Flight price prediction regression

By
Amit Kumar Tiwari

Index

1. Introduction
    I. Problem Statement
    II. Data
2. Methodology
    I. Pre-Processing
    II. Modelling
    III. Model Selection
    IV. Tuning
    V. Prediction
    VI. Model evaluation
3. Pre- processing
    I. Data exploration and Cleaning (Missing Values and Outliers)
    II. Creating some new variables from the given variables
    III. Selection of variables
    IV. Some more data explorations
        a) Dependent and Independent Variables
        b) Uniqueness of Variables
        c) Dividing the variables categories
        d) Feature Scaling
4. Modelling
    I. Training of various regression models
5. Conclusion

II. Model Evaluation
III. Mean Absolute Percentage Error (MAPE)
IV. Accuracy
V. R Square
VI. Cross Validation
VII. Model Selection
6. References

1. Introduction

Problem Statement:

Flight ticket prices can be something hard to guess, today we might see a price, check out the price of the same flight tomorrow, it will be a different story. We might have often heard travelers saying that flight ticket prices are so unpredictable. Here you will be provided with prices of flight tickets for various airlines between the months of March and June of 2019 and between various cities.

Problem Statement:

Size of training set: 10683 records

Size of test set: 2671 records

FEATURES:

Airline: The name of the airline.

Date of Journey: The date of the journey

Source: The source from which the service begins.

Destination: The destination where the service ends.

Route: The route taken by the flight to reach the destination.

Dep Time: The time when the journey starts from the source.

Arrival Time: Time of arrival at the destination.

Duration: Total duration of the flight.

Total Stops: Total stops between the source and destination.

Additional Info: Additional information about the flight

Price: The price of the ticket

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897.0
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	No info	7662.0
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	2 stops	No info	13882.0
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	1 stop	No info	6218.0
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	1 stop	No info	13302.0
...	...	...	...	...	...	...	...	...	...	...	...
2666	Air India	6/06/2019	Kolkata	Banglore	CCU → DEL → BLR	20:30	20:25 07 Jun	23h 55m	1 stop	No info	NaN
2667	IndiGo	27/03/2019	Kolkata	Banglore	CCU → BLR	14:20	16:55	2h 35m	non-stop	No info	NaN
2668	Jet Airways	6/03/2019	Delhi	Cochin	DEL → BOM → COK	21:50	04:25 07 Mar	6h 35m	1 stop	No info	NaN
2669	Air India	6/03/2019	Delhi	Cochin	DEL → BOM → COK	04:00	19:15	15h 15m	1 stop	No info	NaN
2670	Multiple carriers	15/06/2019	Delhi	Cochin	DEL → BOM → COK	04:55	19:15	14h 20m	1 stop	No info	NaN

13354 rows × 11 columns

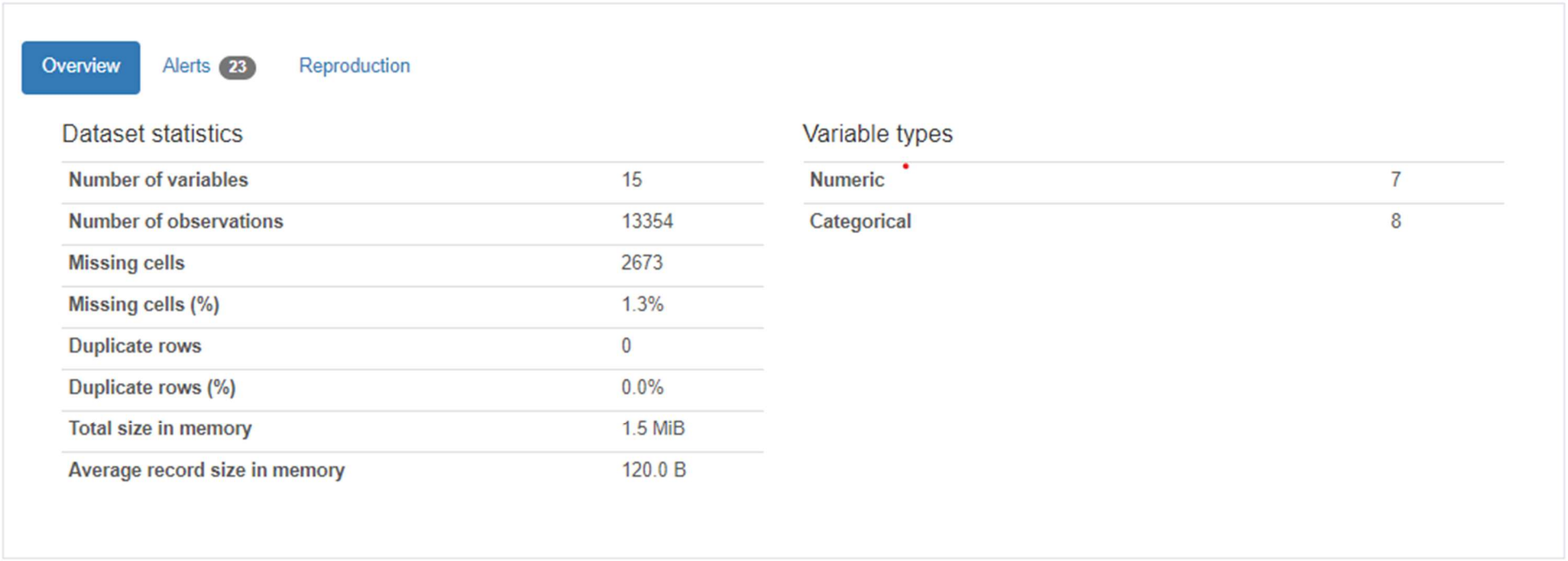
2. Methodology

After going through the dataset in detail and pre-understanding the data the next step is, Methodology that will help achieve our goal. In Methodology following processes are followed:

