | - 2s | | 54us/step | - loss: | 18007.9055 | - mean\_absolute\_error: |





In [25]: *# spliting training and testing data only for telangana*

telangana = np.asarray(data[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',

'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].loc[data['SUBDIVISION'] == 'TELANGANA'])

X = None; y = None

for i in range(telangana.shape[1]-3): if X is None:

X = telangana[:, i:i+3] y = telangana[:, i+3]

else:

X = np.concatenate((X, telangana[:, i:i+3]), axis=0) y = np.concatenate((y, telangana[:, i+3]), axis=0)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.01, random\_state=42)

In [26]: from sklearn import linear\_model

*# linear model*

reg = linear\_model.ElasticNet(alpha=0.5) reg.fit(X\_train, y\_train)

y\_pred = reg.predict(X\_test)

print mean\_absolute\_error(y\_test, y\_pred) 64.72601914484643

In [27]: *#2005*

y\_year\_pred\_2005 = reg.predict(X\_year\_2005)

*#2010*

y\_year\_pred\_2010 = reg.predict(X\_year\_2010)

*#2015*

y\_year\_pred\_2015 = reg.predict(X\_year\_2015)

print "MEAN 2005"

print np.mean(y\_year\_2005),np.mean(y\_year\_pred\_2005) print "Standard deviation 2005"

print np.sqrt(np.var(y\_year\_2005)),np.sqrt(np.var(y\_year\_pred\_2005))

print "MEAN 2010"

print np.mean(y\_year\_2010),np.mean(y\_year\_pred\_2010) print "Standard deviation 2010"

print np.sqrt(np.var(y\_year\_2010)),np.sqrt(np.var(y\_year\_pred\_2010))

print "MEAN 2015"

print np.mean(y\_year\_2015),np.mean(y\_year\_pred\_2015) print "Standard deviation 2015"

print np.sqrt(np.var(y\_year\_2015)),np.sqrt(np.var(y\_year\_pred\_2015))

plot\_graphs(y\_year\_2005,y\_year\_pred\_2005,"Year-2005") plot\_graphs(y\_year\_2010,y\_year\_pred\_2010,"Year-2010") plot\_graphs(y\_year\_2015,y\_year\_pred\_2015,"Year-2015")

MEAN 2005

121.2111111111111 106.49798150231584

Standard deviation 2005

123.77066107608005 76.08558540019227

MEAN 2010

139.93333333333334 112.18662987131034

Standard deviation 2010

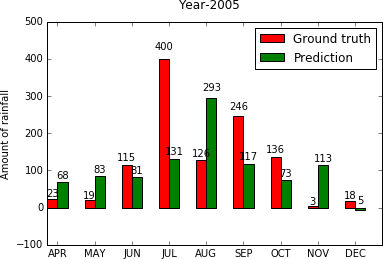
135.71320250194282 84.35813629737324

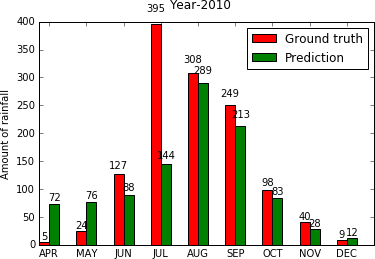
MEAN 2015

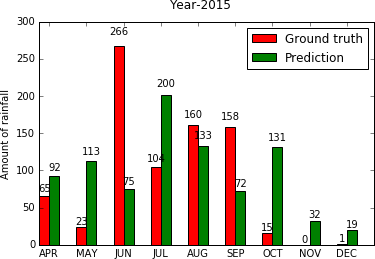
88.52222222222223 96.76817006572782

Standard deviation 2015

86.62446123324875 52.45304841713261







In [28]: from sklearn.svm import SVR

*# SVM model*

clf = SVR(kernel='rbf', gamma='auto', C=0.5, epsilon=0.2) clf.fit(X\_train, y\_train)

y\_pred = clf.predict(X\_test)

print mean\_absolute\_error(y\_test, y\_pred) 115.32415990638656

In [29]: *#2005*

y\_year\_pred\_2005 = reg.predict(X\_year\_2005)

*#2010*

y\_year\_pred\_2010 = reg.predict(X\_year\_2010)

