***An end-to-end Machine Learning model built for Rainfall Prediction-Weather Forecasting for the individuals new to this field of Data Science***

*“Kindly spare me if the article is lengthy, I have tried covering all the requisite and relevant information that an individual new to this field will require”*

This article is a synopsis of and end-to-end Rain Prediction-Weather Forecasting ML project and will be a guiding light for individuals new to the field of data science. Personally, the initial phase of my journey into the field of data science was somewhat disoriented. There is a enormous amount of information present on the world wide web. This information being available in odds and ends would leave one disoriented. All of us need that one bright and shiny guiding light that will pilot us to that polestar, to our destination. There are countless projects already available on the web, the only problem being, they are pretty advanced and will terrify an individual. This in turn would make that individual loose interest in this flourishing field of data science. I would like to contribute my bit of work with the hope that it would help an individual to face Data Science and ML head on rather than running away from it, petrified.

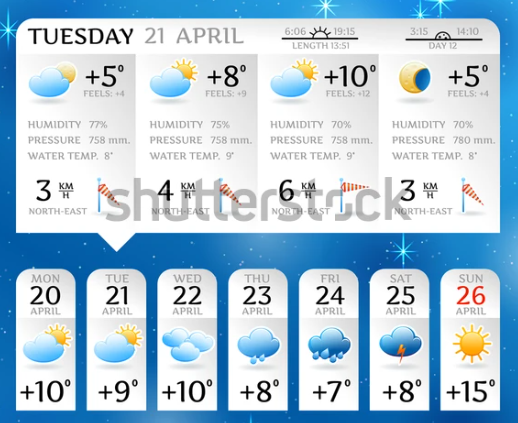
This article contains a detailed synopsis of the Rain Prediction-Weather Forecasting model. I will be further covering all the requisites through:

1. Problem Definition
2. Data Analysis
3. EDA
4. Pre-processing Pipeline
5. Building Machine Learning Models
6. Concluding Remarks
7. ***Problem Statement***

The foremost thing before building a Model is to understand the problem statement. One cannot afford to misinterpret a problem statement as the outcome would be a completely futile model.

I have used the rain prediction-weather forecast model which is formed using the rain in Australia Dataset. This dataset contains daily weather observation from various weather stations in Australia. The dataset for rain in Australia can be found [here](https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package) . The rain in Australia data set contains the daily weather observations namely:

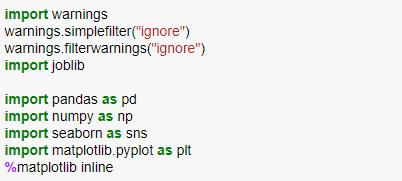
* Date - The date of observation
* Location -The common name of the location of the weather station
* MinTemp -The minimum temperature in degrees Celsius
* MaxTemp -The maximum temperature in degrees Celsius
* Rainfall -The amount of rainfall recorded for the day in mm
* Evaporation -The so-called Class A pan evaporation (mm) in the 24 hours to 9am
* Sunshine -The number of hours of bright sunshine in the day.
* WindGustDi r- The direction of the strongest wind gust in the 24 hours to midnight
* WindGustSpeed -The speed (km/h) of the strongest wind gust in the 24 hours to midnight
* WindDir9am -Direction of the wind at 9am
* WindDir3pm -Direction of the wind at 3pm
* WindSpeed9am -Wind speed (km/hr) averaged over 10 minutes prior to 9am
* WindSpeed3pm -Wind speed (km/hr) averaged over 10 minutes prior to 3pm
* Humidity9am -Humidity (percent) at 9am
* Humidity3pm -Humidity (percent) at 3pm
* Pressure9am -Atmospheric pressure (hpa) reduced to mean sea level at 9am
* Pressure3pm -Atmospheric pressure (hpa) reduced to mean sea level at 3pm
* Cloud9am - Fraction of sky obscured by cloud at 9am.
* Cloud3pm -Fraction of sky obscured by cloud
* Temp9am-Temperature (degrees C) at 9am
* Temp3pm -Temperature (degrees C) at 3pm
* Rain Today -Boolean: 1 if precipitation (mm) in the 24 hours to 9am exceeds 1mm, otherwise 0
* RainTomorrow -The amount of next day rain in mm. Used to create response variable. A kind of measure of the "risk"



Weather forecasting or rainfall forecasting, specifically, has been around for donkey years. Specifically, rainfall forecasting can not only be beneficial in anticipating floods but also in the draught condition as well. Forecasting how much it will rain tomorrow or how much Rainfall is there can extensively help in the agricultural field as well. I have applied different machine learning algorithms to predict whether or not it will rain tomorrow.

In this project we will be forecasting, whether or not it will rain tomorrow on the basis of the features provided. We can catch sight of the fact that the data for anticipating whether or not it will rain tomorrow will hold binary natured data (Yes or No), we will apply different classification algorithms to tackle this.

First and foremost, lets us import all the basic libraries required for building our model. I will be importing libraries as and when required.

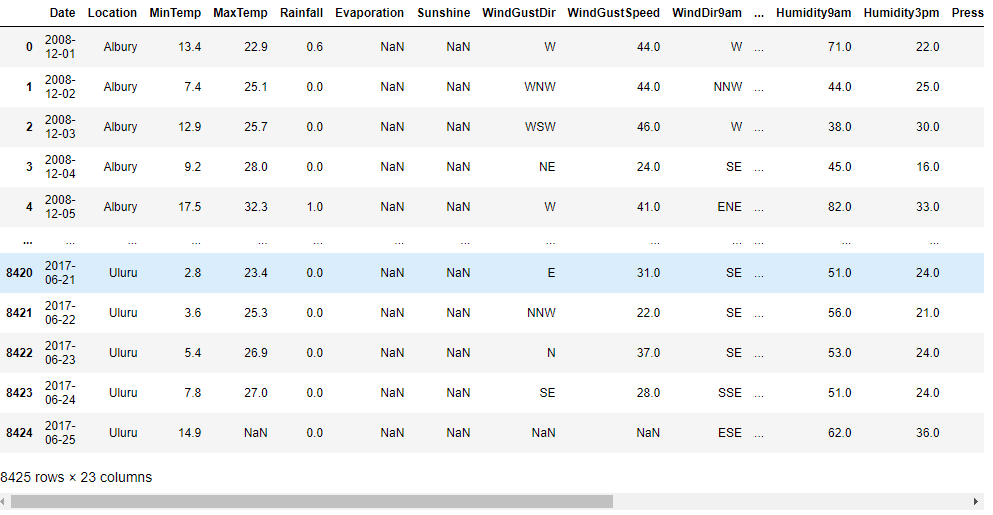


Loading our raw data directly from GitHub



1. ***Data Analysis***

We can glance at our data frame and try to make some sense of the columns. Acknowledge the type of data present in different columns. Identify the features and target, the type of data they contain.

