**Machine Learning Project**

**WEATHER FORECAST USING HISTORICAL DATA**

**REPORT**

**INSTRUCTOR**

1. Logo

   Description automatically generatedC. Fong (P)

**Submitted By**

**Lakshitha Parvathaneni**

**Tejasree Gajjala**

**Mohitha sai Kothapalli**

Text

Description automatically generated

**TABLE OF CONTENTS**

1 **INTRODUCTION** ..............................................................................

**2 REQUIREMENT ANALYSIS** ..........................................................

2.1 FUNCTIONAL REQUIREMENTS ............................................................

2.2 DATASET ............................................................................................

3 **BACKGROUND ANALYSIS**

...........................................................

* 1. A LOOK AT DATA ................................................................................
  2. METHODOLOGY .................................................................................

# 4 EXPLANATION OF CODE

................................................................................

5 CONCLUSION

...............................................................................

6 REFERENCES

................................................................................

# INTRODUCTION

Global climate change have been measurable impact on the atmosphere. Glaciers have shrunk, ice is breaking up faster on rivers and streams, plant and animal ranges have changed, and plants are blooming sooner. There are now consequences that scientists have expected in the past would occur from global climate change: melting of sea ice, rapid increase of sea level and longer, more extreme heat waves.

This project focuses on doing predictive analysis about the weather based on historical weather dataset.

**Techniques Used**

Different supervised learning Machine learning algorithms have been implemented using different software and big data tool such as Jupyter and Spark.

# REQUIREMENT ANALYSIS

**2.1 Functional Requirements**

Providing the details of the project, all requirements and specifications were met. Some of the functionalities in our project requires the basic principles of Data science along with Python3 Programming and implementation is done on software called as Jupyter Notebook. Supervised Learning of data prediction techniques from Machine Learning has been implemented. Most of the important data science related libraries for Python have been implemented in our code in the best possible way to enhance the quality of our project. All the work has been done in the Jupyter Notebook environment using base language as Python3. This provides a better work environments and faster run time for Data Science related work as compared to other IDEs’ of Python available out there.

# 2.2 Dataset

The dataset consists of the historical weather of the data from the year 2006 to 2016.

The data has several complications. There were several empty cells and unused specifications.

The dataset contains 10 years of high temporal resolution (hourly measurements) data.

It has 96454 records and 12 attributes.

The dataset is of 16 MB.

The features of the dataset are Precipitation, Temperature, Apparent Temperature, Humidity, Wind Speed, Wind Bearing, Visibility, Loud Cover, Pressure, Daily Summary.

Link: <https://www.kaggle.com/muthuj7/weather-dataset>

# BACKGROUND ANALYSIS

**3.1 A Look at Data**

Dataset with 96454 rows and 12 columns. Showing the first five rows only.

A screenshot of a computer

Description automatically generated with medium confidence

**3.2 Methodology and approach**

In order to develop a weather prediction model, good amount of data for training is needed. The data has been taken from Kaggle website. The data originally had 12 columns namely date/time, temperature, humidity, wind speed, precipitation type, apparent temperature, wind bearing, visibility, cloud cover, pressure, and summary. Summary column is the target class to be classified into 26 various weather types. The dataset has total 96453 data points or rows. The machine learning pipeline follows specific procedures for efficient prediction with acceptable error rate.

These steps are :

• Data collection

* Data exploration
* Visualization
* Data cleaning
* Feature extraction
* Model selection/Cross validation
* Model validation

* **Data storage locations**: The dataset weather.csv is stored in the local drive of our machines and will be used as the basis for the whole project.
* **Data ingestion**: Importing our set of data from the stored location to our main code written in jupyter notebook which will further be checked for Nan values and outliers and then it will be cleaned and processed. This includes steps like understanding our data, collecting it, describing the data, exploring, and verifying the quality of it.

* **Data Visualization and Cleaning**:

Various properties of every column were explored. After the data exploration the data is drawn on various graphs to identify any hidden patterns or properties previously unknown, or correlation among various columns. For this project bar graphs, distribution plots, count plot, box plot and heat maps were used.

For the visualization, we have used some libraries namely: matplotlib, seaborn and pandas package.

Matplotlib is a comprehensive library for creating static, animated visualizations in python. Matplot.pyplot is a collection of functions that make matplotlib work like MATLAB. Most of the matplotlib utilities lies under the pyplot submodule and are usually imported under the plt. Along with matplotlib we have used “seaborn”, which provides high level interface for drawing attractive and informative statistical graphics. We have also accessed pandas’ package that provided us fast, flexible, and expressive data analysis. The different graphs that we have visualized for the dataset are countplot, distplot, pairplot, heatmap and boxplot.

A picture containing bar chart

Description automatically generated

Fig 1: Countplot of Summary attribute

For the image (Fig 1), we have taken countplot graph for the summary attribute, countplot is like a histogram or bar graph for some categorical area. It shows the number of occurrences of an item based on a certain type of category. Clearly dataset retrieved for the project is unbalanced as we see from the above figure, that some features like partly cloudy, mostly cloudy have got more occurrences and some features like drizzle, rain, etc..., lie far behind.

“Seaborn distplot” shows a histogram with a line on it, which helps us to visualize all kinds of variations, distplot plots a univariate distribution of

observations. In this x axis will be the value of variable and y axis to be the probability density function. For all the observations we have accessed using distplot, we have taken the bins to be 40, bins is the range that how many bars you want to see.

Chart, histogram

Description automatically generated

Fig 2: distplot for Apparent Temperature

Fig 2 is the histogram for Apparent Temperature using distplot and as we clearly see there is a line on the bars that is helping to see all the variations in the attribute values. Fig 2 is a bimodal graph as the bars show the data is varying and not in a single direction.