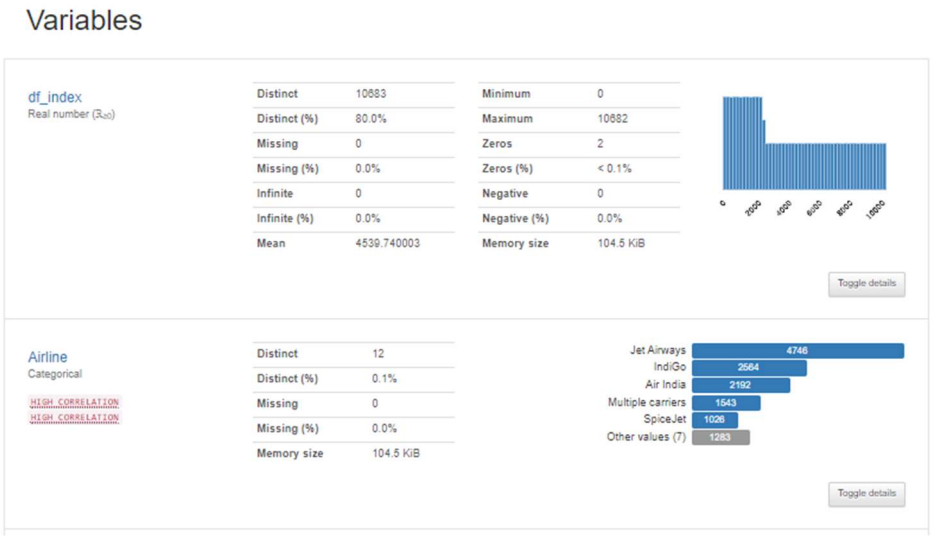
1. Loading the available dataset
2. Pre-processing: It includes missing value analysis, outlier analysis, feature selection and feature scaling.
3. Model development: It includes identifying suitable Machine learning Algorithms and applying those algorithms in our given dataset.
4. Model selection – It includes identifying & selecting the best model depending on various set of parameters defined for best fit model.
5. Prediction: Predicting the outcome based on finalized set of variables i.e., dataset.
6. Model evaluation: Evaluating & tunning the best model & predicting the target.

Pre-processing/ Report generation:

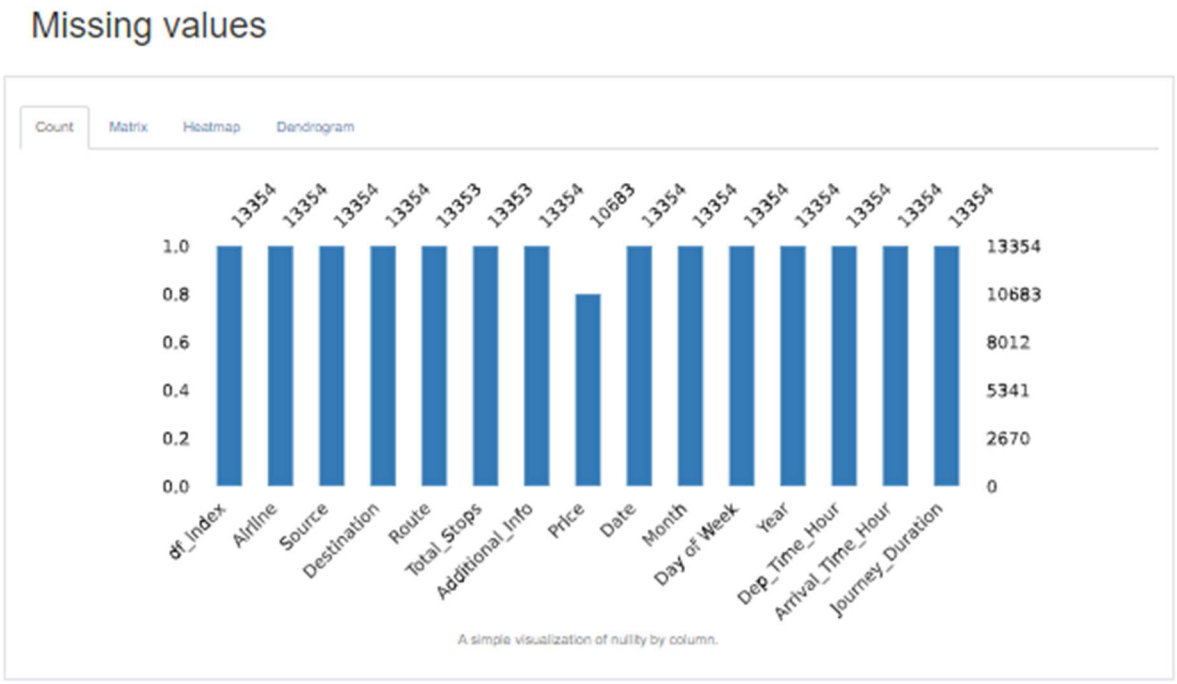
We used pandas profiling here to check various aspect of the dataset & EDA analysis. Report is as below:



Here we can have a look into the data set respective distribution of the various variables. There is an option to toggle the dataset which is helpful in finding more insight tot eh dataset.



Similarly, we can have a look into the various other graphs under the interactions section.