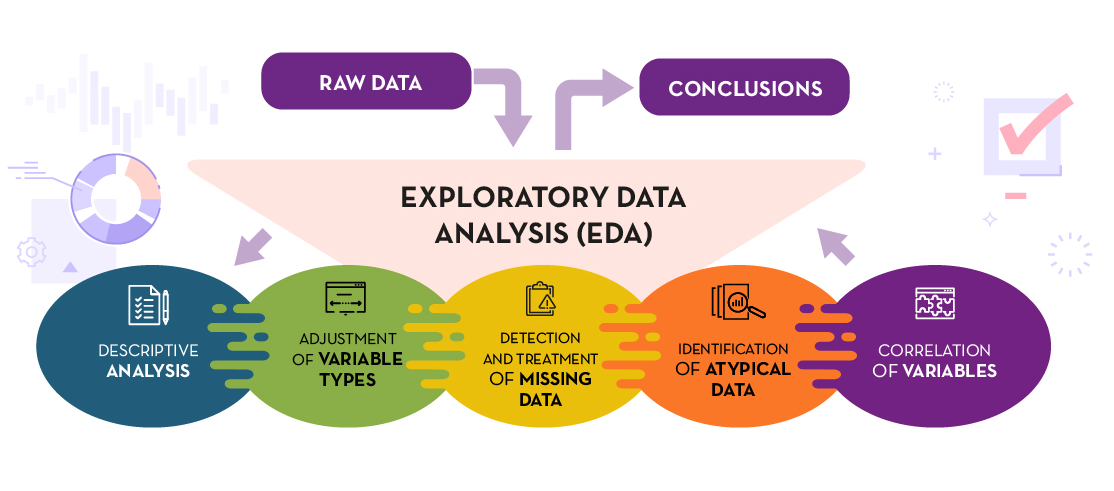


We have 8425 rows and 23 columns in our data frame. We can clearly see that some columns contain NaN values, which we will look into in detail while doing the EDA. We have a large data set considering 8425 rows times 23 columns. Since the dataset is large the visualization gets trimmed.

Since we have 23 columns, we can use the display.max function to display all the columns as well as the rows



1. ***Exploratory Data Analysis (EDA)***



Exploratory data analysis, commonly called as EDA is the most crucial step in building an accurate model. It is similar to an investigation, whether it be an investigation of a crime or a scrutiny leading to any particular unravelling of a mystery. Just as we scrutinize and use findings in order to get close to an unravelling, similar is the case of EDA. We scrutinize our data and manoeuvre these findings to build an accurate model.

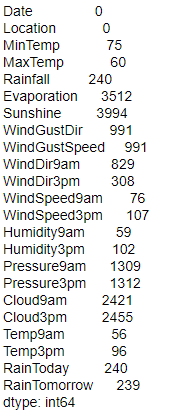
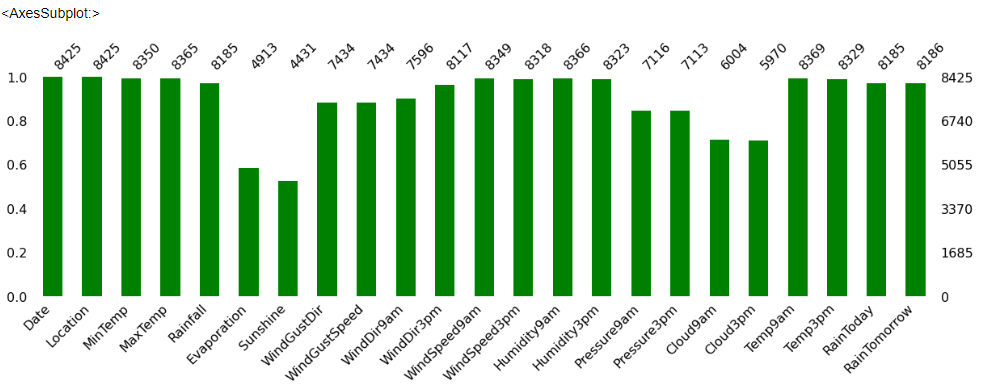
After going through ample articles by expert Data Scientists I can clearly sum it up by saying EDA is an approach and not a technique, it is an approach on how data analysis should be carried out. The various steps like descriptive analysis, adjustment of variable types, detecting and treating missing data, finding anomalies and correlation of variables are all approaches that aid in building an accurate model.

Another thing to note is that doing an EDA will get you familiarized with the data, help you understand the data and provide you with insights that will help you to attain that accuracy in your model.

I can go on and on about EDA but wouldn’t want you to get bored down with just EDA. Let’s move on to the code

The first thing I want to do is to see if there are any null values present in our dataset using the df.isnull().sum() and visually using the missingno.bar



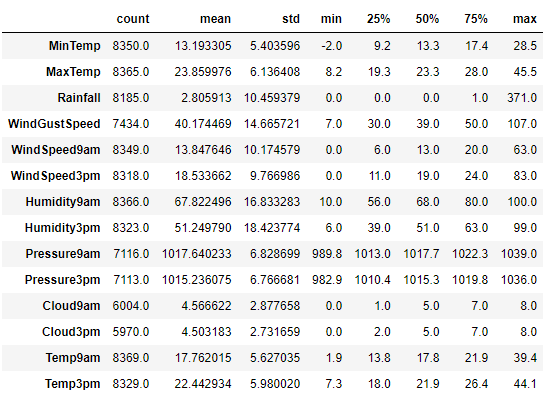
 

We can see there are a lot of null values present in some columns, which will have to be taken care. The columns evaporation and sunshine had almost half of the rows filled with null values, it would not make any sense to fill half of them, therefore I have dropped them.