Chart, histogram

Description automatically generated

Fig 3: distplot for Humidity

Fig 3 is a histogram for Humidity, this is a unimodal graph as the line of graph shows us that the value of humidity is increasing by the years passing as of the dates accessed in the dataset. “Pairplot” function lets us view the pairwise relationships in the dataset, it creates a grid of axes

such that each variable in data will be shared in the Y axis across a single row in the X axis across a single column.

A picture containing window, building

Description automatically generated

Fig 4: Pairplot

Fig 4 is a pairplot graph of the dataset with all the variables on comparison with same variables in x and y axis, this allows to see how the categorized between each attribute. Because of checking pairplot, this helped us to get to know that pressure attribute in the dataset has values that are only either maximum or minimum as shown in the Fig 3, So we have decided to drop the pressure attribute from the dataset as that won’t be helping us with the implementation.

Graphical user interface, chart, treemap chart

Description automatically generated

Fig 5: Heatmap

After pairplot we want to check heatmap as it is graphical representation of data where values are depicted by color, it makes us easy to visualize complex data and understand it briefly.

“Boxplot” shows the distribution of quantitative data on comparing between variables, it has components like interquartile range that is from the 25th percentile to 75th percentile in a box and line in the between that is the medium and outliers in the graph, they are the points that lie outside the range in which we expect them.

Chart

Description automatically generated

Fig 6: Boxplot of Apparent Temperature and Summary

Fig 6 is boxplot of Apparent Temperature and Summary, it shows that some attributes of data are symmetric as line in box is close to center and there are outliers present in partly cloudy, foggy, overcast, these lay out that there is unbalance in attributes like partly cloudy and overcast on comparison with Apparent Temperature values.

Chart, box and whisker chart

Description automatically generated

Fig 7: boxplot of Humidity and Summary

Fig 7 is a boxplot graph on comparison with Humidity values and Summary variables in this lot of variables are symmetric as the line in box are close to centre and less outliers on comparing with Fig 6, In this overcast and foggy are the attributes that exhibits the outliers.

**DATA CLEANING:**

Upon Visualizing the data using various graphs with attribute values presented and looking at all the metrics and to use the best possible accurate input for the implementation of algorithm. We have dropped some of the attributes namely: Formatted Date, Precip Type, Daily Summary, summary map, Cloud Cover, Pressure. As clearly the data is unbalanced with summary attribute, we have dropped it and on observing pairplot with pressure attribute being not useful and values are lie far from expected so we have dropped pressure attribute as well, Daily summary doesn’t help with prediction purpose. Fig 8 is a image of Code that we have used for Cleaning of data.

Some standardized cleaning procedures were followed

•Conversion of string type numeric values to float/double type.

•One-hot encoding.

•Mapping of string type columns to numeric values

Graphical user interface, text, application, email

Description automatically generated

Fig 8: Snippet of Code for cleaning.

❖ **Implementation of Random Forest Classifier in ML-Python and Pyspark:**

In this stage, random forest algorithm was selected to implement the project. Random Forest is a supervised learning algorithmic approach which can be used for both classification and regression problems. It uses a forest or collection of decision trees to obtain the classification result. Random forest is mainly based on two stages, the first stage being the creation of the random forest and the second stage is the prediction from the random forest being created in the first stage.

After the processes of data visualization and data cleaning, the data frame obtained is in ‘clean’ form or in a format ready to be fed to a machine learning model. PySpark has been used for parallelization of training and prediction process of the model. PySpark uses Big Data architecture by connecting to apache spark through an API and running python code on concurrent nodes. This makes the training process on large datasets very fast. PySpark uses a RDD for its implementation. RDD stands for Resilient Distributed Dataset, these are the elements that run and operate on multiple nodes to do parallel processing on a cluster. RDDs are immutable elements, which means once you create an RDD you cannot change it. RDDs are fault tolerant as well, hence in case of any failure, they recover automatically. You can apply multiple operations on these RDDs to achieve a certain task. When a task arrives to be implemented it is divided by a ‘master node’ which divides task and keep track of which task is given to which ‘task node’ for implementation. When each node completes the processing, the result is then presented in a combined form by the master node. This is similar to MapReduce done in Hadoop. PySpark uses Hadoop as base for its implementation.