*#2015*

y\_year\_pred\_2015 = reg.predict(X\_year\_2015)

print "MEAN 2005"

print np.mean(y\_year\_2005),np.mean(y\_year\_pred\_2005) print "Standard deviation 2005"

print np.sqrt(np.var(y\_year\_2005)),np.sqrt(np.var(y\_year\_pred\_2005))

print "MEAN 2010"

print np.mean(y\_year\_2010),np.mean(y\_year\_pred\_2010)

print "Standard deviation 2010"

print np.sqrt(np.var(y\_year\_2010)),np.sqrt(np.var(y\_year\_pred\_2010))

print "MEAN 2015"

print np.mean(y\_year\_2015),np.mean(y\_year\_pred\_2015) print "Standard deviation 2015"

print np.sqrt(np.var(y\_year\_2015)),np.sqrt(np.var(y\_year\_pred\_2015))

plot\_graphs(y\_year\_2005,y\_year\_pred\_2005,"Year-2005") plot\_graphs(y\_year\_2010,y\_year\_pred\_2010,"Year-2010") plot\_graphs(y\_year\_2015,y\_year\_pred\_2015,"Year-2015")

MEAN 2005

121.2111111111111 106.49798150231584

Standard deviation 2005

123.77066107608005 76.08558540019227

MEAN 2010

139.93333333333334 112.18662987131034

Standard deviation 2010

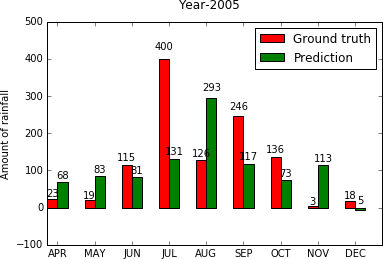
135.71320250194282 84.35813629737324

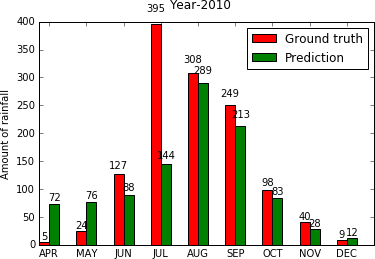
MEAN 2015

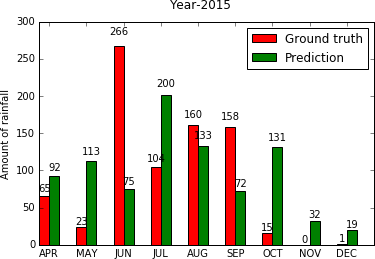
88.52222222222223 96.76817006572782

Standard deviation 2015

86.62446123324875 52.45304841713261







In [30]: model.fit(x=np.expand\_dims(X\_train, axis=2), y=y\_train, batch\_size=64, epochs=10, verbose=1, v

y\_pred = model.predict(np.expand\_dims(X\_test, axis=2)) print mean\_absolute\_error(y\_test, y\_pred)

Train on 921 samples, validate on 103 samples Epoch 1/10

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In | [31]: | *#2005* |  | |
|  |  | y\_year\_pred\_2005 | = | reg.predict(X\_year\_2005) |

*#2010*

y\_year\_pred\_2010 = reg.predict(X\_year\_2010)

*#2015*

y\_year\_pred\_2015 = reg.predict(X\_year\_2015)

print "MEAN 2005"

print np.mean(y\_year\_2005),np.mean(y\_year\_pred\_2005) print "Standard deviation 2005"

print np.sqrt(np.var(y\_year\_2005)),np.sqrt(np.var(y\_year\_pred\_2005))

print "MEAN 2010"

print np.mean(y\_year\_2010),np.mean(y\_year\_pred\_2010) print "Standard deviation 2010"

print np.sqrt(np.var(y\_year\_2010)),np.sqrt(np.var(y\_year\_pred\_2010))

print "MEAN 2015"

print np.mean(y\_year\_2015),np.mean(y\_year\_pred\_2015) print "Standard deviation 2015"

print np.sqrt(np.var(y\_year\_2015)),np.sqrt(np.var(y\_year\_pred\_2015))

plot\_graphs(y\_year\_2005,y\_year\_pred\_2005,"Year-2005") plot\_graphs(y\_year\_2010,y\_year\_pred\_2010,"Year-2010

plot\_graphs(y\_year\_2015,y\_year\_pred\_2015,"Year-2015")

