**Temperature Forecast**

**Multi Output Regression**

**Machine Learning**

**By Sudha Udayakumar**

**Logo

Description automatically generated**

Table of Contents

[Temperature Forecast (Multi-Output Regression) Machine Learning 4](#_Toc73901536)

[1. Multi-Output Regression 4](#_Toc73901537)

[1.1 Types of Multiclass and Multioutput Algorithms 4](#_Toc73901538)

[2. Take Away from this Article 5](#_Toc73901539)

[3. Assumptions 5](#_Toc73901540)

[3.1 Check Scikit-Learn Version 5](#_Toc73901541)

[4. 9 Stages of Machine Learning 5](#_Toc73901542)

[4.1 Understanding the Problem 6](#_Toc73901543)

[4.1.1 Loading the Dataset from the source 6](#_Toc73901544)

[4.1.2 Summary of the Data source, Problem Statement 6](#_Toc73901545)

[4.1.3 Looking at the Data 6](#_Toc73901546)

[4.2 Initial Analysis of the Data 7](#_Toc73901547)

[4.2.1 Checking for Duplicate Rows 7](#_Toc73901548)

[4.2.2 Unique Values 7](#_Toc73901549)

[4.2.3 Type of columns 8](#_Toc73901550)

[4.2.4 Statistical Info of the Data frame 9](#_Toc73901551)

[4.2.5 Checking for missing Values 9](#_Toc73901552)

[4.3 Pre-Processing of Data 10](#_Toc73901553)

[4.3.1 Converting Date Column 10](#_Toc73901554)

[4.3.2 Handling Missing Values 11](#_Toc73901555)

[4.4 Exploratory Data Analysis 12](#_Toc73901556)

[4.4.1 Month Max/Min Temperature 13](#_Toc73901557)

[4.3.2 Min/Max Temperature across Years 14](#_Toc73901558)

[4.3.3 Co-relation- Target Variable(s) 14](#_Toc73901559)

[4.5 Data preparation for Model Building 16](#_Toc73901560)

[4.5.1 Handling Outliers 16](#_Toc73901561)

[4.5.2 Skewness Handling 16](#_Toc73901562)

[4.6 Model building 17](#_Toc73901563)

[4.6.1 Importing the required libraries for Model Building and Evaluation 18](#_Toc73901564)

[4.6.2 Splitting our data into X & Y 18](#_Toc73901565)

[4.6.3 Finding Best Random State 18](#_Toc73901566)

[4.6.4 Instantiating the Algos 19](#_Toc73901567)

[4.6.5 Algorithm & Metrics for Multi-output 19](#_Toc73901568)

[4.6.6 Defining the Function 20](#_Toc73901569)

[4.7 Evaluation Metrics 21](#_Toc73901570)

[4.7.1 Calling each algorithm and its metrics 21](#_Toc73901571)

[4.7.2 Unsupported Algorithms and Error 23](#_Toc73901572)

[4.7.3 Comparing All Algos & Evaluation Metrics 24](#_Toc73901573)

[4.8 Improving the Model 24](#_Toc73901574)

[4.8.1 Random Forest 24](#_Toc73901575)

[4.8.2 KNN 25](#_Toc73901576)

[4.8.3 Bagging Regressor- Grid Search 26](#_Toc73901577)

[4.8.4 Bagging Regressor- Random Search 26](#_Toc73901578)

[4.8.5 Comparing the Cross Validation with Hyper Parameter Tuning 27](#_Toc73901579)

[4.9 Saving & Loading the Model 27](#_Toc73901580)

[5.Conclusion 27](#_Toc73901581)

# Temperature Forecast (Multi-Output Regression) Machine Learning

## Multi-Output Regression

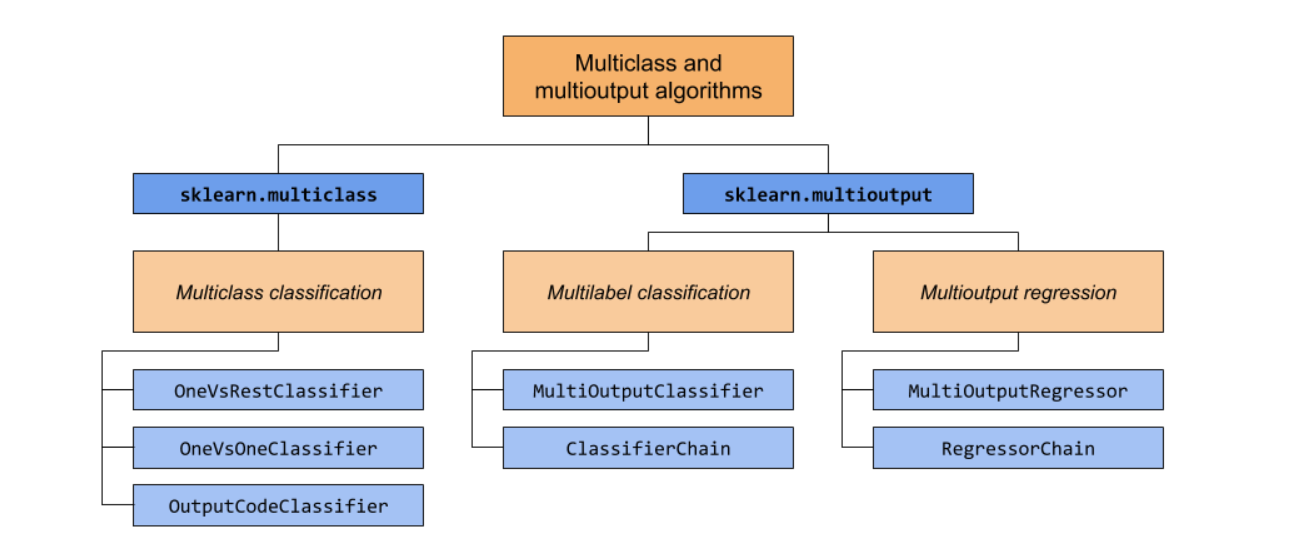
Multioutput regression are regression problems that involve predicting two or more numerical values

In multioutput regression, typically the outputs are dependent upon the input and upon each other. This means that often the outputs are not independent of each other and may require a model that predicts both outputs together or each output contingent upon the other outputs

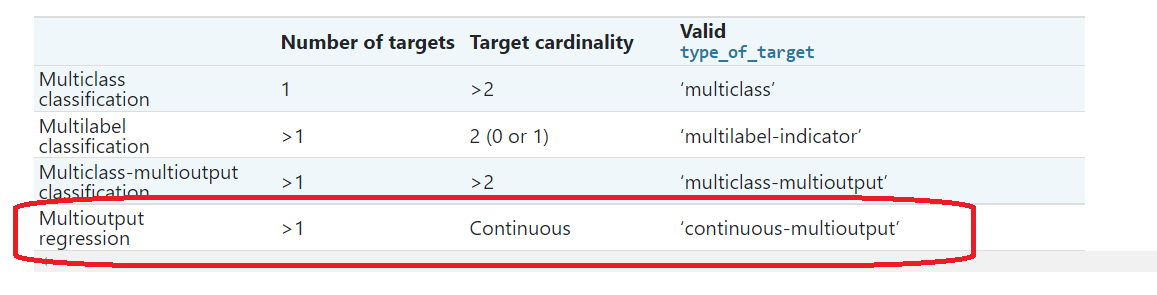
### 1.1 Types of Multiclass and Multioutput Algorithms

The chart below demonstrates the problem types that each module is responsible for, and the corresponding meta-estimators that each module provides.

<https://scikit-learn.org/stable/modules/multiclass.html>



The table below provides a quick reference on the differences between problem types



In this Article, lets focus on the Multioutput Regression

## Take Away from this Article

1. Handling a Multioutput Regression Dataset
2. Algorithms used in Machine Learning - Multioutput Regression Dataset
3. Evaluation Metrics used in Machine Learning - Multioutput Regression Dataset
4. Several Tips, to apply for future datasets

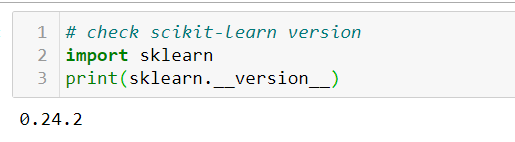
## Assumptions

We assume that you have a basic knowledge of Python, Data Science, Machine Learning and have a familiarity with the Sckit learn and know to load the libraries like Numpy, Pandas, Seaborn, Matplotlib for data loading, pre-processing and EDA. Also have a basic understanding of the various Regression based algorithms

### Check Scikit-Learn Version

Check the version of the library with the following code example:

This is important because some of the models we will explore in this article require the latest version of 0.22 version of scikit-learn or higher.