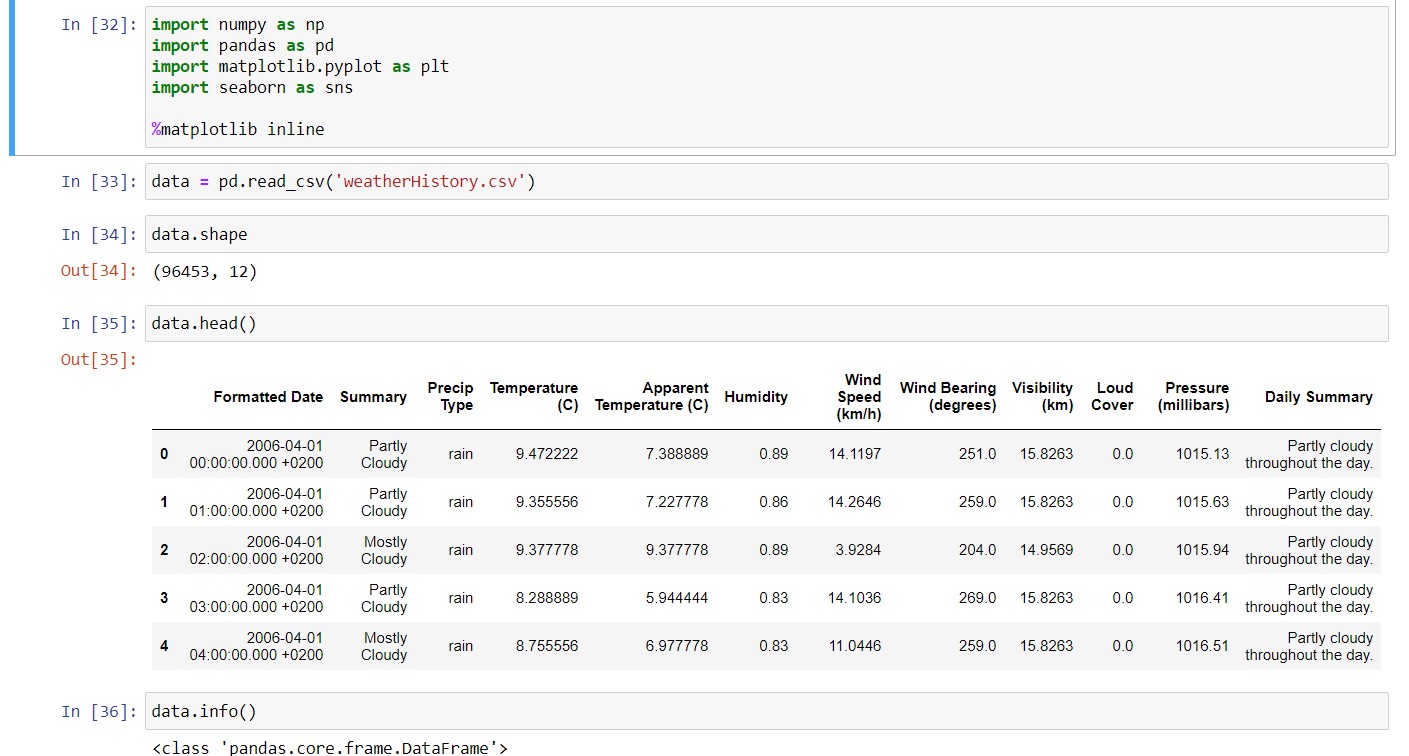
After the data was cleaned, split into train and test dataset, converted to an RDD object, the train set was fed to the machine learning model for training. Random Forest Classifier has been used to prediction in the project. It is an improvement over decision tree model by creating a random bootstrap of trees to reduce bias. After the model is trained, validation error rate/accuracy of the model was recorded. The accuracy achieved was 58.3% with 100 trees in the random forest model. Weighted F1 score achieved was 0.58 with 0.58 precision .

**CODE IN DETAIL**

In this section the code is explained line by line and why that method was applied and its description in detail.

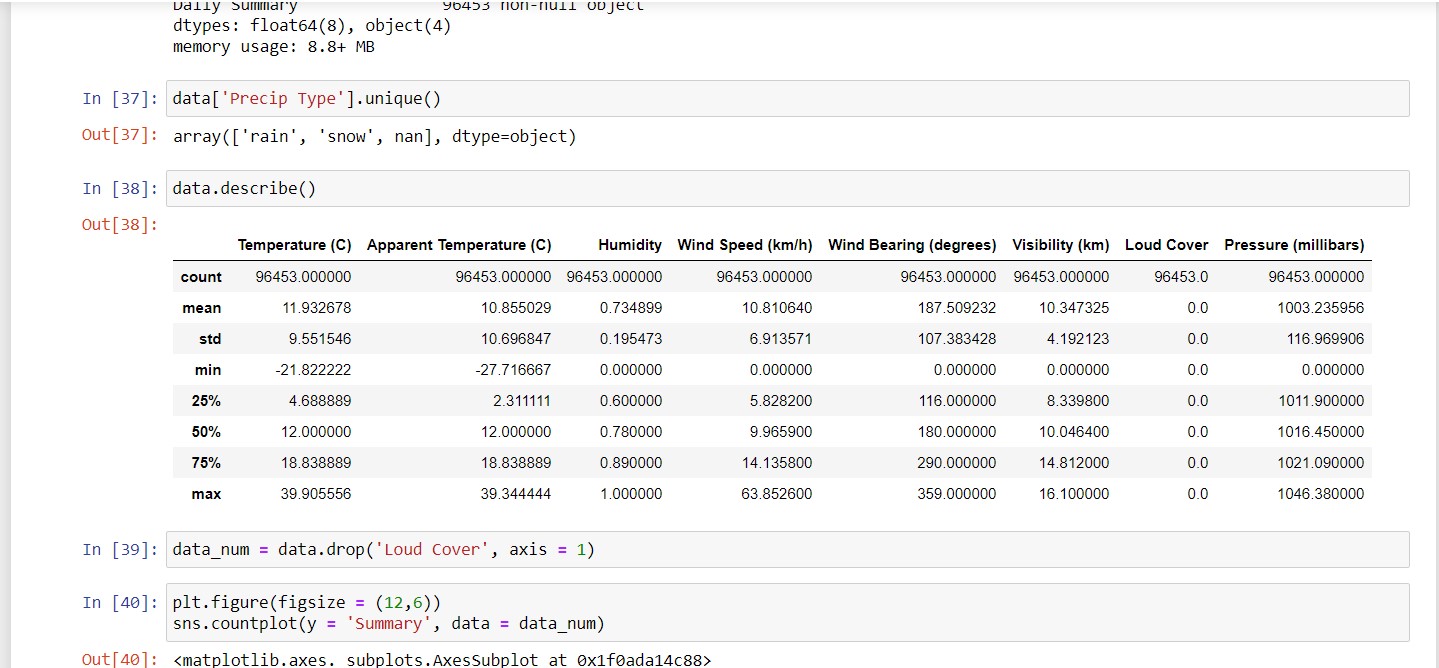
1. **Importing libraries and dataset ‘weatherHistory.csv’ as a data frame object:**

All the required libraries were imported such as numpy, pandas, matplot and seaborn for data manipulation, exploration and visualization. The was imported as a data frame object using ‘pd.read\_csv()’ function under pandas library. Shape of the data is seen to get the idea of size of the dataset.