Now, let us describe our data set using the describe function to check out the count value, mean data, standard deviation info and the 25%,50%,75% quartile details and minimum and maximum values. The describe function relates only to the numerical data, the object data type is ignored. Let us have a look at the code to understand better.



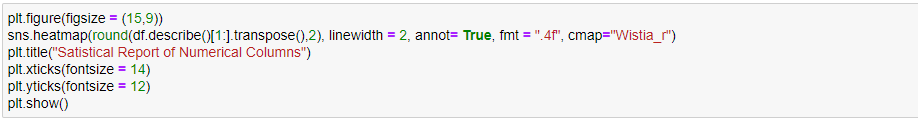


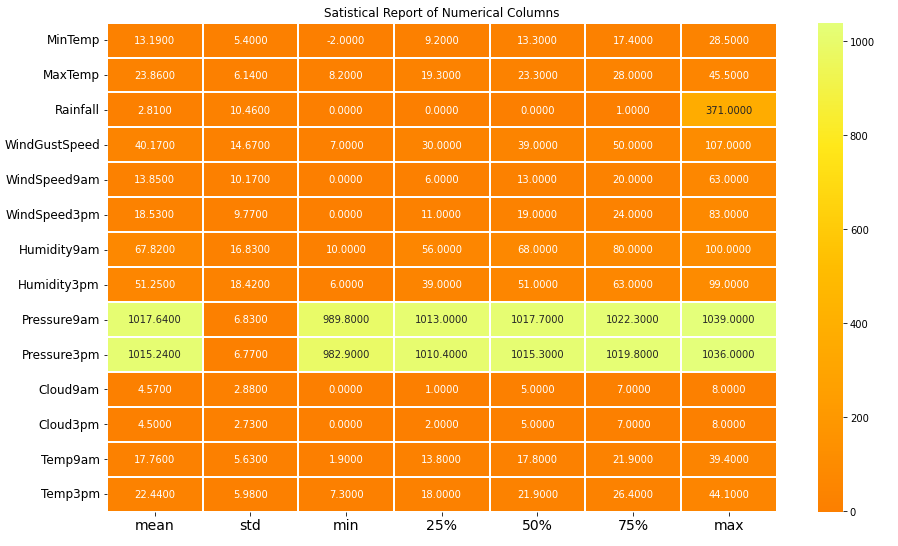
We observe that the mean value of the columns is almost equal to or slightly less than the median, 50% quartile

We can confirm again with the count values that null values are present

We see that some of the columns have the min value as 0 present in them

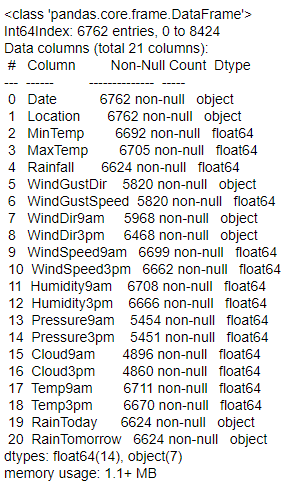
Let us describe the data frame visually also. The human brain is programmed in such a way that we learn and understand things better when we can visually see them



The pressure9am and pressure3pm columns stand out because of their high values

After taking insights from the description of the data frame let us move onto checking the data types. Whether the columns are of object datatype or float or int.

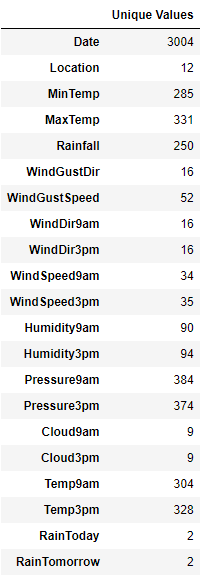




Some of the columns are of object datatype which will have to be converted to numerical datatype in order for the machine to understand it. We can confirm again that null values are present in some of the columns

Another technique that can help us in getting insights, about are independent variables or features, is checking for the unique values which can be done with the help of. nunique function

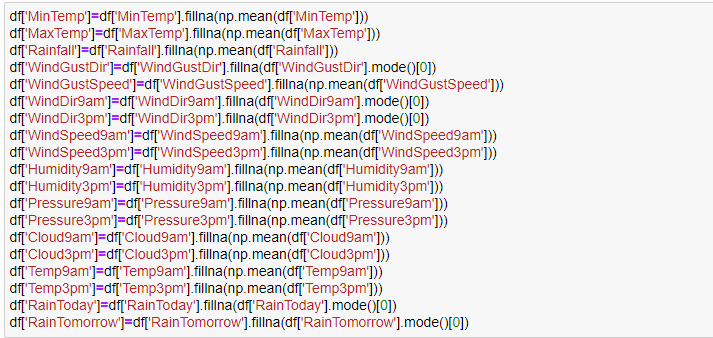




Coming to our target variable Rain Tomorrow we see that it has two unique values, yes or no and hence contains binary data and will be built using classification algorithms

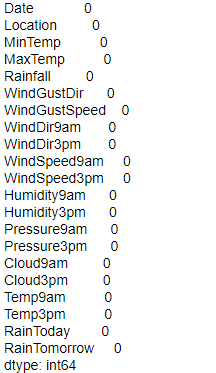
Before we get to the visualization of our data let us fill the missing data or else our model will be biased towards certain columns.

We can fill the missing values either with the mean value or with the mode value depending on the type of data present. We fill the missing values with mean if the data is continuous in nature and with the mode if it is categorical. The fill.na function is used to fill the missing values

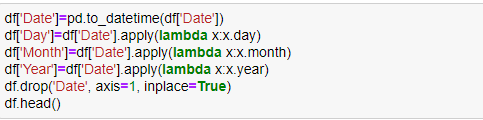


Let us cross check using the df.isnull().sum() function to check if the values have been filled or not

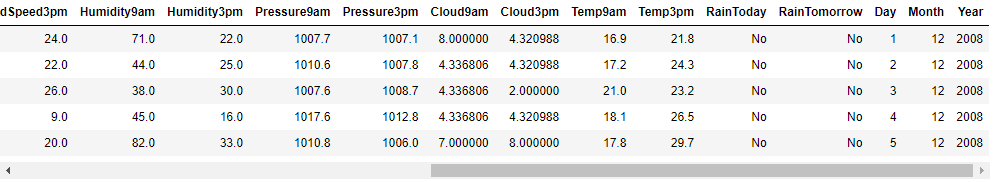




Moving on, I will be separating the data column into separate day, month and year columns by applying lambda and then dropping the Date column.