## 9 Stages of Machine Learning

The stages of Machine Learning are defined different by every user. We are going to follow these steps, in our article.

1. [Problem understanding](#_Understanding_the_Problem) -Understanding the problem, the various column names and type of data
2. [Initial Analysis of the Data](#_4.2_Pre-Processing_the) – Understand the type of data, check for missing data, type of data (numerical, categorical, Date) etc
3. [Pre-Processing the data](#_Pre-Processing_of_Data): Cleaning up the data, converting to format as needed etc, dropping unwanted rows & columns
4. Dataset understanding using [Exploratory Data Analysis](#_Exploratory_Data_Analysis) (EDA)
5. [Data preparation for Model Building](#_Data_preparation_for) - (We will be checking for outliers, skewness, Scaling)
6. [Model building](#_4.5_Model_building) (Use various Algorithms and build)
7. [Model evaluation](#_Evaluation_Metrics) (Use different Evaluation metrics and finalize which one we are going to use)
8. [Improve Model](#_4.7_Improving_the) (There is always, scope for improvement!)
9. [Saving the Model & Loading the Model](#_Saving_&_Loading)

### Understanding the Problem

#### Loading the Dataset from the source

Data Source: [Link](https://raw.githubusercontent.com/dsrscientist/Dataset2/main/temperature.csv)

#### Summary of the Data source, Problem Statement

This data is for the purpose of bias correction of next-day maximum and minimum air temperatures forecast of the LDAPS model operated by the Korea Meteorological Administration over Seoul, South Korea.

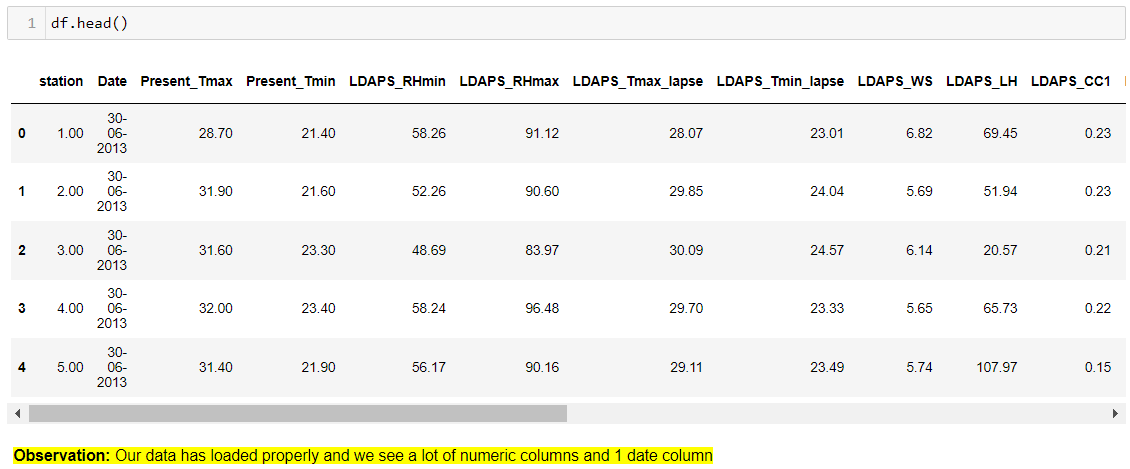
There are two target variables

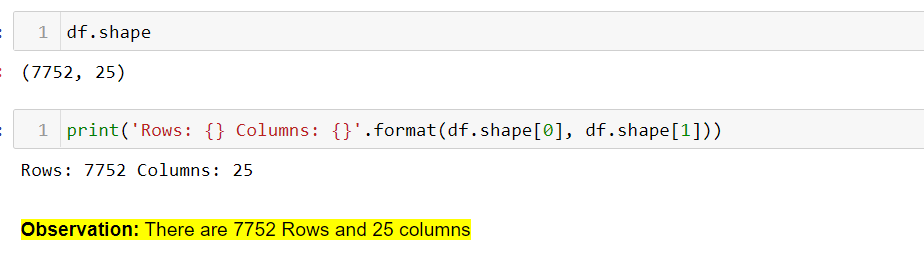
1) Next\_Tmax: Next day maximum temperature

2) Next\_Tmin: Next day minimum temperature

#### Looking at the Data

We have loaded data and df.head() gives us the top 5 rows of our Data frame.





We have also got rows and column with df.shape, and also printed statement of rows & columns count

Tip1:

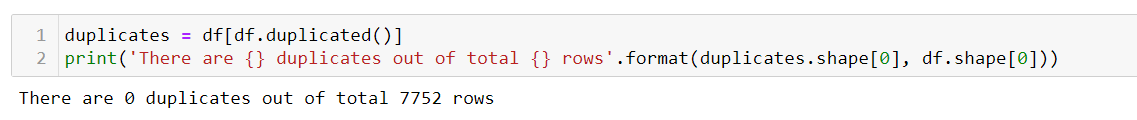
* Writing an Observation after each code or few lines of code, helps us to give a quick glance. Also helps when multiple people work on the same document
* For certain codes, we can even print the observation statement e.g., like how we have printed the rows and columns count as above!

### Initial Analysis of the Data

Before we start analysing our data, we need pre-process the data.

Pre-processing of data contains several methods; we will see few of them which are significant in our case study.

#### Checking for Duplicate Rows



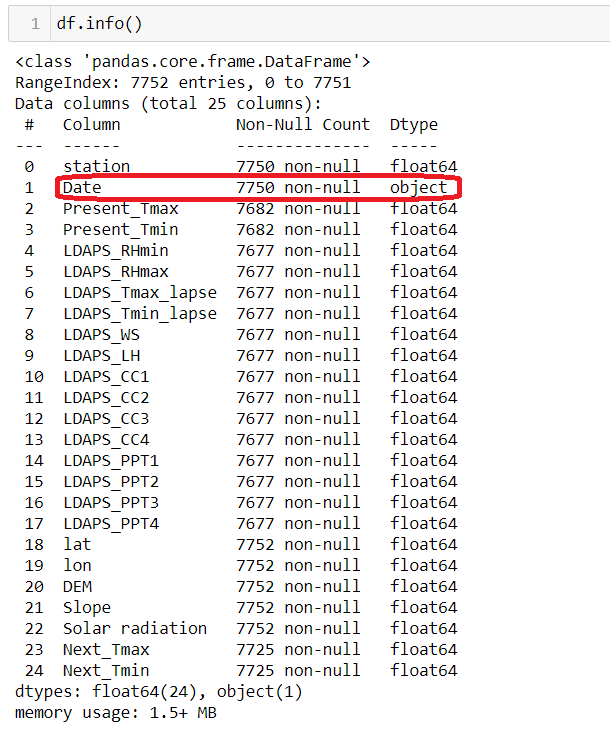
We see there are no duplicate values in our data.

#### Unique Values

As we see in the Observation,

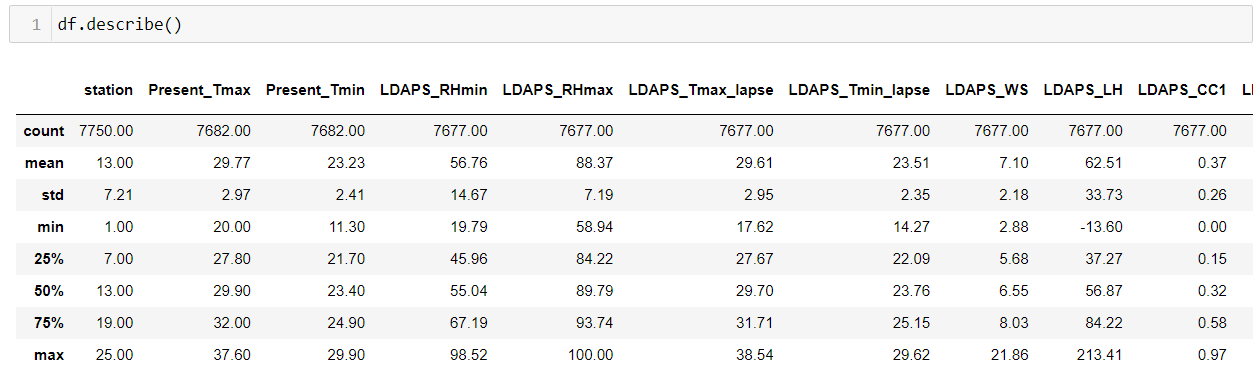
1. “All columns have more than 1 unique values. And no columns can be dropped from this unique data.