Duplicate rows & columns analysis & removal

Duplicate rows & column may result in making the model complex & may help it in producing the false prediction. We are here removing the duplicates from the dataset.

```
Data_Day.dropna(inplace = True)
Data_Day
```

	Airline	Source	Destination	Route	Total_Stops	Additional_Info	Price	Date	Month	Day of Week	Year	Dep_Time_Hour	Arrival_Time_Hour	Journey_Duration
0	IndiGo	Banglore	New Delhi	BLR → DEL	non-stop	No info	3897.0	24	3	6	2019	22	1	170m
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	2 stops	No info	7662.0	5	1	5	2019	5	13	445m
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	2 stops	No info	13882.0	6	9	4	2019	9	4	1140m
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	1 stop	No info	6218.0	5	12	3	2019	18	23	325m
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	1 stop	No info	13302.0	3	1	3	2019	16	21	285m
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10678	Air Asia	Kolkata	Banglore	CCU → BLR	non-stop	No info	4107.0	4	9	2	2019	19	22	150m
10679	Air India	Kolkata	Banglore	CCU → BLR	non-stop	No info	4145.0	27	4	5	2019	20	23	155m
10680	Jet Airways	Banglore	Delhi	BLR → DEL	non-stop	No info	7229.0	27	4	5	2019	8	11	180m
10681	Vistara	Banglore	New Delhi	BLR → DEL	non-stop	No info	12648.0	3	1	3	2019	11	14	160m
10682	Air India	Delhi	Cochin	DEL → GOI → BOM → COK	2 stops	No info	11753.0	5	9	3	2019	10	19	500m

10682 rows × 14 columns

```
[20] #Deleting duplicate rows
Data_Day = Data_Day.drop_duplicates()
Data_Day
```

	Airline	Source	Destination	Route	Total_Stops	Additional_Info	Price	Date	Month	Day of Week	Year	Dep_Time_Hour	Arrival_Time_Hour	Journey_Duration
0	IndiGo	Banglore	New Delhi	BLR → DEL	non-stop	No info	3897.0	24	3	6	2019	22	1	170m
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	2 stops	No info	7662.0	5	1	5	2019	5	13	445m
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	2 stops	No info	13882.0	6	9	4	2019	9	4	1140m
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	1 stop	No info	6218.0	5	12	3	2019	18	23	325m
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	1 stop	No info	13302.0	3	1	3	2019	16	21	285m
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10678	Air Asia	Kolkata	Banglore	CCU → BLR	non-stop	No info	4107.0	4	9	2	2019	19	22	150m
10679	Air India	Kolkata	Banglore	CCU → BLR	non-stop	No info	4145.0	27	4	5	2019	20	23	155m
10680	Jet Airways	Banglore	Delhi	BLR → DEL	non-stop	No info	7229.0	27	4	5	2019	8	11	180m
10681	Vistara	Banglore	New Delhi	BLR → DEL	non-stop	No info	12648.0	3	1	3	2019	11	14	160m
10682	Air India	Delhi	Cochin	DEL → GOI → BOM → COK	2 stops	No info	11753.0	5	9	3	2019	10	19	500m

10438 rows × 14 columns

remove or replace the

Label encoding

Label encoding is a process of encoding the categorical values into numbers it will help us in making data more easier for the model to interpret it & make predictions

```
# Import label encoder
from sklearn import preprocessing
# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()
Data_Day['Airline']= label_encoder.fit_transform(Data_Day['Airline'])
Data_Day['Source']= label_encoder.fit_transform(Data_Day['Source'])
Data_Day['Destination']= label_encoder.fit_transform(Data_Day['Destination'])
Data_Day['Route']= label_encoder.fit_transform(Data_Day['Route'])
Data_Day['Total_Stops']= label_encoder.fit_transform(Data_Day['Total_Stops'])
Data_Day
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if __name__ == '__main__':
```

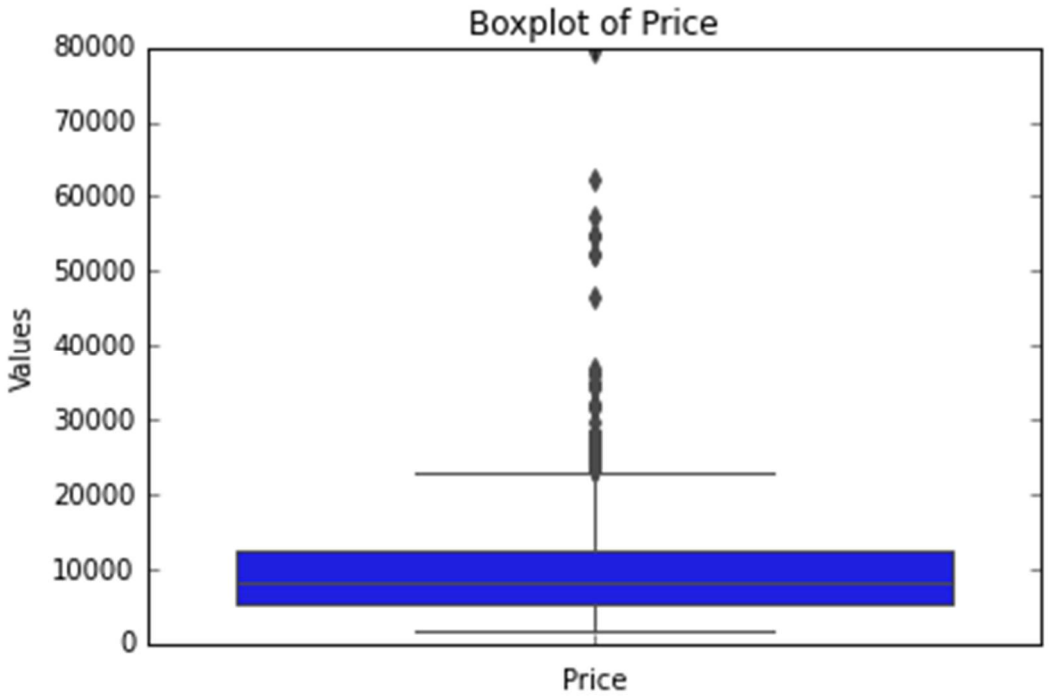
	Airline	Source	Destination	Route	Total_Stops	Additional_Info	Price	Date	Month	Day of Week	Year	Dep_Time_Hour	Arrival_Time_Hour	Journey_Duration
0	3	0	5	18	4	No info	3897.0	24	3	6	2019	22	1	170m
1	1	3	0	84	1	No info	7662.0	5	1	5	2019	5	13	445m
2	4	2	1	118	1	No info	13882.0	6	9	4	2019	9	4	1140m
3	3	3	0	91	0	No info	6218.0	5	12	3	2019	18	23	325m
4	3	0	5	29	0	No info	13302.0	3	1	3	2019	16	21	285m
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10678	0	3	0	64	4	No info	4107.0	4	9	2	2019	19	22	150m
10679	1	3	0	64	4	No info	4145.0	27	4	5	2019	20	23	155m
10680	4	0	2	18	4	No info	7229.0	27	4	5	2019	8	11	180m
10681	10	0	5	18	4	No info	12648.0	3	1	3	2019	11	14	160m
10682	1	2	1	108	1	No info	11753.0	5	9	3	2019	10	19	500m

```
✓ 0s  completed at 11:26 PM
```