MEAN 2005

121.2111111111111 106.49798150231584

Standard deviation 2005

123.77066107608005 76.08558540019227

MEAN 2010

139.93333333333334 112.18662987131034

Standard deviation 2010

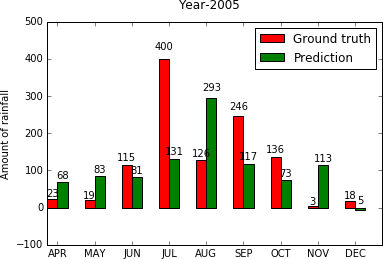
135.71320250194282 84.35813629737324

MEAN 2015

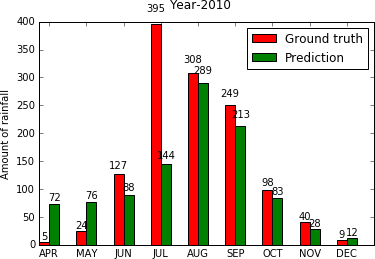
88.52222222222223 96.76817006572782

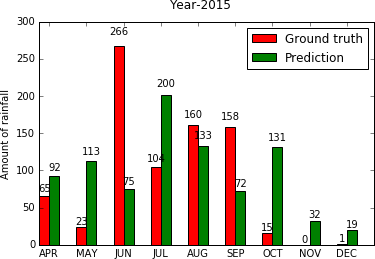
Standard deviation 2015

86.62446123324875 52.45304841713261



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 921/921 [==============================] | - 0s | 66us/step | - loss: | 7274.9487 | - mean\_absolute\_error: | 63.502 |
| Epoch 2/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 56us/step | - loss: | 6431.8426 | - mean\_absolute\_error: | 56.767 |
| Epoch 3/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 56us/step | - loss: | 6046.0127 | - mean\_absolute\_error: | 58.486 |
| Epoch 4/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 56us/step | - loss: | 5883.5181 | - mean\_absolute\_error: | 56.438 |
| Epoch 5/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 57us/step | - loss: | 5764.2698 | - mean\_absolute\_error: | 55.178 |
| Epoch 6/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 55us/step | - loss: | 5706.7510 | - mean\_absolute\_error: | 55.346 |
| Epoch 7/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 56us/step | - loss: | 5636.2414 | - mean\_absolute\_error: | 54.452 |
| Epoch 8/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 57us/step | - loss: | 5564.0726 | - mean\_absolute\_error: | 54.566 |
| Epoch 9/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 57us/step | - loss: | 5529.6288 | - mean\_absolute\_error: | 54.002 |
| Epoch 10/10 |  |  |  |  |  |  |
| 921/921 [==============================] | - 0s | 57us/step | - loss: | 5478.9296 | - mean\_absolute\_error: | 53.525 |
| 65.82400645938786 |  |  |  |  |  |  |





# 

**Chapter 7**

**OBSERVATION & CONCLUSION**

# 7 Observations & Conclusion

## 7.1 Training on complete dataset

|  |  |
| --- | --- |
| Algorithm | MAE |
| Linear Regression | 94.94821727619338 |
| SVR | 127.74073860203839 |
| Artificial neural nets | 85.2648713528865 |

* + - * Various visualizations of data are observed which helps in implementing the approaches for prediction.
      * Observations indicates machine learning models won’t work well for prediction of rainfall due to fluctuations in rainfall.
      * Neural Networks performs better than SVR etc.
      * Observed MAE is very high which indicates machine learning models won’t work well for prediction of rainfall.
      * Approximately close means, noticed less standard deviations.

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

# 

**Chapter 8**

**REFERENCES / BIBLIOGRAPHY**

**8 REFERENCES**

[1] Thirumalai, Chandrasegar, et al. "Heuristic prediction of rainfall using machine learning techniques." 2017 International Conference on Trends in Electronics and Informatics (ICEI). IEEE, 2017.

[2] Geetha, A., and G. M. Nasira. "Data mining for meteorological applications: Decision trees for modeling rainfall prediction." 2014 IEEE International Conference on Computational Intelligence and Computing Research. IEEE, 2014

[3] Parmar, Aakash, Kinjal Mistree, and Mithila Sompura. "Machine learning techniques for rainfall prediction: A review." 2017 International Conference on Innovations in information Embedded and Communication Systems. 2017.

[4] Dash, Yajnaseni, Saroj K. Mishra, and Bijaya K. Panigrahi. "Rainfall prediction for the Kerala state of India using artificial intelligence approaches." Computers & Electrical Engineering 70 (2018): 66-73.

[5] Singh, Gurpreet, and Deepak Kumar. "Hybrid Prediction Models for Rainfall Forecasting." 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, 2019.

[6] Kar, Kaveri, Neelima Thakur, and Prerika Sanghvi. "Prediction of Rainfall Using Fuzzy Dataset." (2019).

[7] Sardeshpande, Kaushik D., and Vijaya R. Thool. "Rainfall Prediction: A Comparative Study of Neural Network Architectures." Emerging Technologies in Data Mining and Information Security. Springer, Singapore, 2019. 19-28.

[8] Chen, Binghong, et al. "Non-Linear Machine Learning Approach to Short-Term Precipitation Forecasting." (2018).

[9] Moon, Seung-Hyun, et al. "Application of machine learning to an early warning system for very short-term heavy rainfall.―Journal of hydrology 568 (2019): 1042-1054. [10] https://data.gov.in/resources/subdivision-wise-rainfall-andits-departure-1901-2015