#### Type of columns



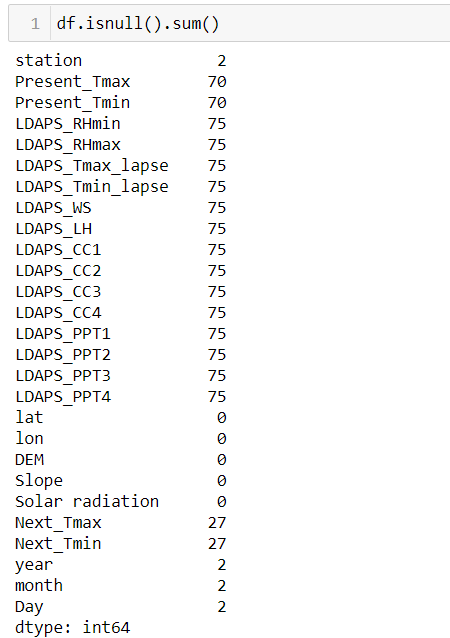
All columns are numeric in nature. However, there is one Object Column. And the column name is Date. So, Date has to be converted from Object to DateTime Format

#### Statistical Info of the Data frame



* Date is not listed, as it is listed as an Object column
* We see a lot of missing values as well

#### Checking for missing Values



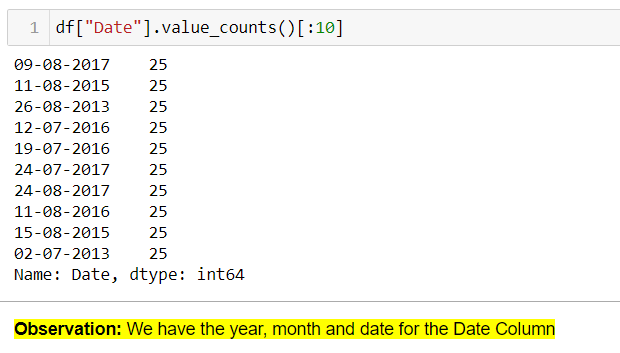
* We see many columns have missing data.

### Pre-Processing of Data

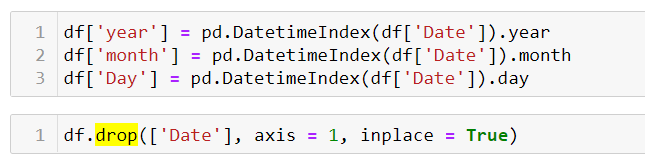
Here we handle the data, which we had studied earlier.

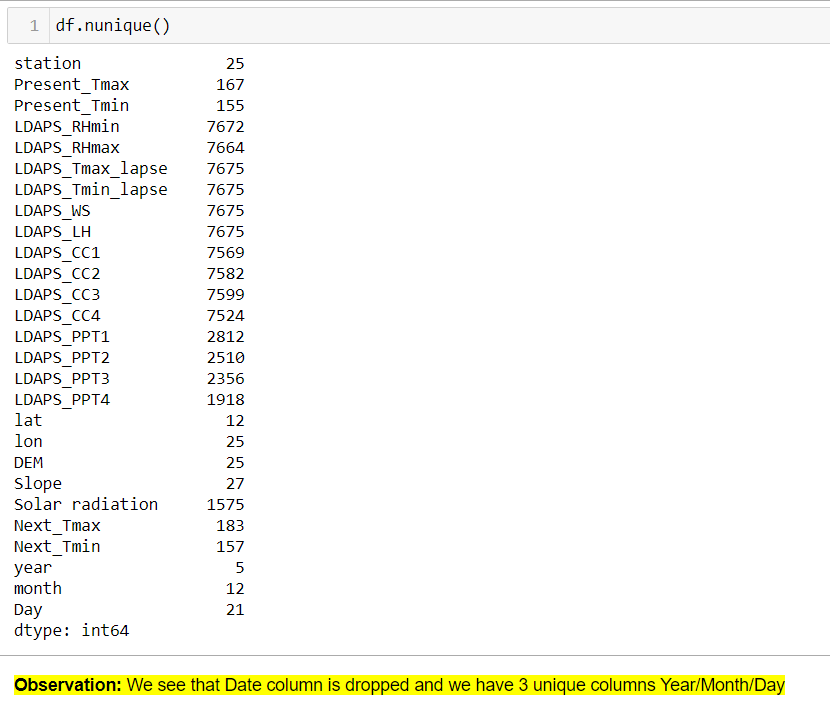
* We had seen the “Date” column was object, which needs to be converted
* Also, there were missing values, which needs to be handled

#### Converting Date Column

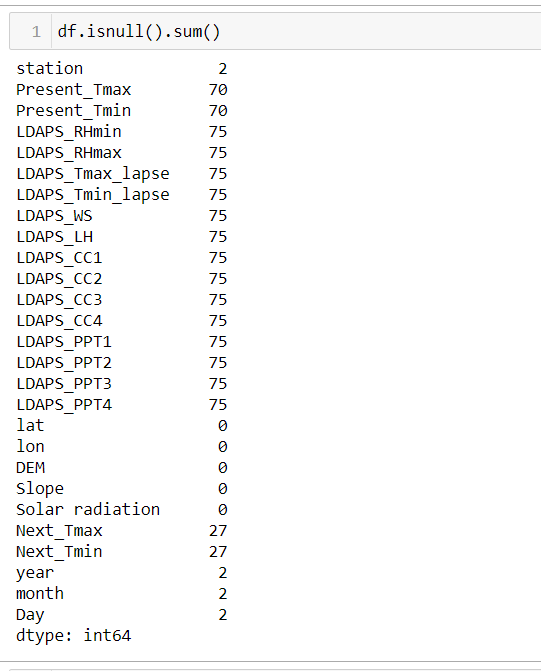


* “Date” column has year, month and date.
* So, let’s split into 3 columns Date/Month/Year and then drop the Date column after conversion





#### Handling Missing Values

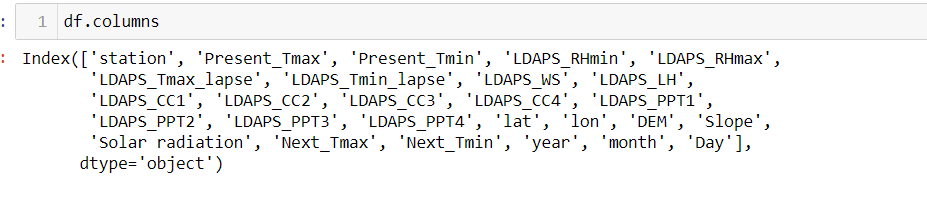


* Even the newly created columns have missing values
* And the max missing value is only 75, which is less than 1% of data.
* So, we can replace the data with mean/median

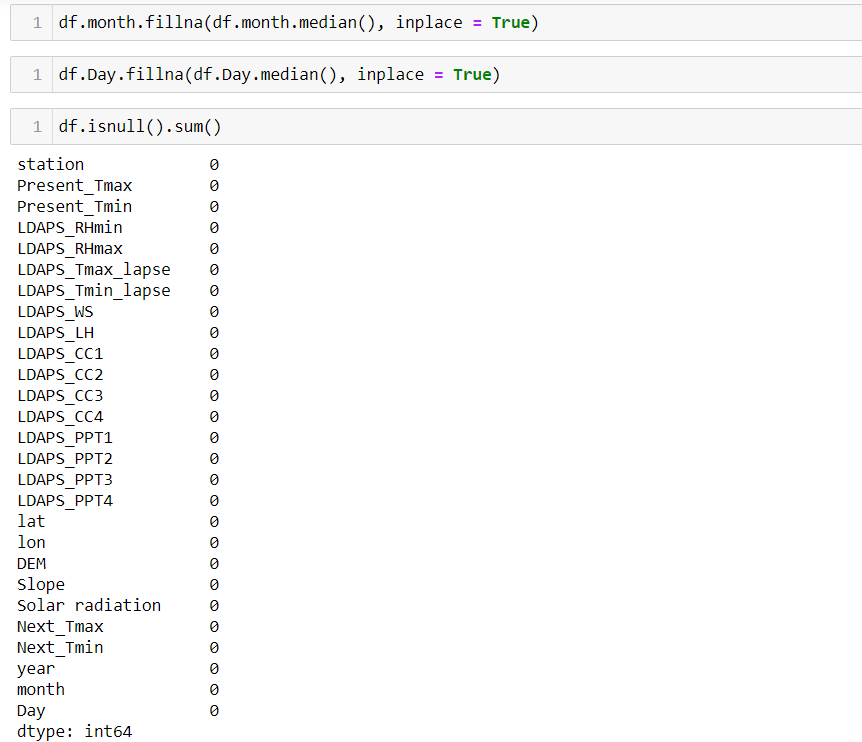
Tip2:

* For Normally distributed data we can replace with mean, else with Median. So instead of checking for each column, we are going to replace for all rows will Median. Since in Normally distributed data, mean=median. So, in both cases (Normally distributed or not), we can replace with median data.
* For Categorical Data, we will replace with Mode