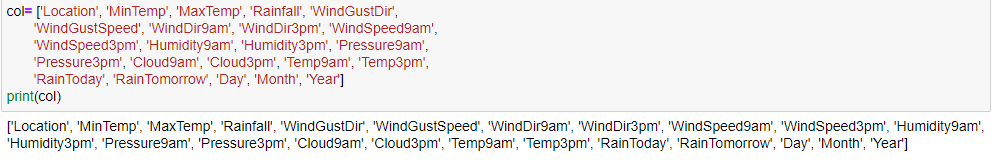
Let us have a look at our data frame



The number of columns also increase but that shouldn’t be of any trouble.

Moving on to a very cardinal step in EDA, visualization. The way in which one is able to visualize the data and gather insights tells a lot about how good one is at analysis. The analytical skills vary from person to person and there is no substandard analysis. In the end all that matters is what your insights are and how well you are able to put it across to the target audience.

I have made a list for all the columns

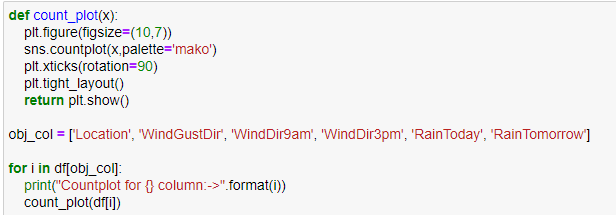


I will be visualizing the data and will be using different plotting techniques for different data

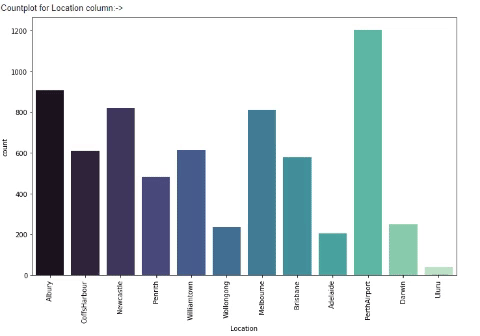
Let us begin with most basic plot, count plot.

Count plot for categorical or object data type columns.

I have written a function named count plot for the style and specifications of the count plot. I have then run the object data type list through a for loop.



Let us check the output



Scatter Plot for continuous columns

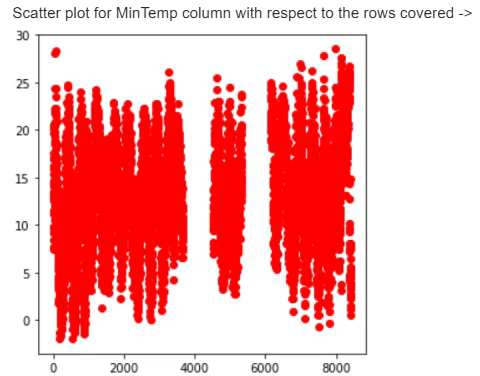
Here I have made a list, using the col list that I had made earlier, and stored it with columns that are not in the obj\_col



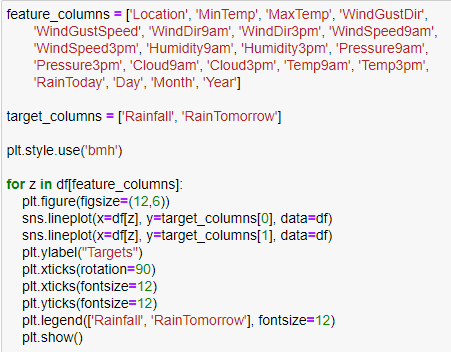




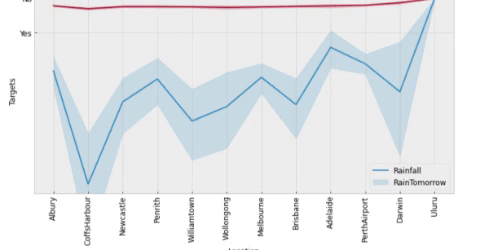
Let us look at the outputs now



Line plot for all the features with the target variable (RainTomorrow)



Let us have a look at our output and try to gather as much insights as possible



We have visualized our data and tried to gather as many insights as possible. We have had a look at the description of our data frame, identified and filled null values, visualized the data. One important thing that we gathered from visualization is that our classification target variable ‘RainTomorrow’ data is imbalanced, which will be taken care of later in pre-processing using SMOTE. These are some of the challenges that one faces and needs to take care of before even thinking about building a ML model. There is this very common saying, practice makes one perfect and the more one practices, the more hands-on experience you have, the easier it will be to overcome these challenges.

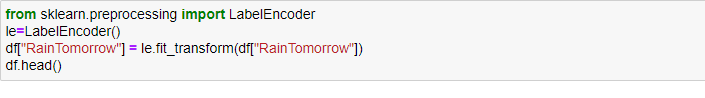
I have tried my best to cover everything with the visualization and any suggestions on anything I have missed out on or could have used a different approach, are most welcome.

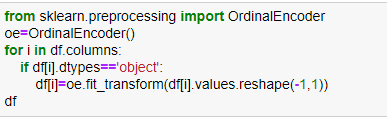
1. ***Pre-processing Data***

In the pre-processing step I am going to tackle all the miss fits and fix them one by one starting with encoding the object datatype values so that our Machine Learning models can understand the converted numeric values. I am making use of the encoding methods to convert all the object datatype values. For our label I am using Label Encoder while for other categorical feature columns I am using the Ordinal Encoder. Instead of the Ordinal Encoder I could have used the One Hot Encoder method but it would increase the number of columns, whereas applying Ordinal Encoder on data values would offer an order that deems to be a better option to me. I have seen quite a lot of people that use Label Encoding on the feature columns as well. It does not make any sense to me since the name itself suggests Label Encoder, and that it is use for encoding the label(s) columns. Hope my readers would keep this in mind and take it positively.