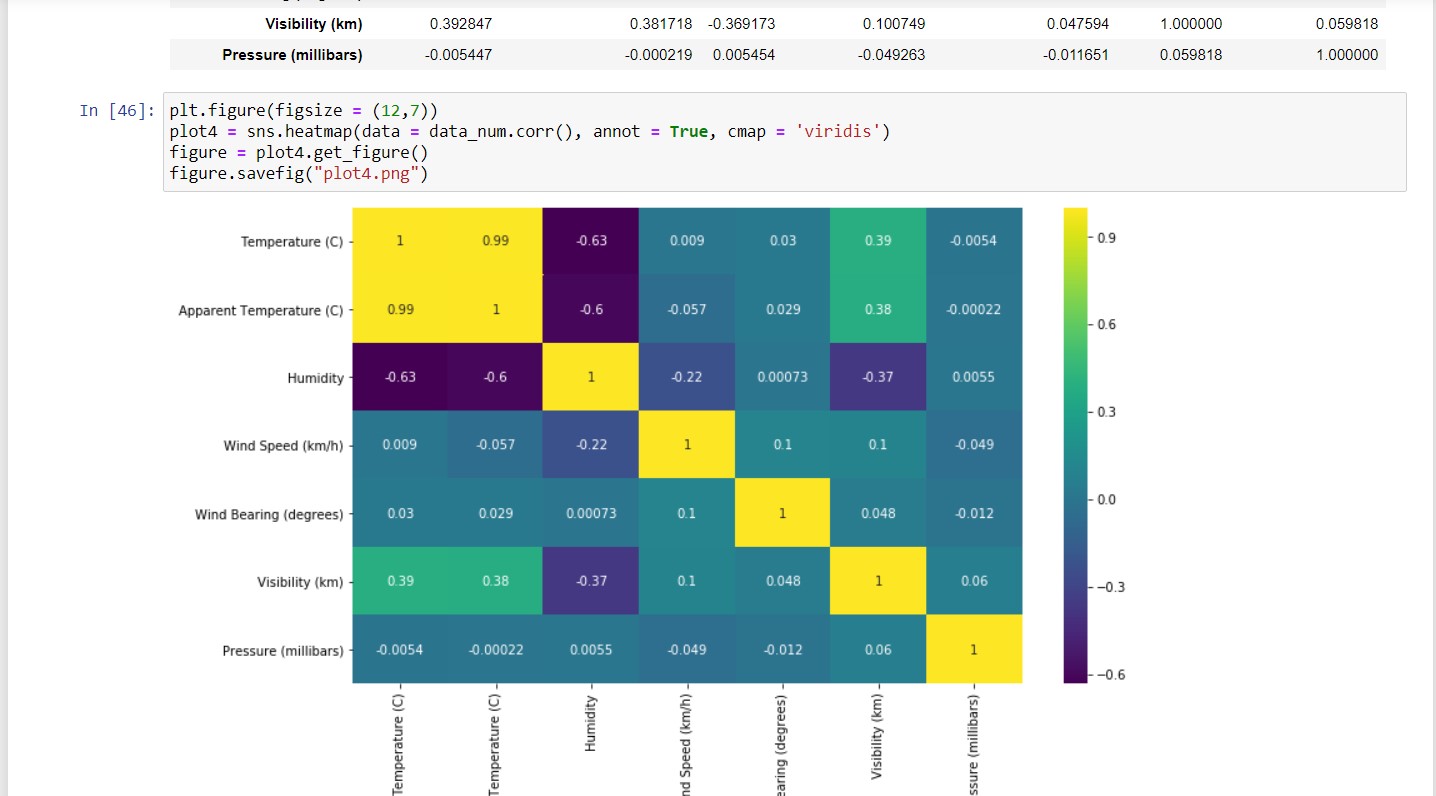
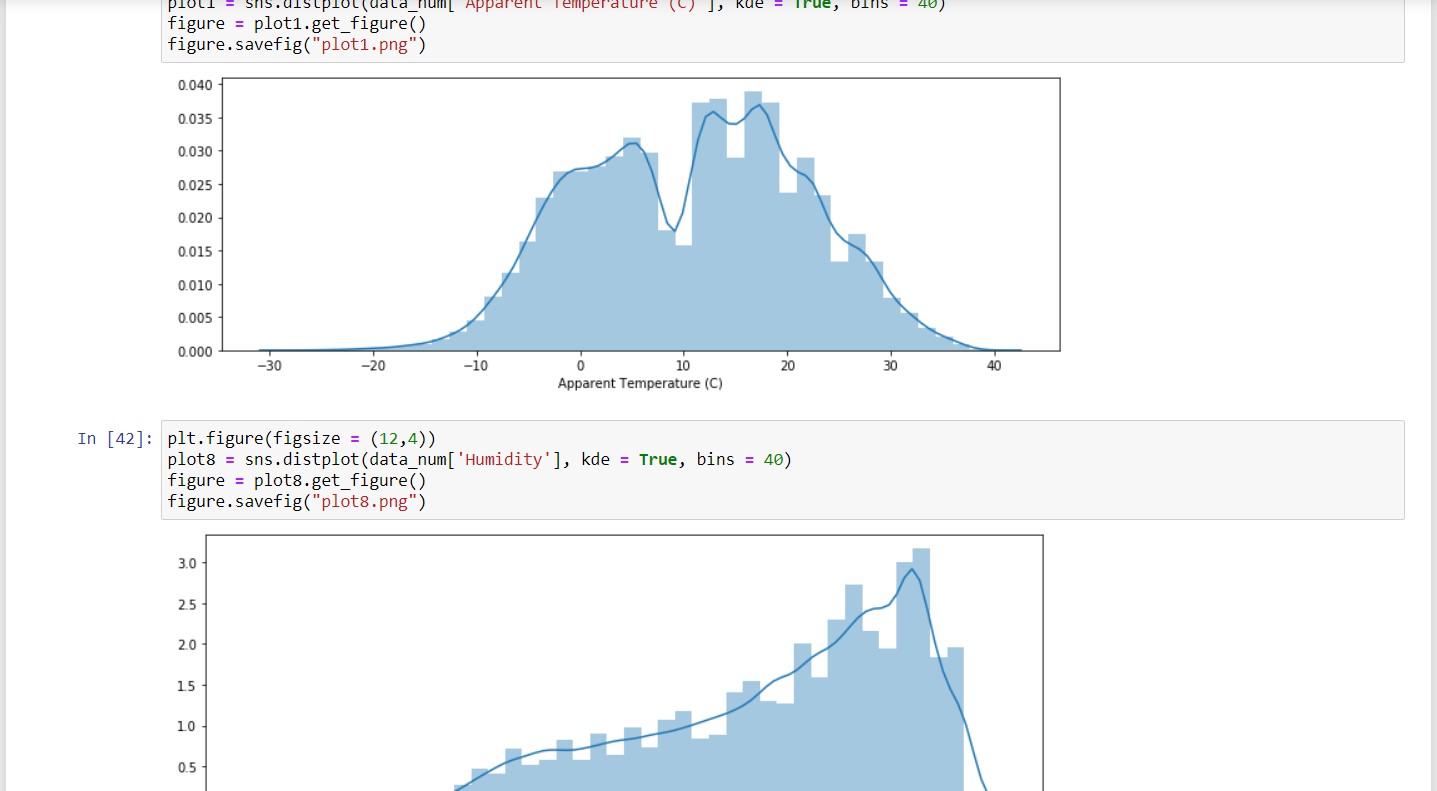
1. **Showcasing statistical summary of the data:**