Tip3:



* When working with Column names, get list of column names using df.columns, so you can copy column name while using it in any part of the code.
* The purpose of doing it, we don’t have to remember the naming convention (caps, small etc)
* Any special characters, or spaces etc will also be copied



* For all the columns, which had missing data, we replaced with Median. Sample 2 columns are shown
* Now when we check after the replacement, we see no Missing values, all have been replaced by Median

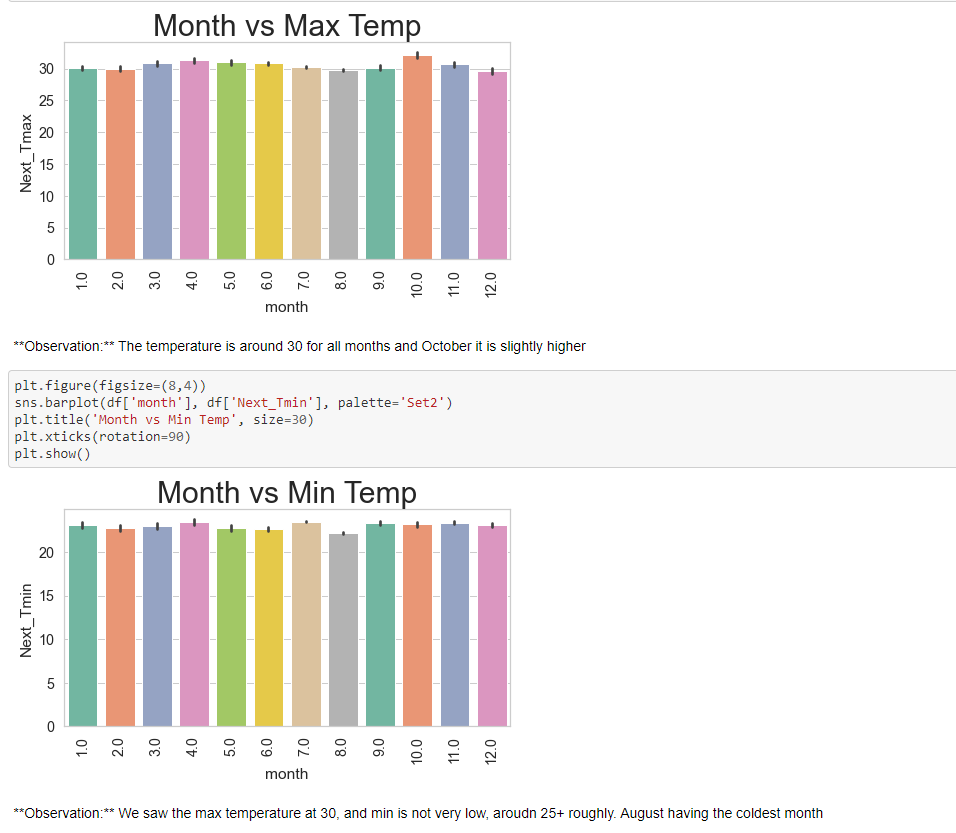
### Exploratory Data Analysis

* Since we have 2 target Variables, we are going to see few of the EDA we did.
* The idea is to see the relationship between the 2 target variables and their dependency if any on the independent variables

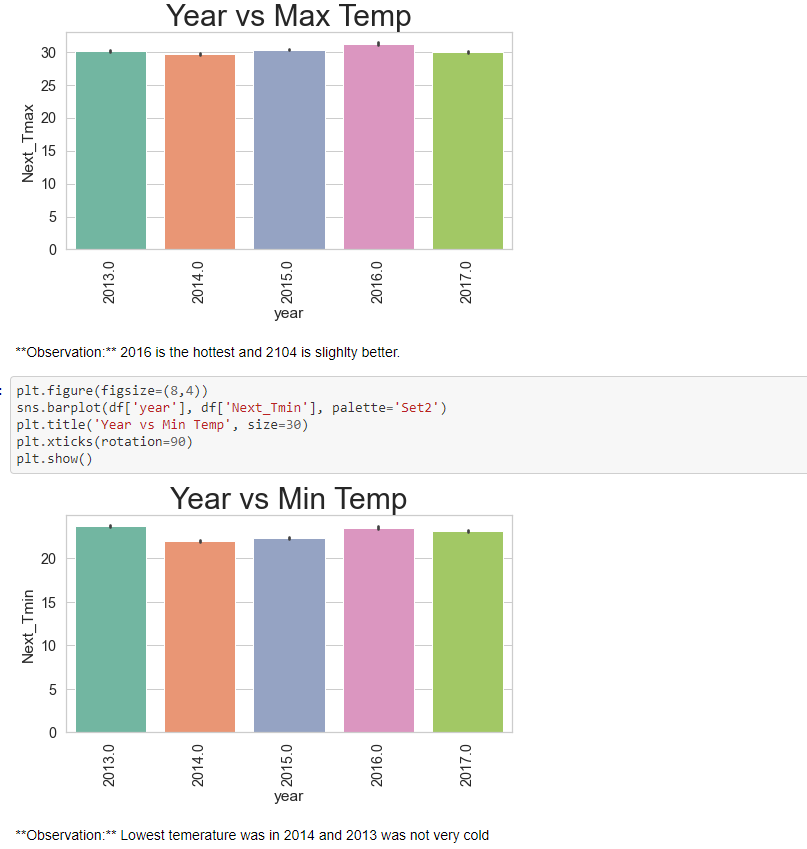
Tip 4:

* We can do a Uni-variate Analysis of the Target Variable
* Then do a Bi-Variate Analysis of the Target Variable Vs Independent Variables
  + Since there might be multiple Independent Variables, if it is binomial, we can run it in a loop