Outlier Analysis

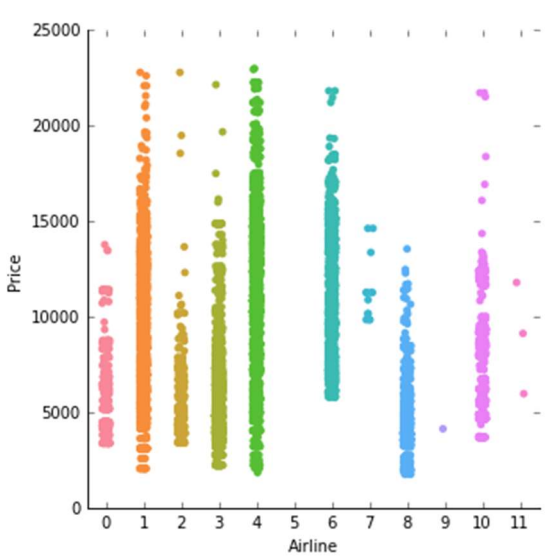
Outlier is an abnormal observation that stands or deviates away from other observations. These happen because of manual error; poor quality of data and it is correct but exceptional data. But it can cause an error in predicting the target variables. So, we must check for outliers in our data set and outliers wherever required. In this project, outliers are found in only two variables this are Humidity and wind speed, following are the box plots for both the variables and dots outside the quartile ranges are outliers. All these outliers mentioned above happened because of manual error, or interchange of data, or may be correct data but exceptional. But all these outliers can hamper our data model. So there

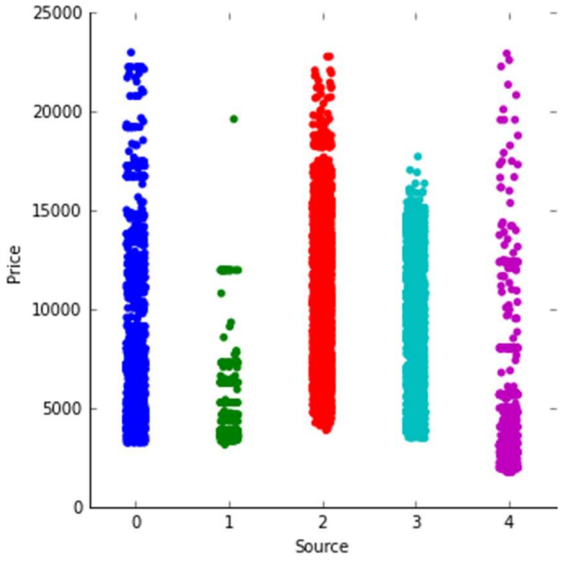
is a requirement to eliminate or replace such outliers and impute with proper methods to get better accuracy of the model. In this project, I used median method to impute the outliers in wind speed and humidity variables.



Data Understanding

Data Understand is a process where we know our data in a better way by the help of visual representations and come up with initial ideas to develop our model. Here, the specific variables are plotted with respect to the target variable. In some cases, two variables are compared, whereas in some cases three variables are plotted together for our better understanding and visualization. Similarly various variables analyzed.