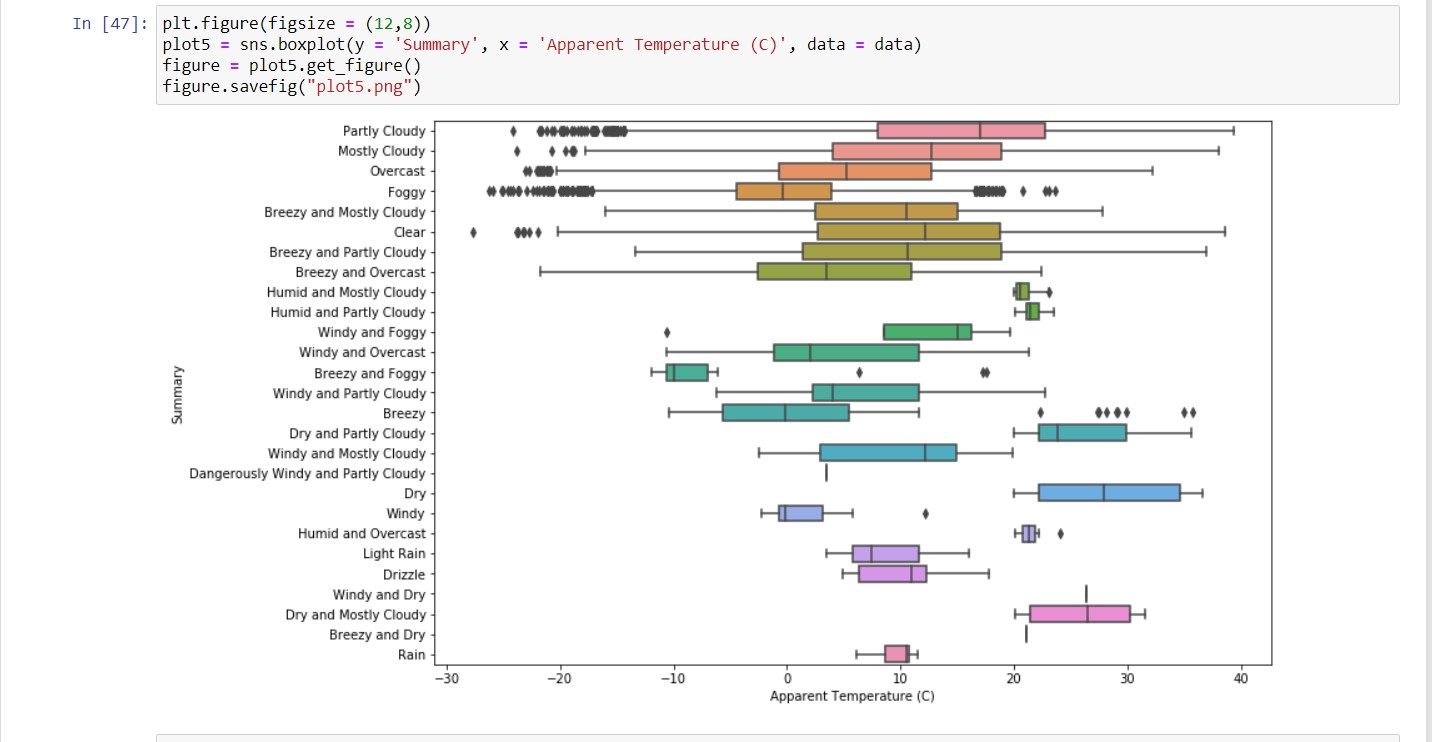
‘describe()’ function is used to get statistical summary of the data.