#### Month Max/Min Temperature



#### 4.3.2 Min/Max Temperature across Years

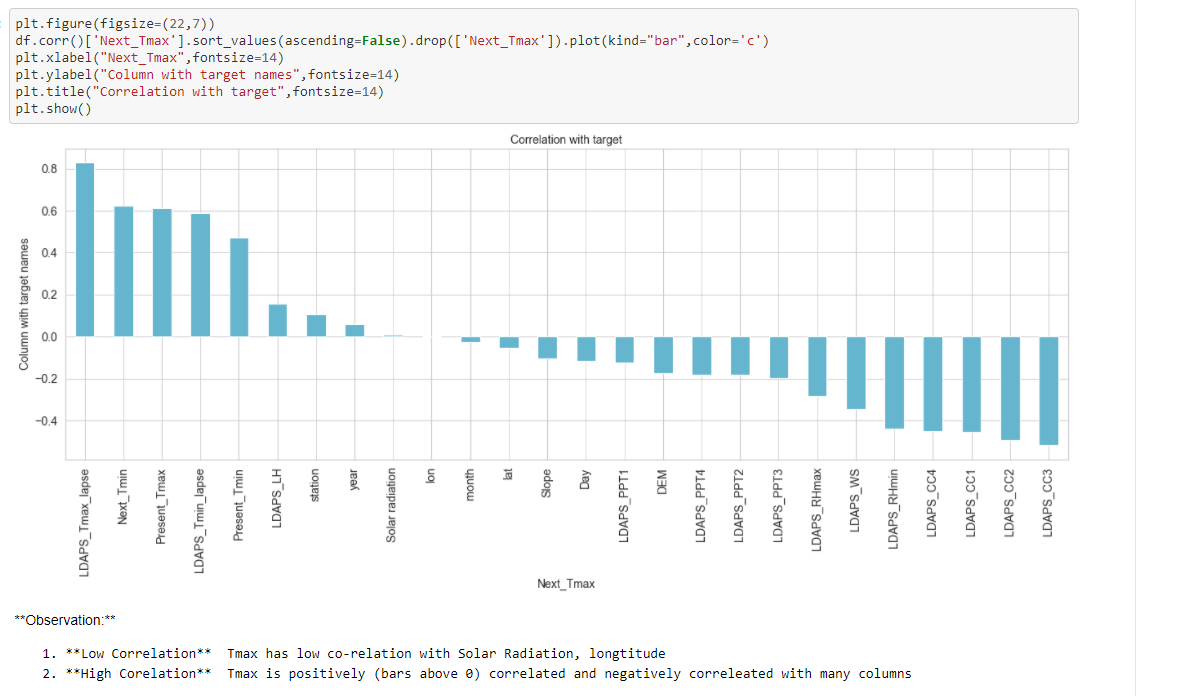


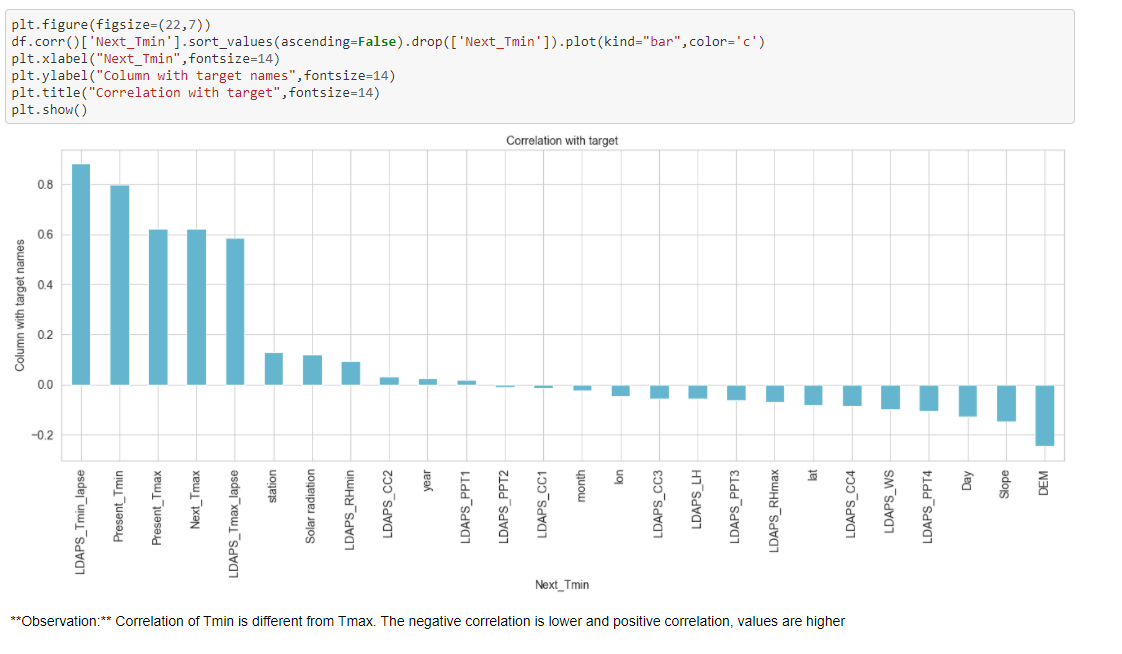
#### 4.3.3 Co-relation- Target Variable(s)

We would like to see the co-relation with the Target Variables. Since we have 2 target variables, lets draw the co-relation and plot them, as a visual view makes it more easier for us to read!

Tip5:

* Heatmap is good to find a co-relation of multiple variables together
* However, when we have multiple columns like in this dataset, we can focus on each variable, or directly the Target Variable(s)
* Also, instead of having the entire heatmap, having just one part of the triangle (lower half) makes it easier to read.



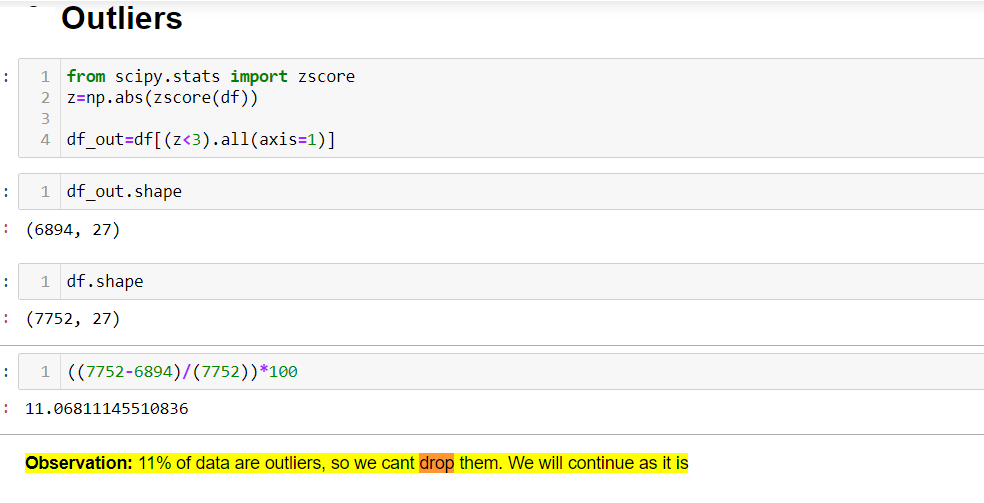


### Data preparation for Model Building

We will be doing the following

* 1. Check and Remove Outliers if possible
  2. Check for Skewness and handle them

#### Handling Outliers



* We have 11% of outliers and we can’t remove the outliers without discussing with biz
* So, we will continue to use our data as it is

#### Skewness Handling

Tip6:

* Before handling the Skewness, remove the target columns from the data frame.
* Reason is the biz won’t be able to understand the target data, as after we skew the numbers will change



We have handled the skewness and only 4 columns are skewed, initially it was 14. So, we are good to go as it is now

Tip7:

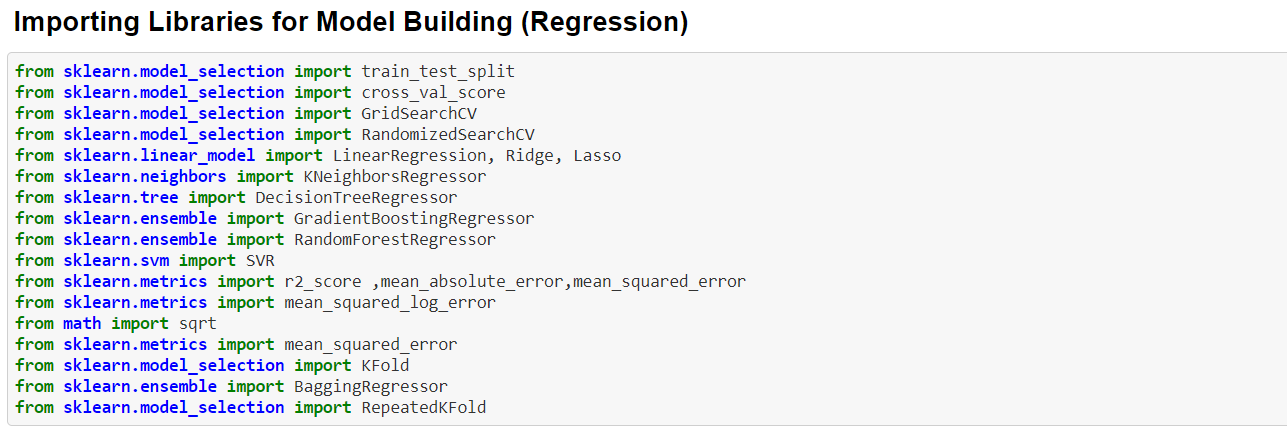
* Since we used Power Transform, our data might be Scaled appropriately.
* We need not use Min Max scalar or Standard Scalar after applying Power Transform on our data
* You can do a quick check with df.describe to check the range of data and decide

### Model building

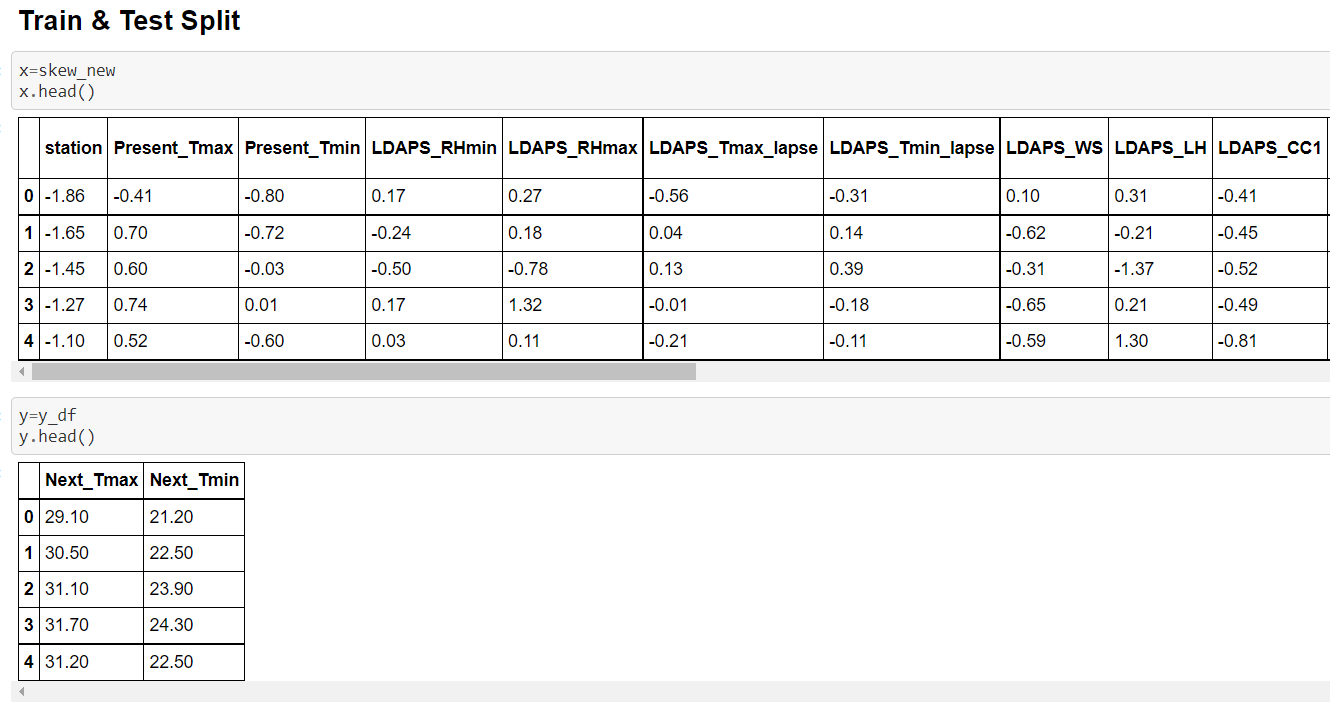
* Many machine learning algorithms are designed for predicting a single numeric value, referred to simply as regression. Some algorithms do support multioutput regression inherently, such as linear regression and decision trees