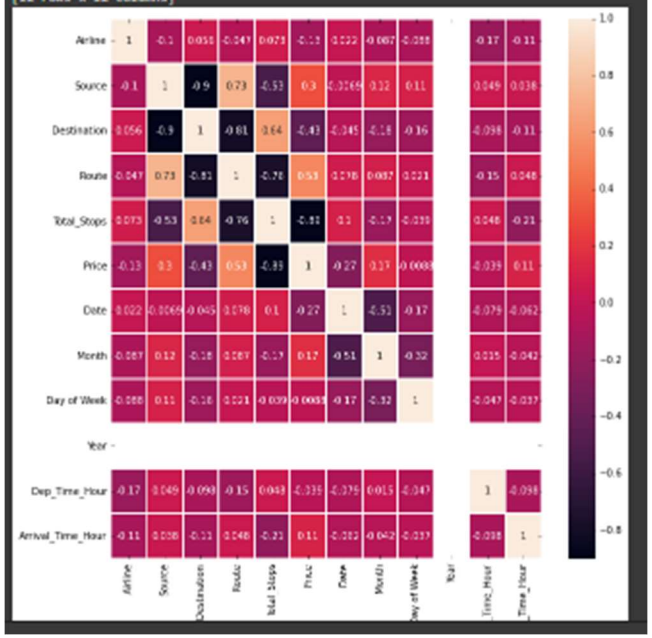
Feature Selection

Sometimes it happens that, all the variables in our data may not be accurate enough to predict the target variable, in such cases we need to analyze our data, understand our data, and select the dataset variables that can be most useful for our model. In such cases we follow feature selection. Feature selection helps by reducing time for computation of model and reduces the complexity of the model. Here, in this project correlation analysis is done with numerical variables and ANOVA test is done with categorical variables to check if there is collinearity among the variables. And if there is any collinearity it's better to drop such variables, else this redundant variable can hamper the accuracy of the model. a. Correlation Analysis for Numerical Variables.

```
Data_Day_cor = Data_Day.corr()
correlation_result = Data_Day_cor.corr()
print(correlation_result)
fig, ax = plt.subplots(figsize=(10,10)) ##To plot the heatmap into our required size
heatmap = sns.heatmap(correlation_result, annot=True, linewidths=.8)
```

	Airline	Source	...	Dep_Time_Hour	Arrival_Time_Hour
Airline	1.000000	-0.100885	...	-0.166430	-0.110440
Source	-0.100885	1.000000	...	0.049184	0.037826
Destination	0.055534	-0.900180	...	-0.098471	-0.108895
Route	-0.046790	0.726247	...	-0.147335	0.048175
Total_Stops	0.073255	-0.529884	...	0.047884	-0.207354
Price	-0.128963	0.295684	...	-0.038584	0.113647
Date	0.022102	-0.006863	...	-0.078526	-0.061536
Month	-0.086535	0.124160	...	0.015866	-0.042220
Day of Week	-0.087878	0.108649	...	-0.046698	-0.037332
Year	NaN	NaN	...	NaN	NaN
Dep_Time_Hour	-0.166430	0.049184	...	1.000000	-0.097586
Arrival_Time_Hour	-0.110440	0.037826	...	-0.097586	1.000000

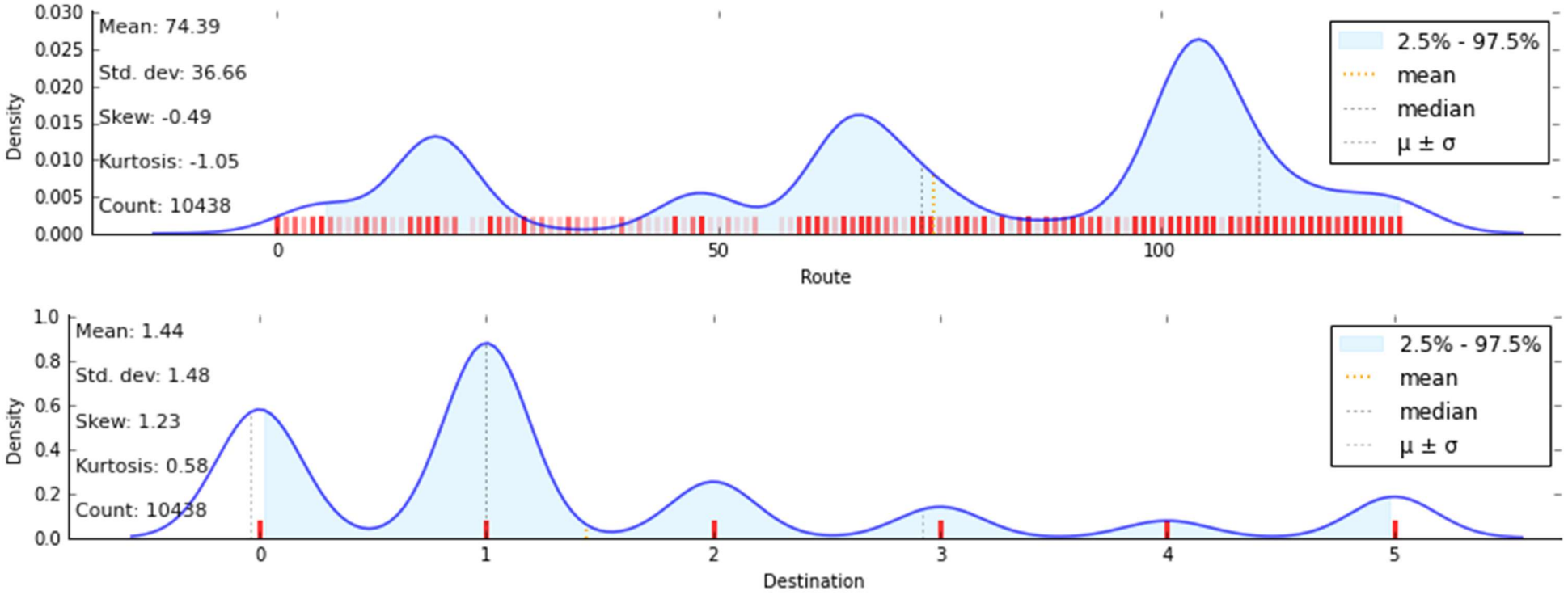
[12 rows x 12 columns]



We can here remove the variables with low relationship with target variable. We may also remove the variable which have very strong correlations with each other.

Data Distribution checkup & feature scaling

Here, In Feature Scaling ranges of variables are normalized or standardized, such that variables can be compared with same range. This is done for an unbiased and accurate model. In this project, as the data are found as approximately symmetric. The feature scaling is not required. Following are the plots of approximately symmetric data visuals.