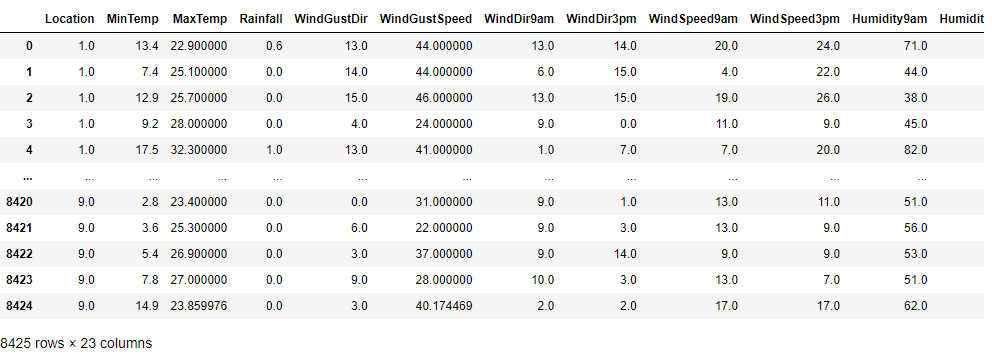
Code:

Label Encoding

Ordinal Encoding

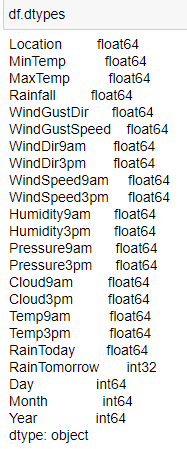


Output



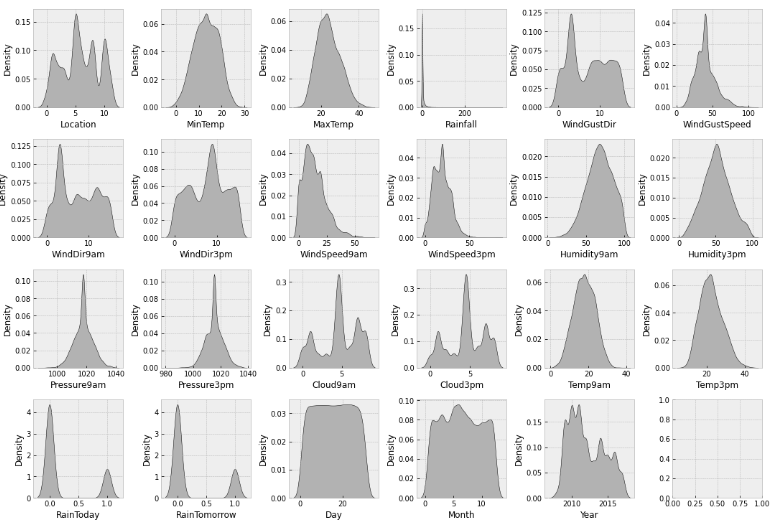
After encoding all our data, I will now plot the Normal Distribution curve: to check the

skewness and Boxplot: to check for any outliers.



Normal Distribution Curve

With the shape of the normal distribution curve, we can make out whether or not positive or negative skewness is present in our data. A perfect bell curve indicates that no skewness is present

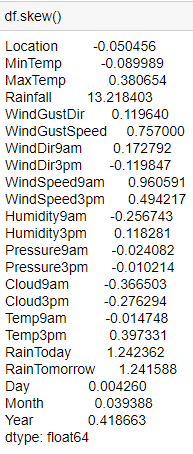


We can see that skewness is present in some of the columns. Let us use the df,skew function as well to check the values of skewness.

Code:



Output:

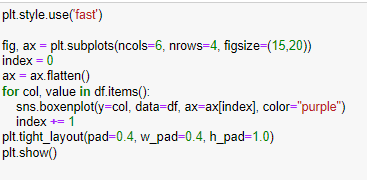


I will be using Log transform to fix the skewness, as the square root method is used to fix moderate skewness.

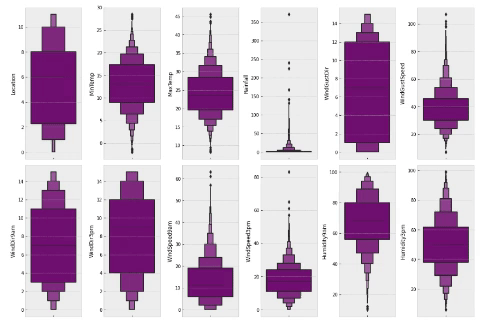
Now let us plot a box plot to check any outliers

Outliers are the data points that lie far away from the outer quartiles. These outliers need to be removed or else they will bring down the accuracy of our model.

Code:



Output:

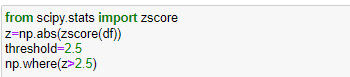


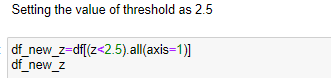
I will be using the z-score method to remove outliers. I can also use the IQR method but the data loss will be far greater and we want to retain as much data as possible

Let us remove the outliers now using z-score

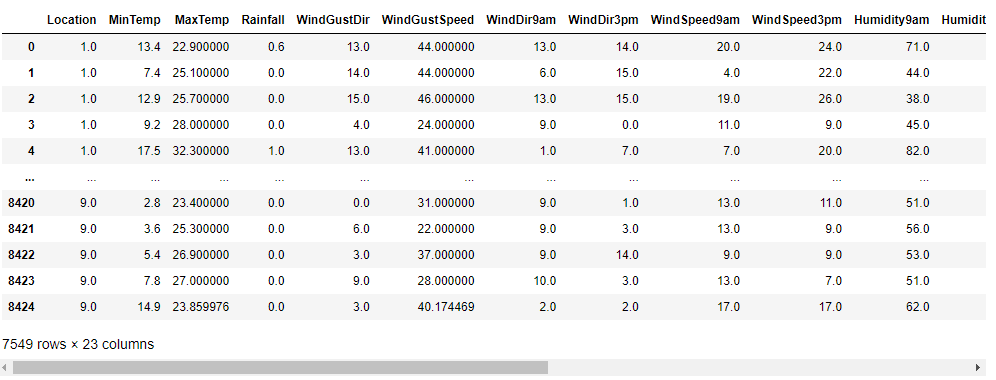


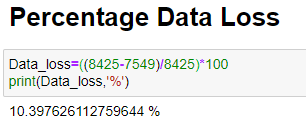
Code:





Output:

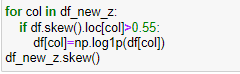




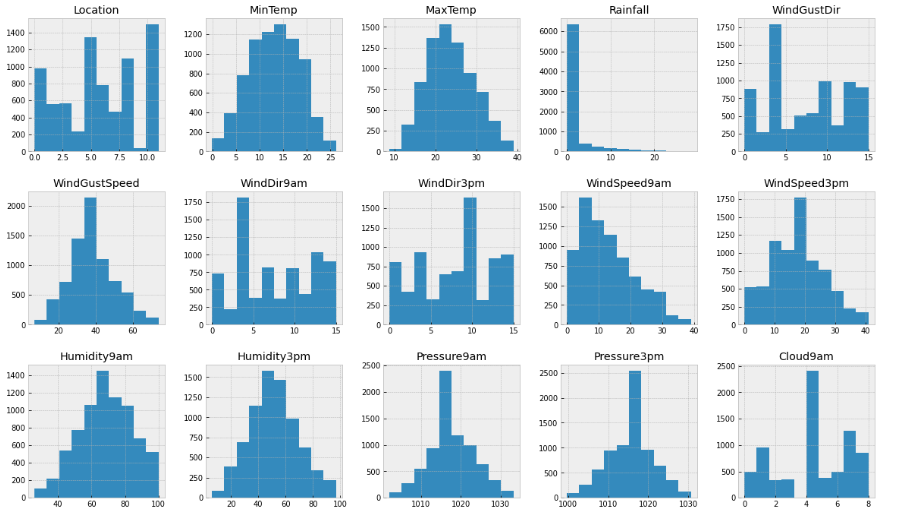
After applying Z score, I lost only about 10% of data but when I used the IQR method it took away almost 25% of the data and as a Data Scientist retaining valuable data and fixing it always takes priority rather than simply deleting it. Deleting it, is the last resort.

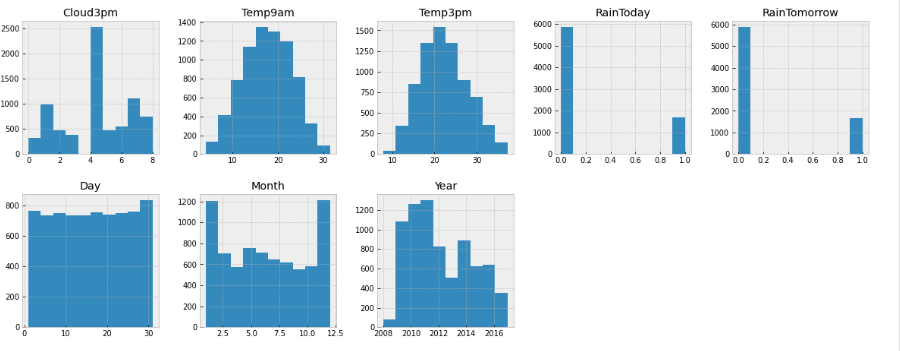
After this I am applying the Log transformation to deal with the skewness since the acceptable range lies between +/-0.5 value for each column.

Code:



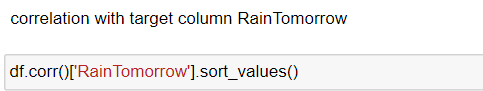
After removing the outliers and skewness I will now plot a histogram just to visually observe the changes in distribution.





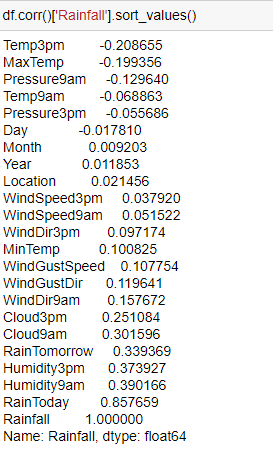
Now let us move onto checking the correlation between our target variables and feature variables

Code:



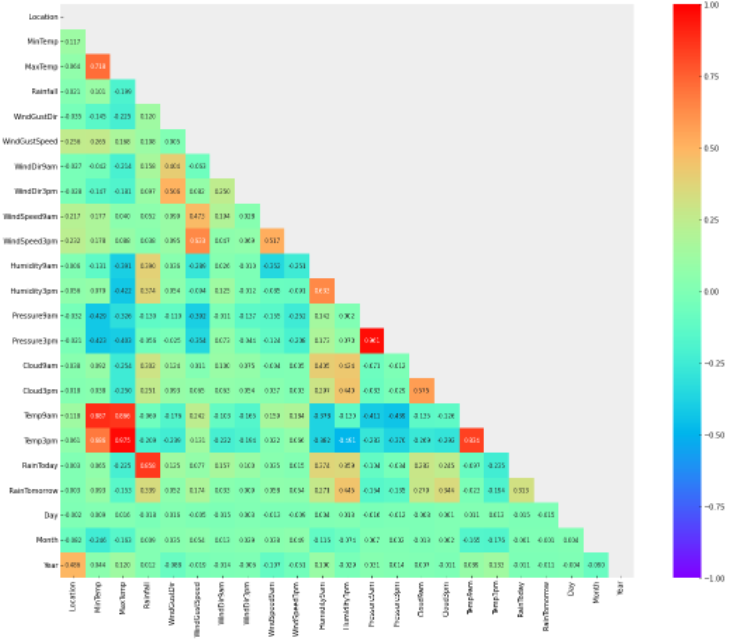
The sort\_values function will sort the values in ascending order

Output:



we see that there is only one high correlation value with RainToday, but I will retain it as it is an important feature in forecasting the RainTomorrow

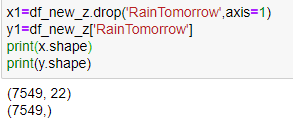
Let us understand the correlation with the help of a heatmap

The heat map is not clear but trust me it is not clear in my Jupiter notebook either. The advantage is that we can identify high correlation values based on their colour as they stand out easily.