For Model building we have the following steps. Each I have given a screen shot below

#### Importing the required libraries for Model Building and Evaluation



#### Splitting our data into X & Y

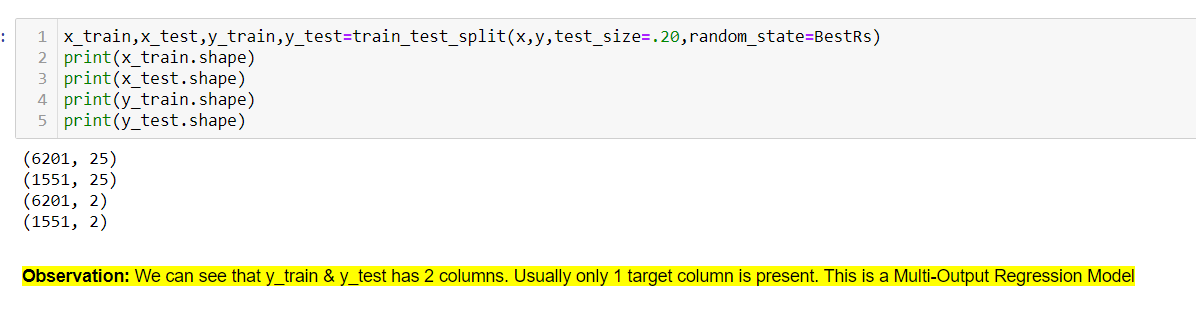


* Note that there 2 target variables for y

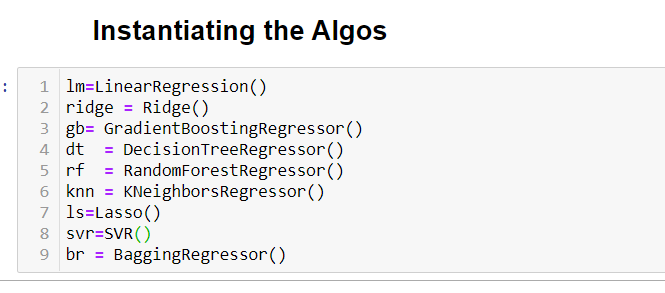
#### Finding Best Random State

Below we will find the Best Random State and use the same in our Train /Test Split





#### Instantiating the Algos



#### Algorithm & Metrics for Multi-output

Tip8:

* Instead of using each algorithm one by one, we can do a user defined function
* Then we will call this user defined function, for each Algo
* E.g., SVR, Gradient Boost etc wont work with multi-output regression. We will see it gives an error. So, we can use only the Algorithms that will work

**Algorithms**

Some regression machine learning algorithms support multiple outputs directly.

This includes most of the popular machine learning algorithms implemented in the scikit-learn library, such as:

1. LinearRegression (and related)
2. KNeighborsRegressor
3. DecisionTreeRegressor
4. RandomForestRegressor (and related)

**Multioutput**

In this Dataset we fit all the 4 algorithms on the multioutput regression dataset, then makes a single prediction with the fit model.

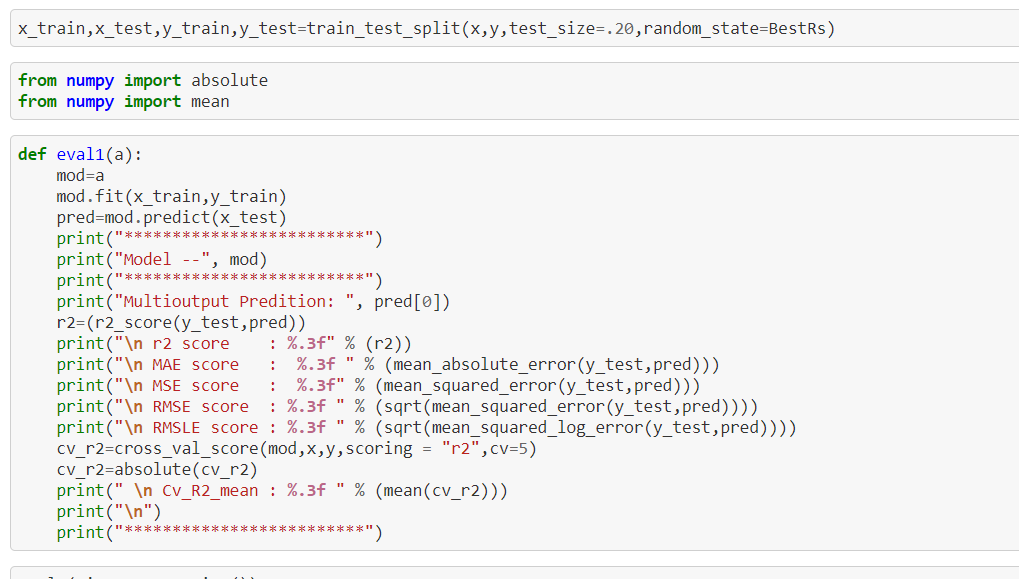
Running dataset with all the 4 algorithm and then makes a prediction for one input, confirming that the model predicted two required values.

**Cross Validation**

* To avoid Over-fitting and Under-fitting issues we can do Cross Validation.
* Error is reported across both output variables, rather than separate error scores for each output variable.
* We will be using the default fold size =5 and then take a mean
* Also, since R2 is our preferred metric, we will be using our scoring method as “r2” in cross validation.

#### Defining the Function

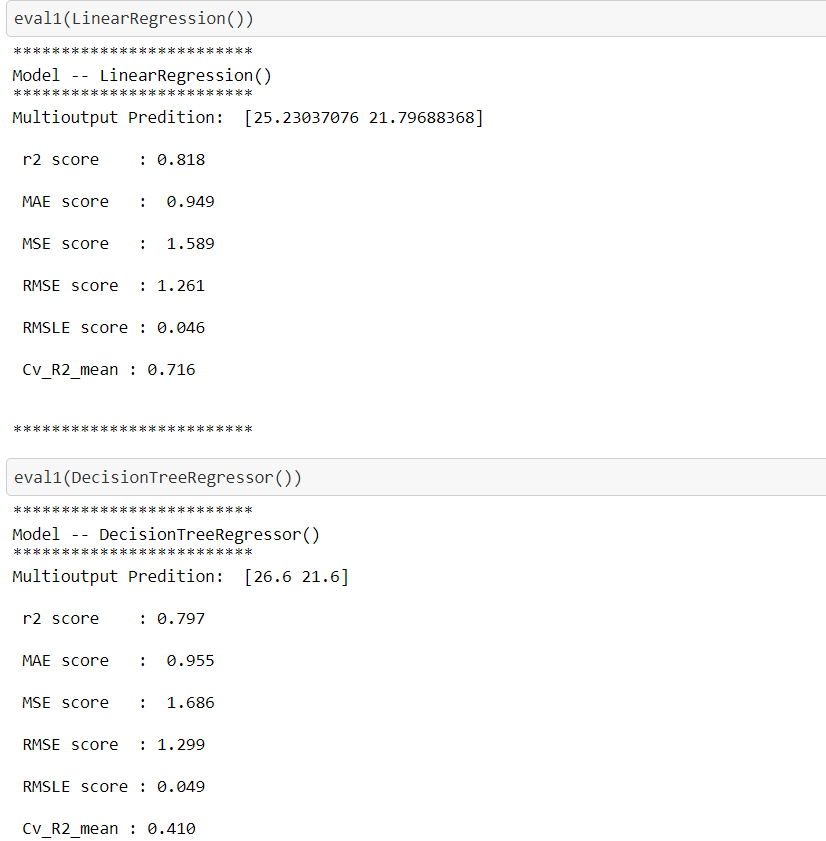
* We are defining a function, and using the Best Random State as well