3. Model Development, Evaluation & Tuning

The next step after Exploratory Data Analysis and Data Pre-Processing is Model Development. Now we have our data ready to be implemented to develop a model. There are number of models and Machine learning algorithms that are used to develop model. Here I have used pyCaret for model development & checked the various available models for the dataset after proving the target variable as input.

```
models()
```

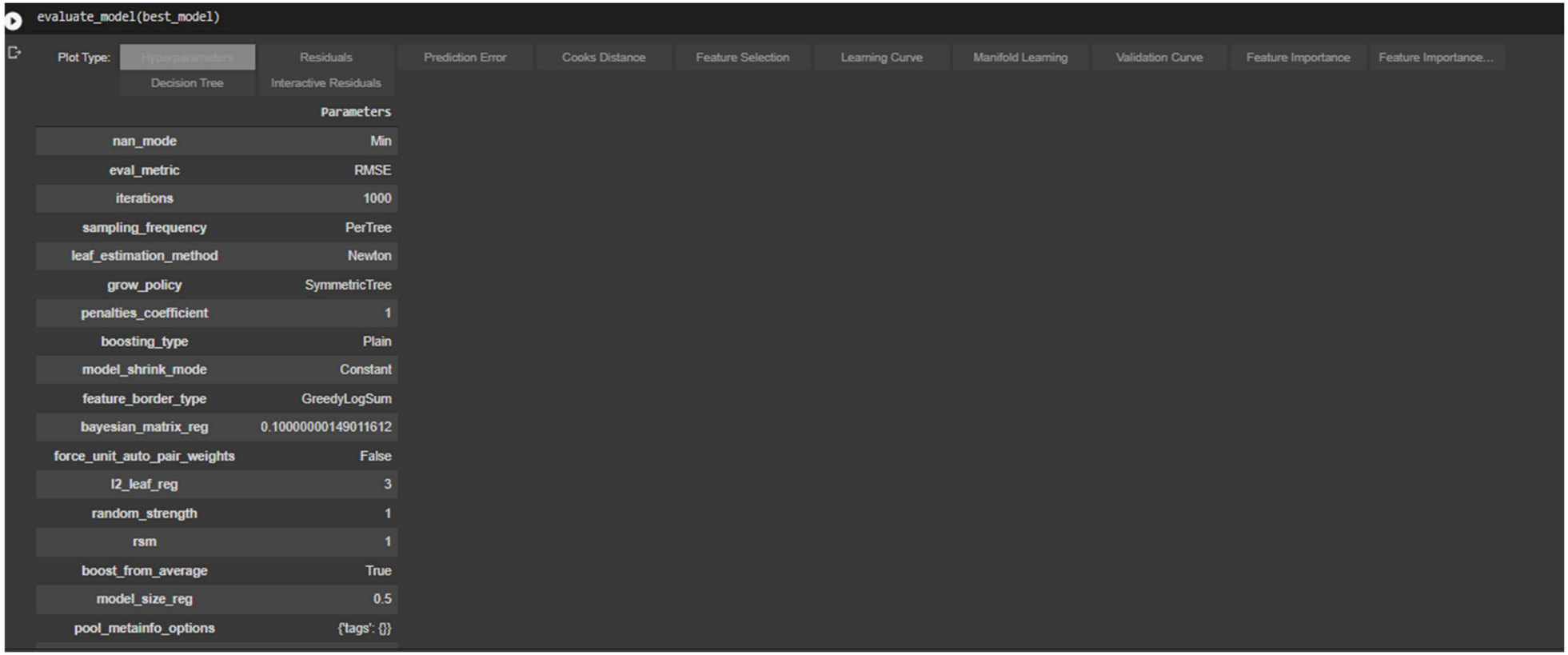
ID	Name	Reference	Turbo
lr	Linear Regression	sklearn.linear_model._base.LinearRegression	True
lasso	Lasso Regression	sklearn.linear_model._coordinate_descent.Lasso	True
ridge	Ridge Regression	sklearn.linear_model._ridge.Ridge	True
en	Elastic Net	sklearn.linear_model._coordinate_descent.Elast...	True
lar	Least Angle Regression	sklearn.linear_model._least_angle.Lars	True
llar	Lasso Least Angle Regression	sklearn.linear_model._least_angle.LassoLars	True
omp	Orthogonal Matching Pursuit	sklearn.linear_model._omp.OrthogonalMatchingPu...	True
br	Bayesian Ridge	sklearn.linear_model._bayes.BayesianRidge	True
ard	Automatic Relevance Determination	sklearn.linear_model._bayes.ARDRegression	False
par	Passive Aggressive Regressor	sklearn.linear_model._passive_aggressive.Passi...	True
ransac	Random Sample Consensus	sklearn.linear_model._ransac.RANSACRegressor	False
tr	TheilSen Regressor	sklearn.linear_model._theil_sen.TheilSenRegressor	False
huber	Huber Regressor	sklearn.linear_model._huber.HuberRegressor	True
kr	Kernel Ridge	sklearn.kernel_ridge.KernelRidge	False
svm	Support Vector Regression	sklearn.svm._classes.SVR	False
knn	K Neighbors Regressor	sklearn.neighbors._regression.KNeighborsRegressor	True
dt	Decision Tree Regressor	sklearn.tree._classes.DecisionTreeRegressor	True
rf	Random Forest Regressor	sklearn.ensemble._forest.RandomForestRegressor	True
et	Extra Trees Regressor	sklearn.ensemble._forest.ExtraTreesRegressor	True
ada	AdaBoost Regressor	sklearn.ensemble._weight_boosting.AdaBoostRegr...	True
gbr	Gradient Boosting Regressor	sklearn.ensemble._gb.GradientBoostingRegressor	True
mlp	MLP Regressor	sklearn.neural_network._multilayer_perceptron....	False
xgboost	Extreme Gradient Boosting	xgboost.sklearn.XGBRegressor	True
lightgbm	Light Gradient Boosting Machine	lightgbm.sklearn.LGBMRegressor	True
catboost	CatBoost Regressor	catboost.core.CatBoostRegressor	True