1. **Data Visualization using seaborn plots:**

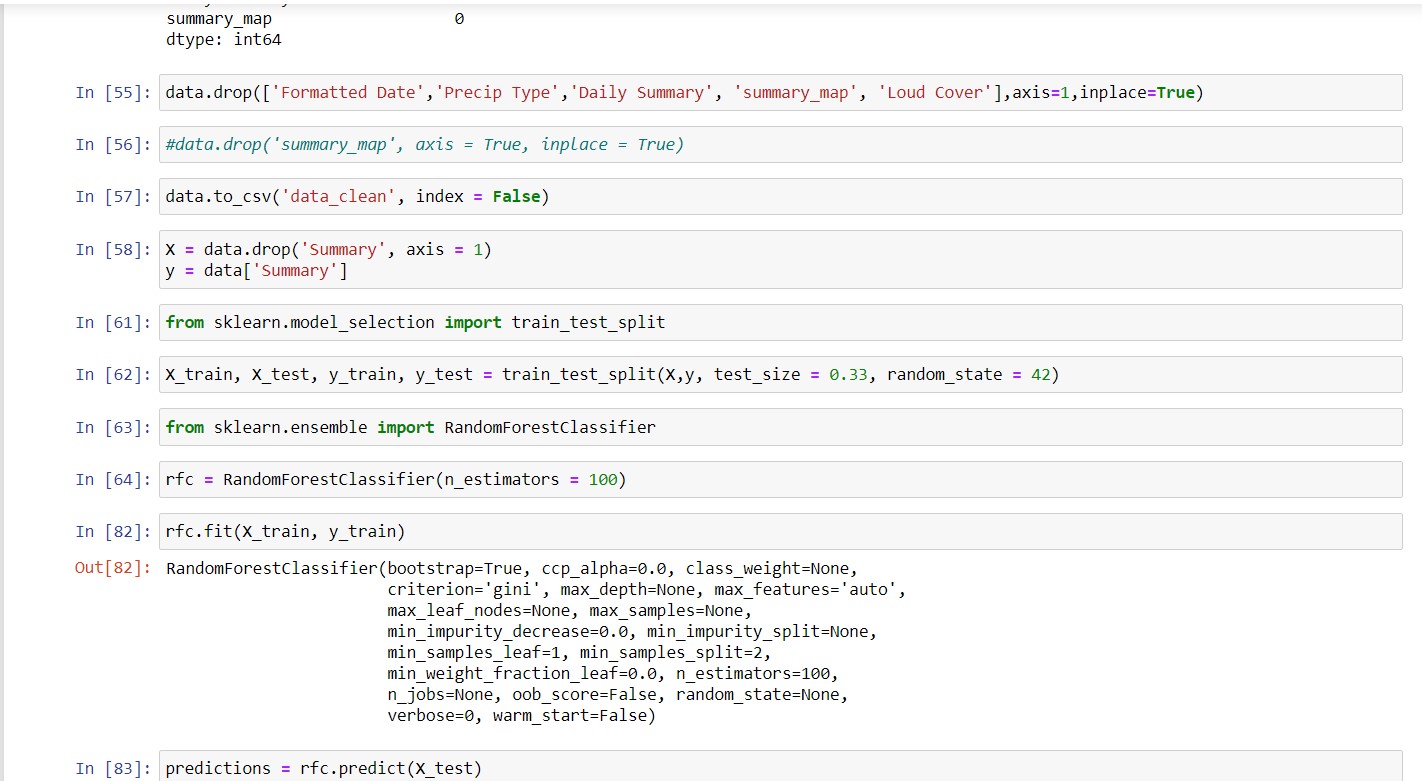
Distribution plot, pair plot, box plot and heatmap plot from seaborn library was used to plot various graphs for data visualization and finding correlation among various attributes of the data.





1. **Cleaning the data, saving it and importing again using PySpark:**

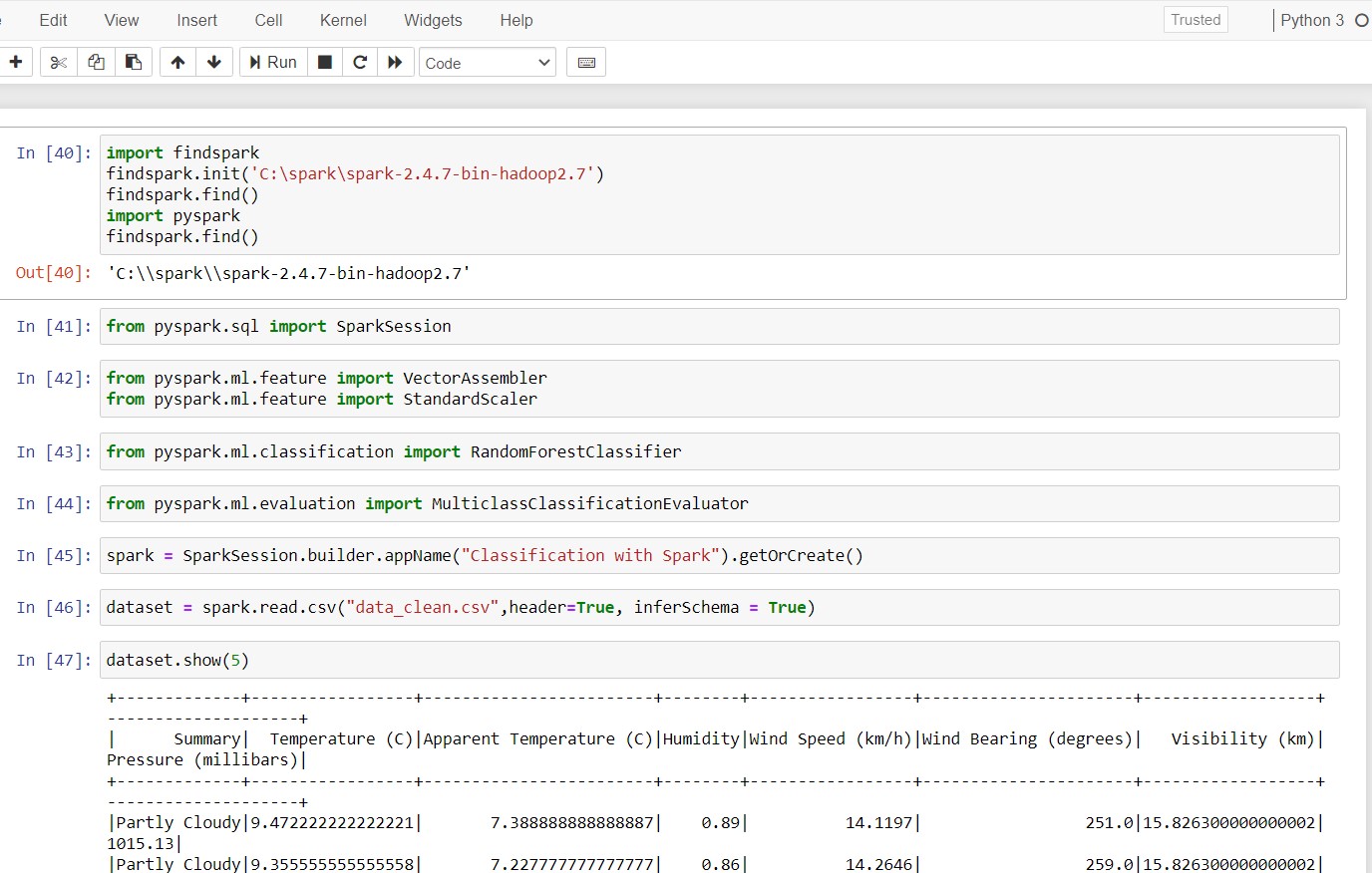
Redundant or not useful columns/attributes are dropped, one-hot encoded, mapped, or type converted. The cleaned dataset is then saved as a ‘.csv’ file and then imported again as a sql data frame using PySpark to use directly as an RDD. For this a PySpark session is first created and all required libraries for data manipulation as a sql data frame or RDD are imported.