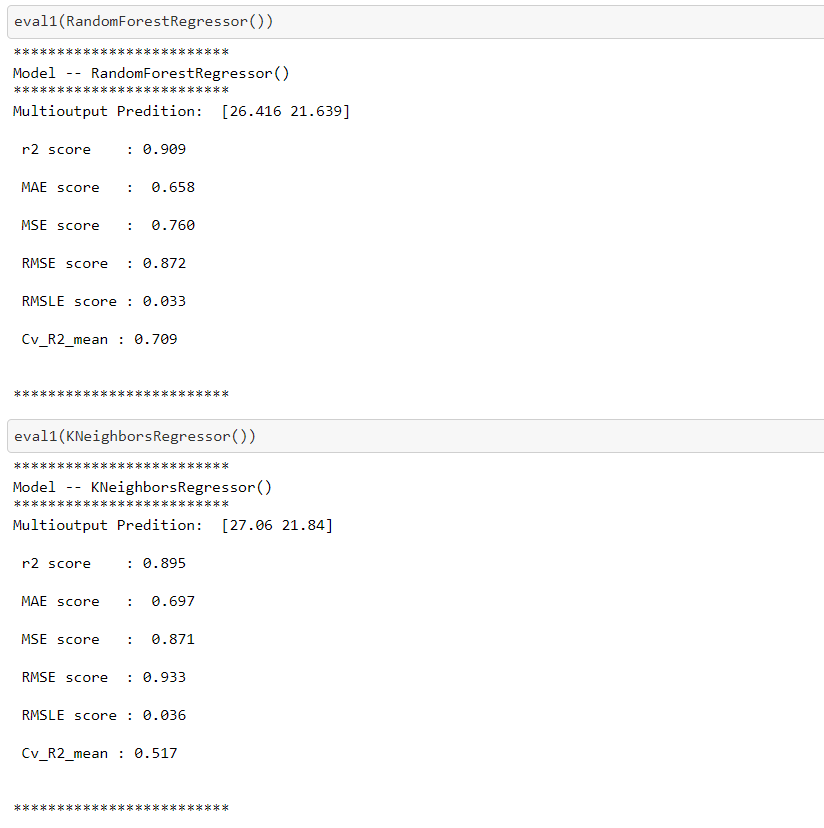
### Evaluation Metrics

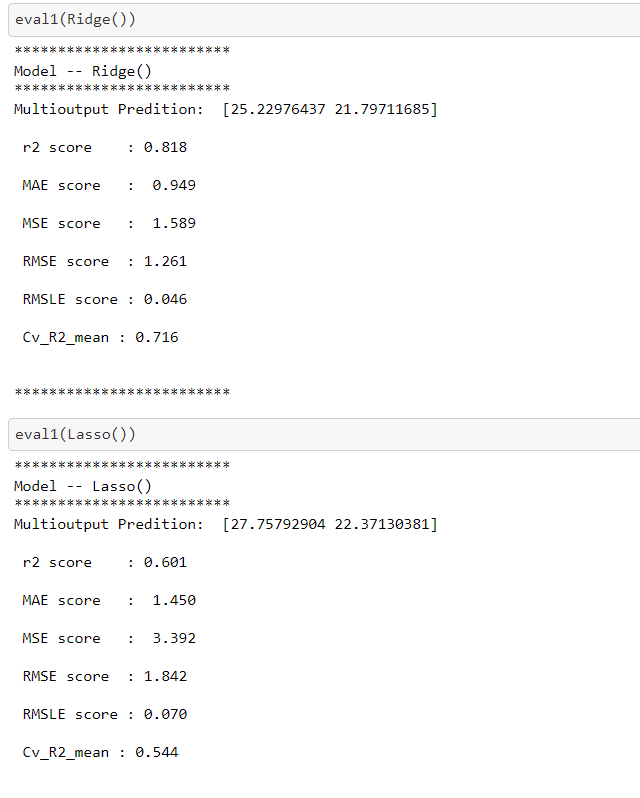
* Though we have used to find the Error, R2, MAE, MSE, RMSE, RMSLE
* We will call each algo using the User Defined function, “eval1”

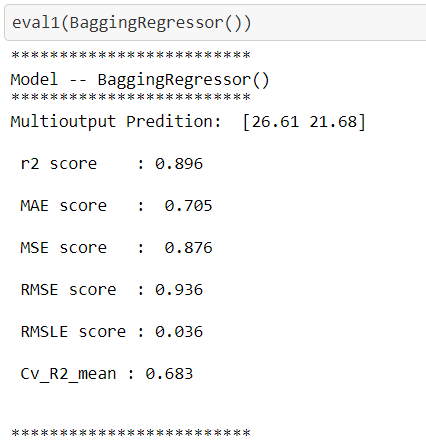
#### Calling each algorithm and its metrics



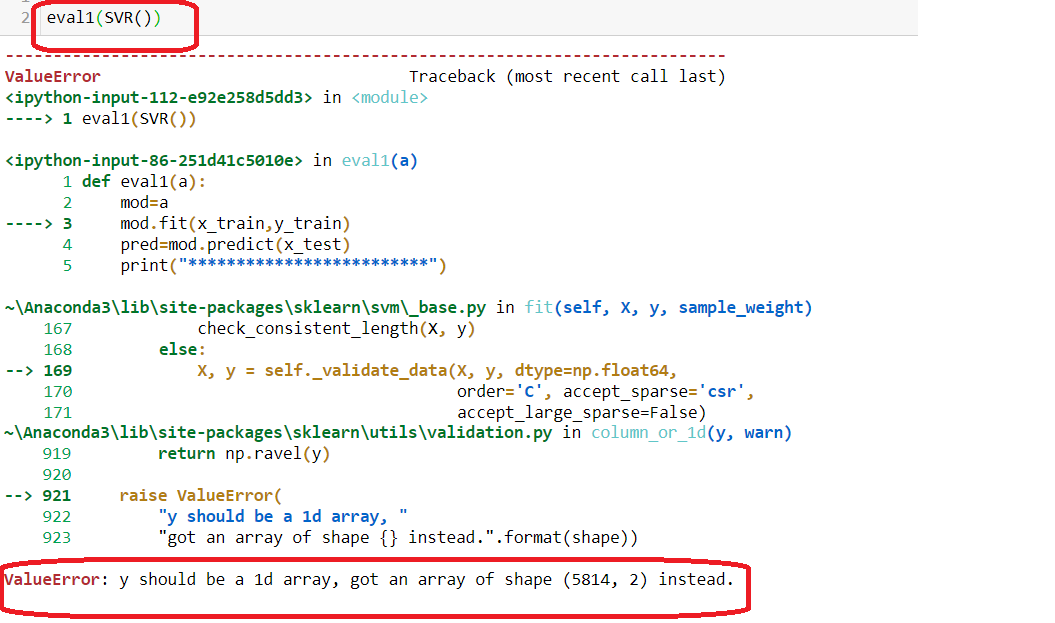
* Each fits the model and then makes a prediction for one input, confirming that the model predicted two required values.
* Both the values have been highlighted in the screen shot
* Error is reported across both output variables, rather than separate error scores for each output variable, so that is why we have only one output for each Metric







#### Unsupported Algorithms and Error



* Highlighted the algo used (SVR) and the error. Since we have 2 target variables, it is not supported here

#### Comparing All Algos & Evaluation Metrics

**Observation:**

1. In **Multi-output Regression** we see 2 values for each Algorithm as expected, as we have 2 target variables in this Dataset
2. **R2** is a good metric to measure, and we have got good scores in Random Forest, Bagging Regressor and KNN. Highlighted only the top 2. So based on this we can use these 3 algos for Hyperparameter Tuning
   1. **RMSE** Scores are high, as the errors are punished higher. The **RMSLE** negates it, and we see good low scores on RMSLE as well for Random Forest, Bagging, and KNN. Another reason to use them in finding RMSLE
3. R2 and **Cross Validation** difference is low for Linear Algos (Linear, Lasso, Ridge), however their R2 scores are not that high. So, if we remove them, again Random Forest and Bagging tops our chart

We will use the 3 algos which are topping in every column for Hyper Parameter Tuning- Random Forest/KNN/Bagging