After this we compared various available models based on the 10-cross validation. Observation is that catboost is the best available model for the given problem.

```
# compare all models
best_model = compare_models()  ##Best model with 10 fold validation
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
catboost	CatBoost Regressor	1.377870e+03	3.828857e+06	1.955898e+03	7.689000e-01	0.2306	0.1774	2.106
xgboost	Extreme Gradient Boosting	1.358767e+03	3.862931e+06	1.964369e+03	7.669000e-01	0.2326	0.1749	3.789
lightgbm	Light Gradient Boosting Machine	1.401476e+03	3.899188e+06	1.973895e+03	7.647000e-01	0.2325	0.1802	0.155
rf	Random Forest Regressor	1.372602e+03	4.023956e+06	2.005161e+03	7.571000e-01	0.2369	0.1761	1.543
dt	Decision Tree Regressor	1.376858e+03	4.138340e+06	2.033355e+03	7.502000e-01	0.2399	0.1767	0.051
et	Extra Trees Regressor	1.386008e+03	4.185421e+06	2.044873e+03	7.472000e-01	0.2396	0.1770	1.588
gbr	Gradient Boosting Regressor	1.577316e+03	4.444520e+06	2.107885e+03	7.319000e-01	0.2465	0.2019	0.563
knn	K Neighbors Regressor	1.580761e+03	5.263611e+06	2.293573e+03	6.824000e-01	0.2657	0.2023	0.108
lasso	Lasso Regression	2.070570e+03	6.789461e+06	2.605039e+03	5.903000e-01	0.3033	0.2654	0.133
br	Bayesian Ridge	2.070820e+03	6.790937e+06	2.605308e+03	5.902000e-01	0.3036	0.2655	0.044
lr	Linear Regression	2.070026e+03	6.791696e+06	2.605432e+03	5.902000e-01	0.3037	0.2653	0.194
ridge	Ridge Regression	2.070574e+03	6.790297e+06	2.605180e+03	5.902000e-01	0.3036	0.2654	0.030
llar	Lasso Least Angle Regression	2.122800e+03	7.233450e+06	2.689034e+03	5.637000e-01	0.3067	0.2717	0.035
huber	Huber Regressor	2.090727e+03	7.342966e+06	2.709275e+03	5.570000e-01	0.3083	0.2594	0.380
omp	Orthogonal Matching Pursuit	2.164182e+03	7.576430e+06	2.752060e+03	5.430000e-01	0.3199	0.2808	0.031
ada	AdaBoost Regressor	2.306613e+03	7.819624e+06	2.796112e+03	5.283000e-01	0.3481	0.3354	0.480
par	Passive Aggressive Regressor	2.397701e+03	9.917088e+06	3.107845e+03	3.982000e-01	0.4145	0.2857	0.071
en	Elastic Net	2.768993e+03	1.099792e+07	3.315673e+03	3.371000e-01	0.4096	0.3999	0.033
lar	Least Angle Regression	5.855306e+06	8.777388e+14	1.559000e+07	-5.253179e+07	3.5223	841.7745	0.039

After comparing various model & selecting the best model its evaluation is done. We may switch between various available evaluation parameters for better insight ablaut the model performance.



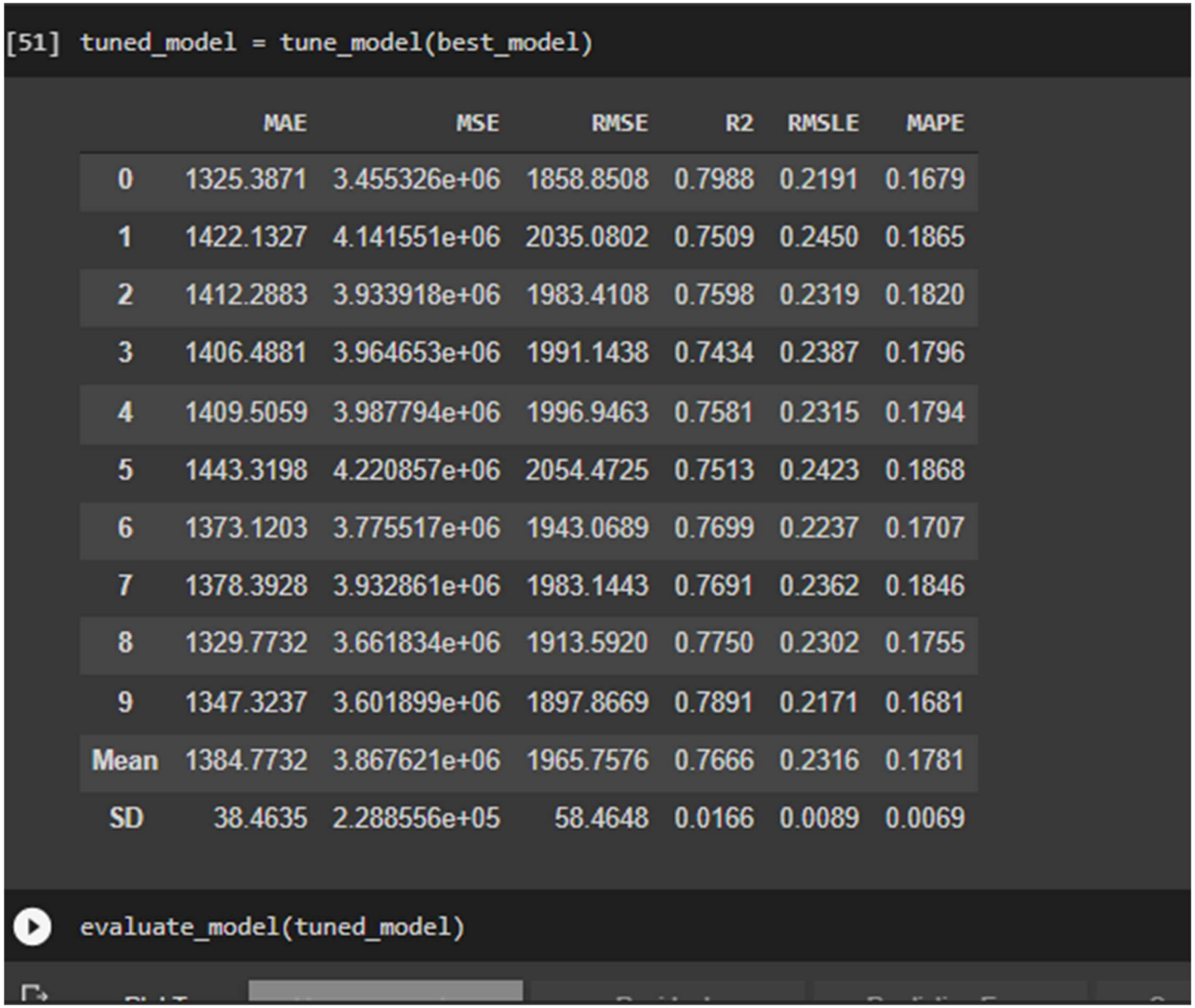
Once the model is evaluated, we may check the predictions made by the model before we go for the model tuning.