1. **Manipulation of imported clean data:**

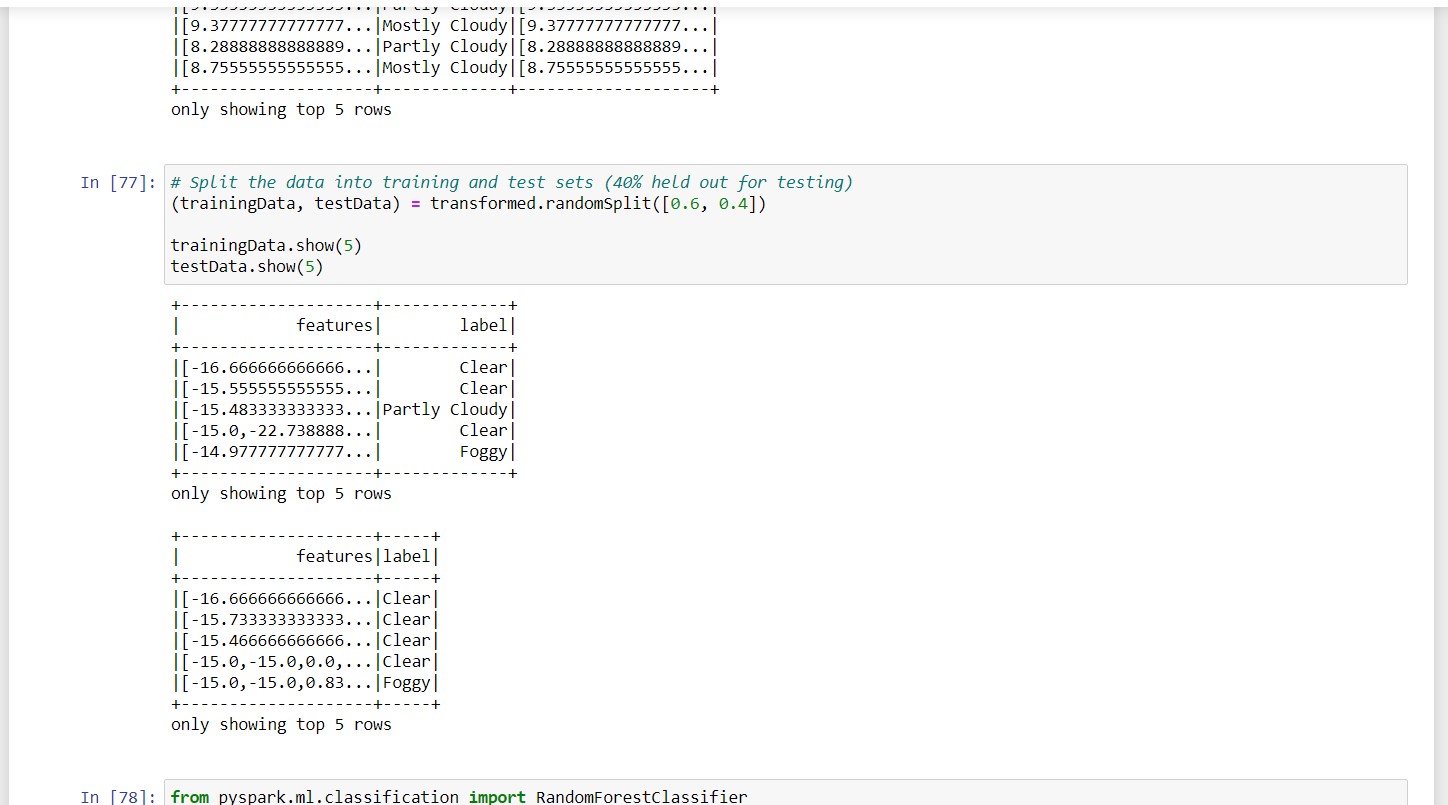
The imported data as an sql data frame is again checked for its data type using

‘printSchema()’ function. PySpark Machine Learning does not accept string type target class. So, the ‘Summary’ column is mapped to numerical values. All of the other attributes are checked if required mapping, encoding or type conversion.