Let us also check out the bar plot for our target variable with all the features which will help us to clearly differentiate between the positively and negatively correlated values.

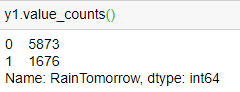


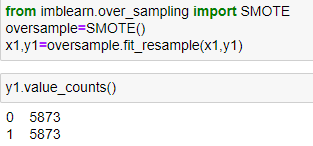
After taking care of the outliers, skewness and checking the correlation I will now split my data into features and target. For our target variable RainTomorrow (Classification Problem) I will split my data into features x1 and target y1

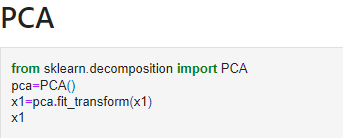




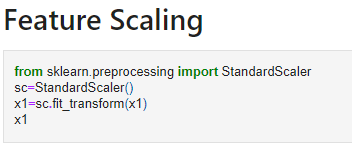
We had observed during visualization that there was an imbalance between the label classes. There was a huge difference between the “Yes” and “No” data. Therefore, I will have to resolve it as the imbalance can make our machine learning model biased towards the “No” value.



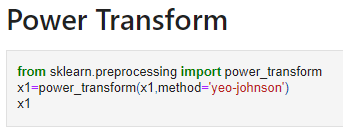




I have performed PCA to simplify the complexity in high dimensional data and to denoise it as well



Then I have scaled the feature columns that are stored in the x1 variable to avoid any kind of biasness over column values. Some integers cover thousands place and some cover hundreds or tens place, therefore it can make the machine learning model assume that the column with thousands place has a higher importance when actually that won’t be the case.



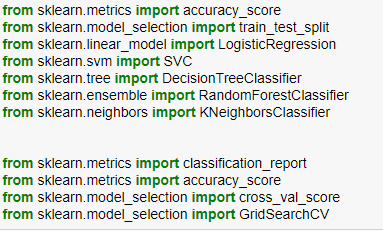
I have used power transform to increase the symmetry of the distribution of the features.

We have completed all the major steps involved in EDA and pre-processing pipeline that will assist us in building an accurate Machine Learning model.

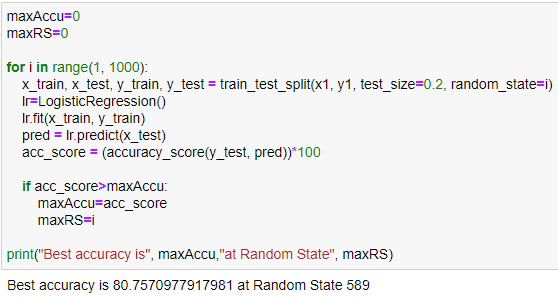
Now let us move onto building our Machine Learning Model

1. ***Building Machine Learning Models***

Let us import all the necessary libraries first

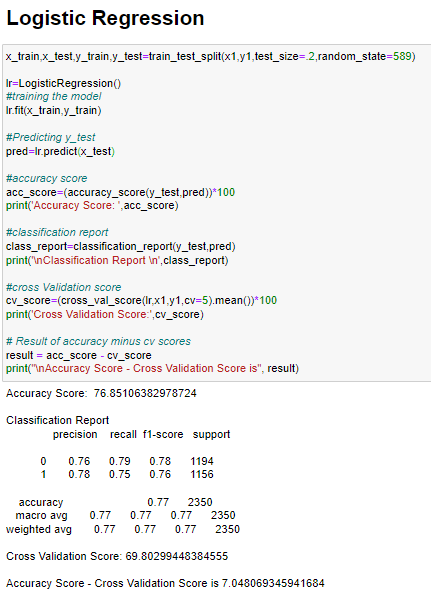


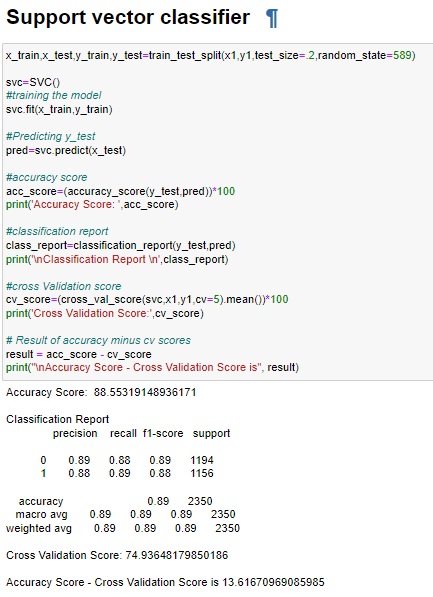
Before applying different machine learning algorithms let us find the best random state. I have divided my training and testing data, 80% for training and 20% for testing. There is no hard and fast rule to split the data, you have to use hit and trial to get the best accuracy at a random state.

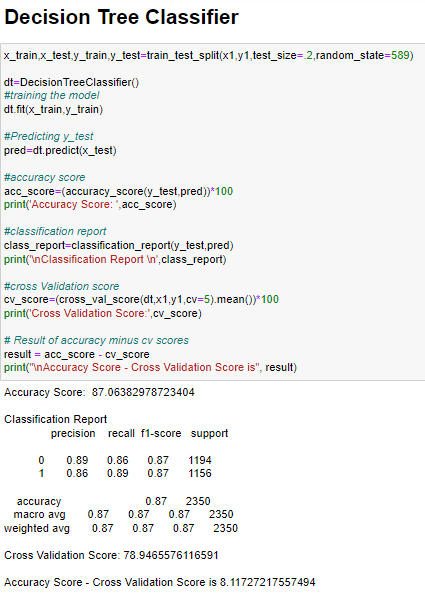


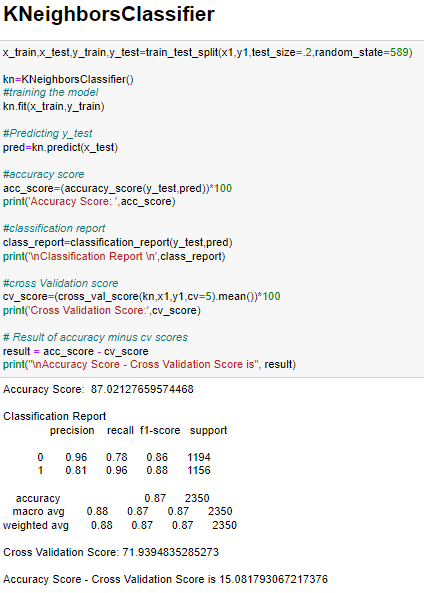
Now I will be choosing my random state value ‘i’ as 589