```
# create copy of data drop target column
Data = Data_Day.copy()
Data.drop('Price', axis=1, inplace=True)
# generate predictions
predictions = predict_model(best_model, data = Data)
predictions
```

	Destination	Route	Total_Stops	Additional_Info	Date	Month	Day of Week	Label
0	5	18	4	No info	24	3	6	5630.663288
1	0	84	1	No info	5	1	5	9691.785927
2	1	118	1	No info	6	9	4	13219.038886
3	0	91	0	No info	5	12	3	6860.473581
4	5	29	0	No info	3	1	3	13540.603832
...	...	...	...	...	...	...	...	...
10678	0	64	4	No info	4	9	2	4230.171000
10679	0	64	4	No info	27	4	5	4347.993843
10680	2	18	4	No info	27	4	5	4987.284584
10681	5	18	4	No info	3	1	3	13581.518679
10682	1	108	1	No info	5	9	3	11988.735741

10344 rows × 8 columns

Once the prediction is made, we may again go for the model tuning i.e., finding best parameters for the model to make more accurate predictions. Later the tuned model is reevaluated & predictions were made.



After the same we went with learn & NG model as well. Catboost & GP learn were the best fitted & most accurate models amongst the model trained.

So far what we have seen is training and model selection for all the available models in PyCaret. However, the way PyCaret works for custom models is exactly the same. As long as, your estimator is compatible with sklearn API style, it will work the same way. Let's see few examples.

GPLearn Models - While Genetic Programming (GP) can be used to perform a very wide variety of tasks, gplearn is purposefully constrained to solving symbolic regression problems. Symbolic regression is a machine learning technique that aims to identify an underlying mathematical expression that best describes a relationship. It begins by building a population of naive random formulas to represent a relationship between known independent variables and their dependent variable targets to predict new data. Each successive generation of programs is then evolved from the one that came before it by selecting the fittest individuals from the population to undergo genetic operations.

```
# install gplearn
!pip install gplearn
```

```
Collecting gplearn
  Downloading gplearn-0.4.1-py3-none-any.whl (41 kB)
     |████████████████████████████████| 41 kB 312 kB/s
Requirement already satisfied: joblib>=0.13.0 in /usr/local/lib/python3.7/dist-packages (from gplearn) (1.1.0)
Requirement already satisfied: scikit-learn>=0.20.0 in /usr/local/lib/python3.7/dist-packages (from gplearn) (0.23.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->gplearn) (3.0.0)
Requirement already satisfied: scipy>=0.19.1 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->gplearn) (1.5.4)
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->gplearn) (1.19.5)
Installing collected packages: gplearn
Successfully installed gplearn-0.4.1
```

```
# import untrained estimator
from gplearn.genetic import SymbolicRegressor
sc = SymbolicRegressor()
# train using create_model
sc_trained = create_model(sc)
```

	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	3576.1658	2.501252e+07	5001.2515	-0.4562	1.3410	0.4667
1	4076.2319	2.875887e+07	5362.7295	-0.7299	1.1302	0.5532
2	3862.1934	2.768540e+07	5261.6918	-0.6904	1.6089	0.5210
3	3580.0700	2.361830e+07	4859.8658	-0.5286	1.2186	0.4841
4	3562.8223	2.507121e+07	5007.1157	-0.5210	1.1792	0.4900

4. References

1. For Data Cleaning and Model Development - https://edwisor.com/career-data-scientist
2. For other code related - Analyticsvidya.com
3. For Visualization & Model deployment – Medium.com
4. Model evaluation – TDS & TAI