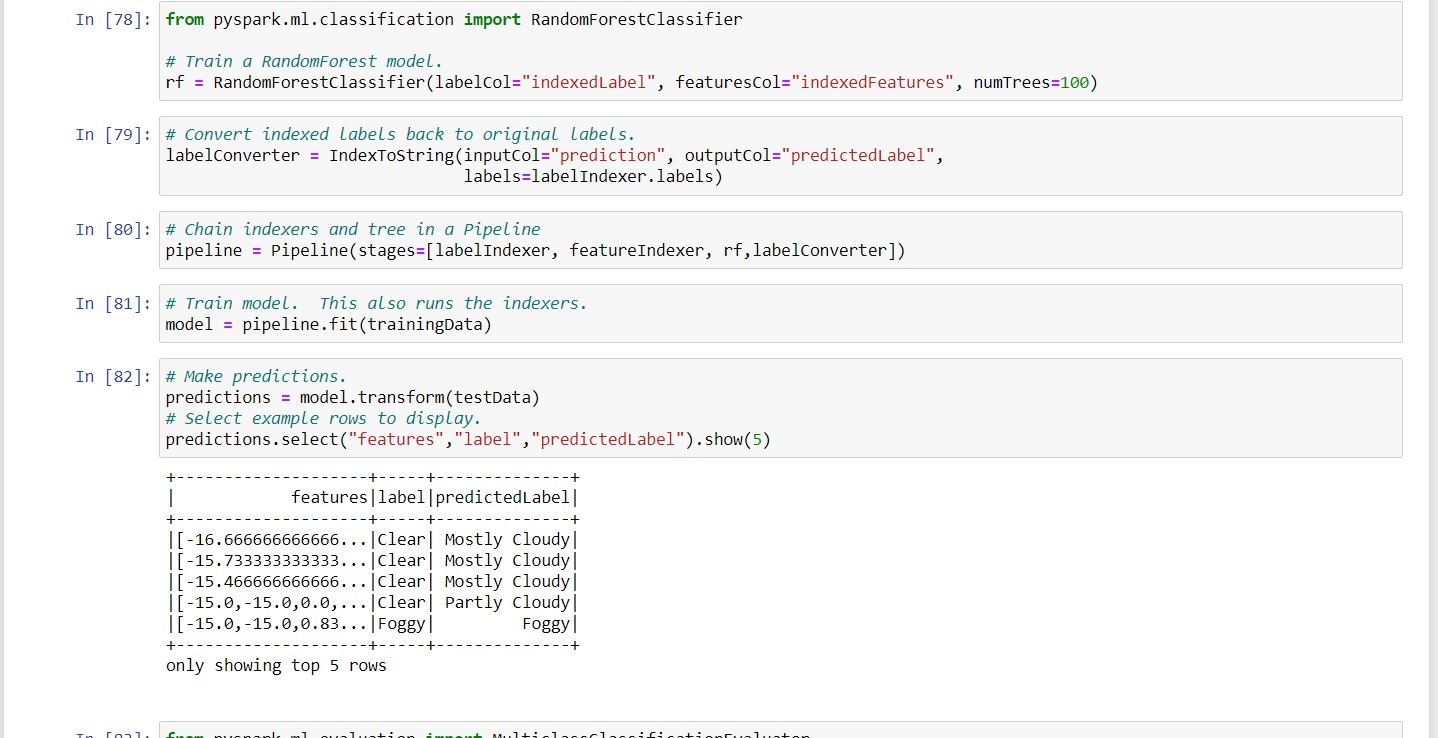


All parameters to be fed to ML model for training are then converted to vector or list types and put into ‘features’ column. The target attribute to be predicted is put into ‘label’ column and mapped to numerical values.



1. **Split final cleaned data into train and test data set and converted to RDD object:**

The data frame ready to be fed into the prediction model is split into train and test sets. The sets are then converted to RDD objects for MapReduce given to the Random Forest Classifier.



1. **Implementation of Random Forest Classifier and model validation:**

Random forests or random decision forests is an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification,](https://en.wikipedia.org/wiki/Statistical_classification) [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks that operate by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time and outputting the class that is the [mode](https://en.wikipedia.org/wiki/Mode_(statistics)) of the classes (classification) or mean/average prediction (regression) of the individual trees. The model is included in a pipeline object with other data cleaning procedures and trained by giving train dataset. Then test dataset is fed to get prediction and validation of our model. The accuracy achieved is approximately 58% with 100 trees. Fq score achieved was 0.58.

**GITHUB REPOSITORY LINK:**

**CONCLUSION**

After development and deployment of the proposed project model, it can be concluded that the model was able to do successful implementation of “WHAT HAS HAPPENED” predictive analysis.

Also given certain parameters with values, prediction of what might be the weather class can be done.

**REFERENCES**

* [https://spark.apache.org/docs/latest/ml-classification-regression.html#logistic-regression](https://spark.apache.org/docs/latest/ml-classification-regression.html)
* <https://spark.apache.org/docs/latest/ml-pipeline.html>
* [https://www.datacamp.com/community/tutorials/apache-spark-tutorial-machinelearning#basics](https://www.datacamp.com/community/tutorials/apache-spark-tutorial-machine-learning)
* [https://www.nbshare.io/notebook/187478734/How-To-Read-CSV-File-Using-PythonPySpark/](https://www.nbshare.io/notebook/187478734/How-To-Read-CSV-File-Using-Python-PySpark/)
* <https://databricks.com/session/deploying-machine-learning-models-with-apache-spark>