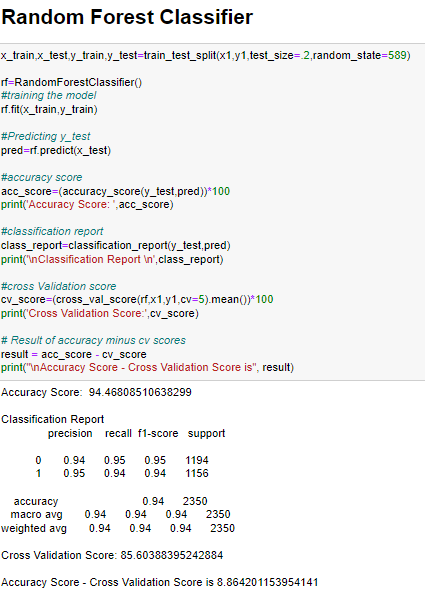
It is always best to build at least 5 machine learning models so that one can choose from the best performing model and then apply hyper parameter tuning to make it perform even better.



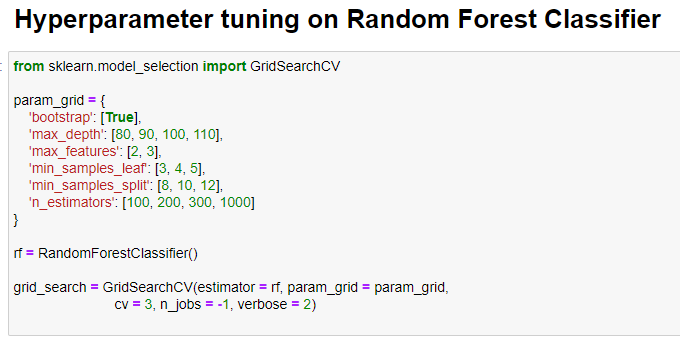








The best model is the one which gives us a good accuracy and at the same time has the least difference between the Accuracy Score-Cross Validation Score and that is why I have chosen Random Forest Classifier for hyperparameter tuning.



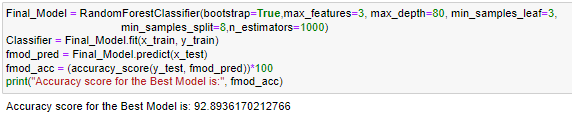
We can refer the sklearn webpage to get the parameters of any classifier



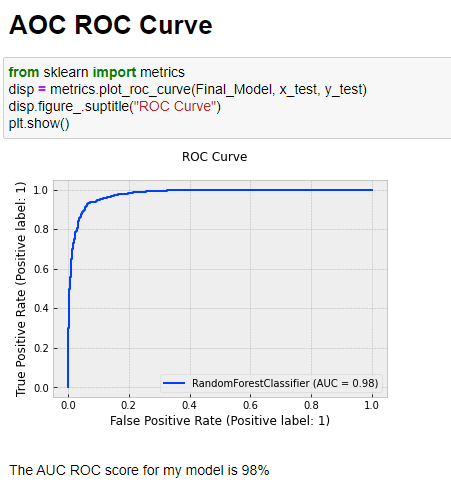




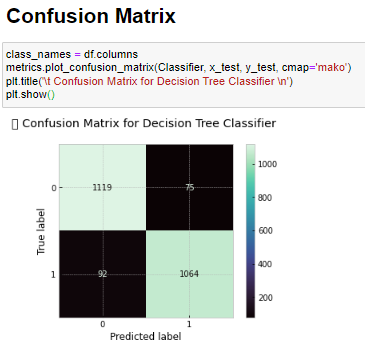
Now we just have to choose our best parameters to get the best accuracy.



I have also plotted the AUC ROC curve and the confusion matrix as well



The more area the curve covers the better the AUC score



From the confusion matrix we can see that our TP=1119, FP=75, FN=92, TN=1064

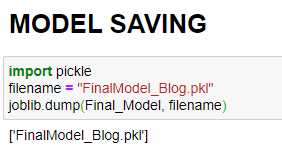
that means that are 1119 positive values were correctly classified

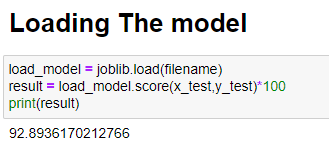
75 negative values were correctly classified

92 positive values were incorrectly classified

1064 negative values were incorrectly classified

Once you have gone through all the previous steps and are satisfied with the outcome you can then save the final model using either joblib or pickle. I have used the joblib method to save and then load my model from the same, saved as filename.





1. ***Concluding Remarks***

Let us have a quick rundown of all the steps that we have accomplished from the onset, from understanding the Problem Definition to performing the Data Analysis and EDA. We executed the necessary Pre-processing Data steps before building our final Machine Learning Model.  
  
In the initial phase of your journey into building ML models I would recommend to use a reference model for building your model. Having a look at multiple models and try to gain insights at what techniques and steps should be performed for what datasets. It will only come after practice and soon it’ll come to you naturally. The more you write your code yourself and try to gather maximum insights from every line of code, the better you will get at it. This will be beneficial for you in the long run. Build up your own unique model and then compare it with a good model on the internet, try to understand what could have made your model better, what steps are you lacking in or what steps have you missed. What plots should be used for what type of data, which techniques should be used in the pre-processing pipeline. Put a lot of effort into EDA, as it is the basis for building an accurate ML model, practice it again and again using different approaches.   
  
I would like to conclude by saying that nothing in life comes easy, you have to put in efforts from your end. Remember the time when you were struggling to walk and now it comes to you naturally. We can apply the same thinking to data science as well, you will have to struggle in the begin to enjoy the fruits of your effort later on.