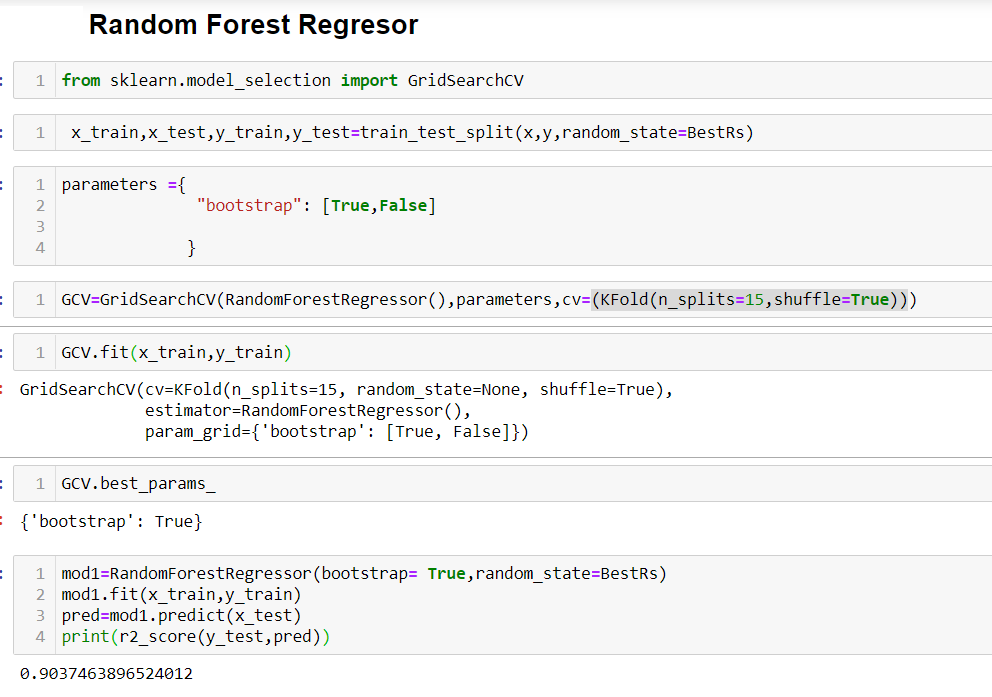
### Improving the Model

We can fine tune our Model using

1. Random Search
2. Grid Search

Since our dataset is moderate let’s use both as needed

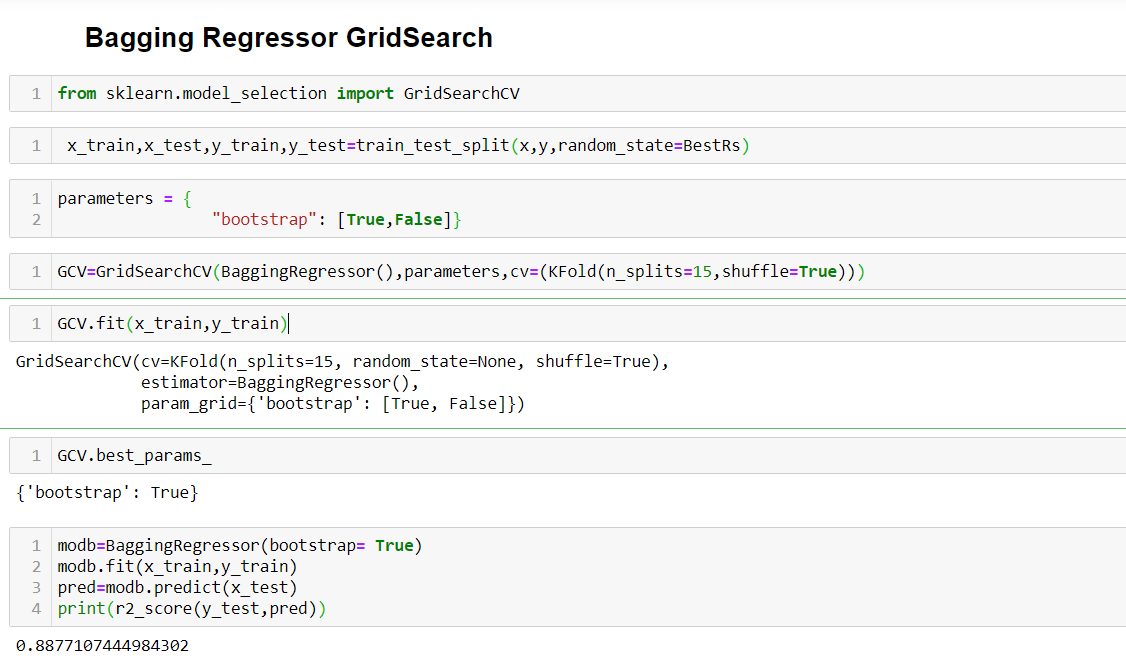
#### Random Forest



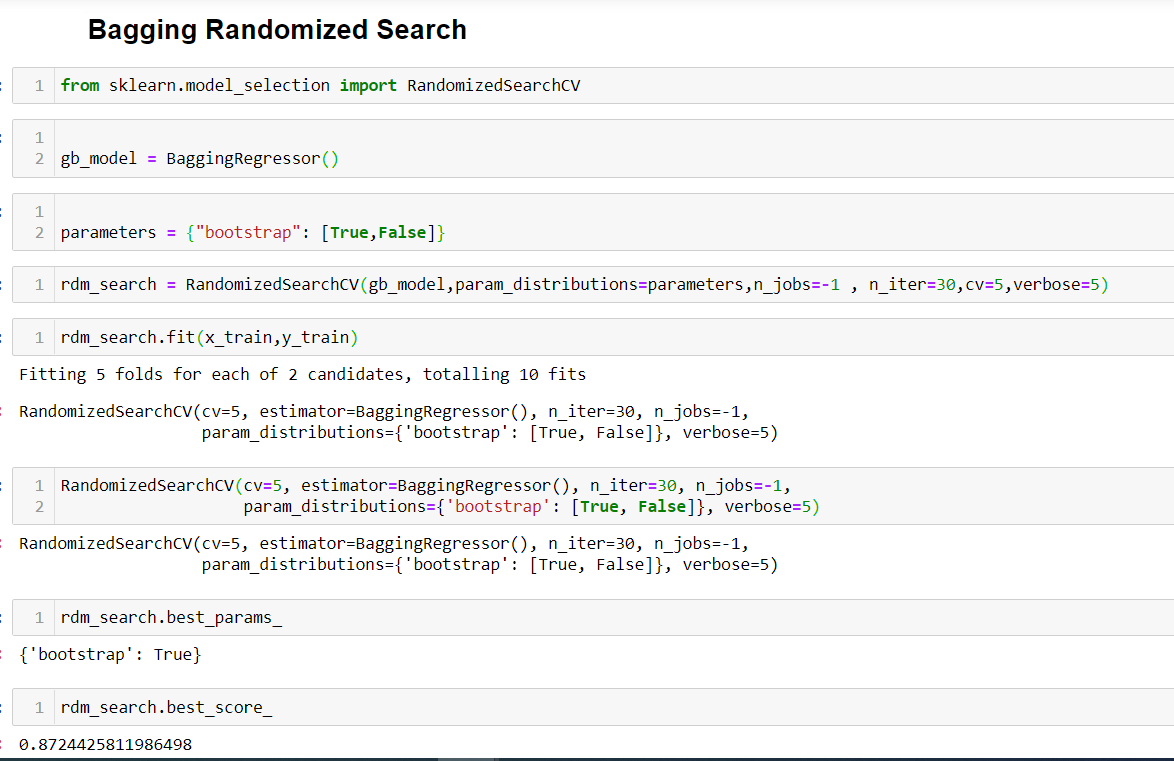
#### KNN



#### Bagging Regressor- Grid Search



#### Bagging Regressor- Random Search



#### Comparing the Cross Validation with Hyper Parameter Tuning

|  |  |  |  |
| --- | --- | --- | --- |
| **Hyper Parameter Tuning** | | | |
|  | **CV** | **Grid Search** | **Random Search** |
| **Random Forest** | 0.709 | 0.9037 |  |
| **KNN** | 0.517 | 0.8902 |  |
| **Bagging Regressor** | 0.683 | 0.887 | 0.887 |

* We have seen our scores have improved with Grid Search and Random Search Hyper Parameter Tuning
* Random Forest Has the highest score, so let’s save that and share it with Biz

### Saving & Loading the Model



## 5.Conclusion

We got a good score of 90.37% & predict for 2 Variables, since

* Proper Cleaning of the Data
* Good EDA to understand the relation between the target variables and the independent variables, to help us determine the data needed
* Chose the Best Algorithms for Multi-output Regression
* Good Evaluation Metrics for Multi-output Regression
* Hyper Tuning using both methods of Grid & Random Search