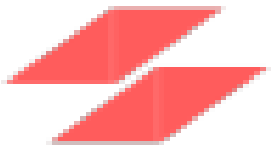


FLIGHT PRICE PREDICTION PROJECT

Submitted By:-

RISHABH JOHRI

DATA SCIENCE-INTERN

ACKNOWLEDGEMENT:-

It is a great pleasure to express my gratitude to Team Flip Robo, for giving me the opportunity to work on a interesting project, which helped me in improving my knowledge, coding skills and my analysation skills.

Team Flip Robo also gave me opportunity to build PowerPoint Presentation and Project Report, which will help me to share steps taken while building the entire model. It has helped me in deciding about the future prospects of various Data Science fields. Now, I will explain the understanding of the project through this report.

INTRODUCTION OF PROJECT :-

The tourism industry is changing fast and this is attracting a lot more travellers each year. The airline industry is considered as one of the most sophisticated industry in using complex pricing strategies. Nowadays, flight prices are quite unpredictable. The ticket prices change frequently. Customers are seeking to get the lowest price for their ticket, while airline companies are trying to keep their overall revenue as high as possible. Using technology, it is actually possible to reduce the uncertainty of flight prices. So here we will be predicting the flight prices using efficient machine learning techniques.

Airline : The Name of flight.

Travel Date: The date when the journey starts from the source.

From: From Which destination to fly.

To : The destination where to arrive

Departure Time :- Time when the flight takes off.

Arrival Time :- Time when the flight arrives at the destination.

Stops:- Number of layovers in between reaching destination.

Price: The price of the ticket.

MOTIVATION FOR THE PROBLEM UNDERTAKEN:-

For Modelling this dataset, Flight Price Prediction with all given available independent variables. This model will then be used for management of how the customer will be able to spend money on high priced tickets based on the independent variables. With the help of this prediction model, it will be decided accordingly and manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be prediction based insights to the management to understand whether the customer will pay the suitable price as compared to high priced flight Fares.

Importing Libraries:-

Here, we are importing all the libraries which are required for EDA, visualization, prediction and finding all matrics. The reason of doing this is that it become easier to use all the import statement at one go and we do not require to import the statement again at each point.

```
# For importing neccessary libraries:-
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore")
```

Data Sources and their Formats:-

Now I am going to upload or read the files/datasets using pandas. For this I have used read_csv method:-

In [3]:
```
# Loading the csv file:-
data=pd.read_csv("Flightsdata.csv")
```

In [4]:
```
# .head used for fetching first five rows of the dataset:-
data.head()
```

Out[4]:

	Unnamed: 0	Airlines	Travel_date	From	To	Departuretime	Arrivaltime	Stops	Price
0	0	AirAsia	01/03/2022	NewDelhi	Mumbai	08:20	14:10	1stopviaRanchi	5953
1	1	AirAsia	01/03/2022	NewDelhi	Mumbai	08:20	14:10	1stopviaRanchi	5953
2	2	AirAsia	01/03/2022	NewDelhi	Mumbai	20:00	02:25	1stopviaGoa	5953
3	3	AirAsia	01/03/2022	NewDelhi	Mumbai	12:20	02:25	1stopviaGoa	5953
4	4	AirAsia	01/03/2022	NewDelhi	Mumbai	20:45	07:15	1stopviaBengaluru	5953

Some EDA steps:-

1. For checking the rows and columns present in the dataset
Command Used:- data.shape

2. For Checking the null values in the dataset:-
Command used:- data.isnull().sum()

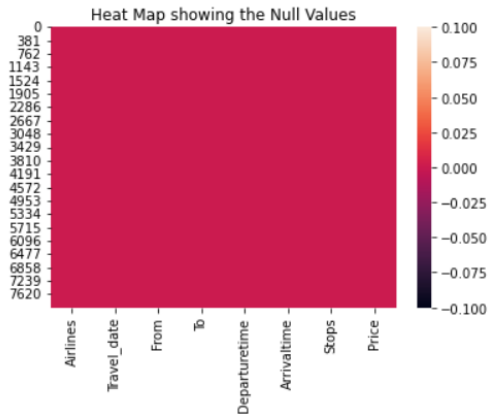
3. For checking the available columns in the dataset:
Command used:- data.columns

4. FOR CHECKING THE DATATYPE OF EACH FEATURES:-
Command Used:- data.dtypes()

5. FOR OBSERVING THE INFORMATION ABOUT DATASET:-
Command Used :- data.info()

DATA VISUALISATIONS:-

In [15]:
```
# For visualizing presence of null values using heatmap:-
sns.heatmap(data.isnull())
plt.title("Heat Map showing the Null Values")
plt.show()
```



In [16]:
```
# For checking the value counts in all the columns of the dataset:-
for i in data.columns:
    print(data[i].value_counts())
    print('----------------')
```

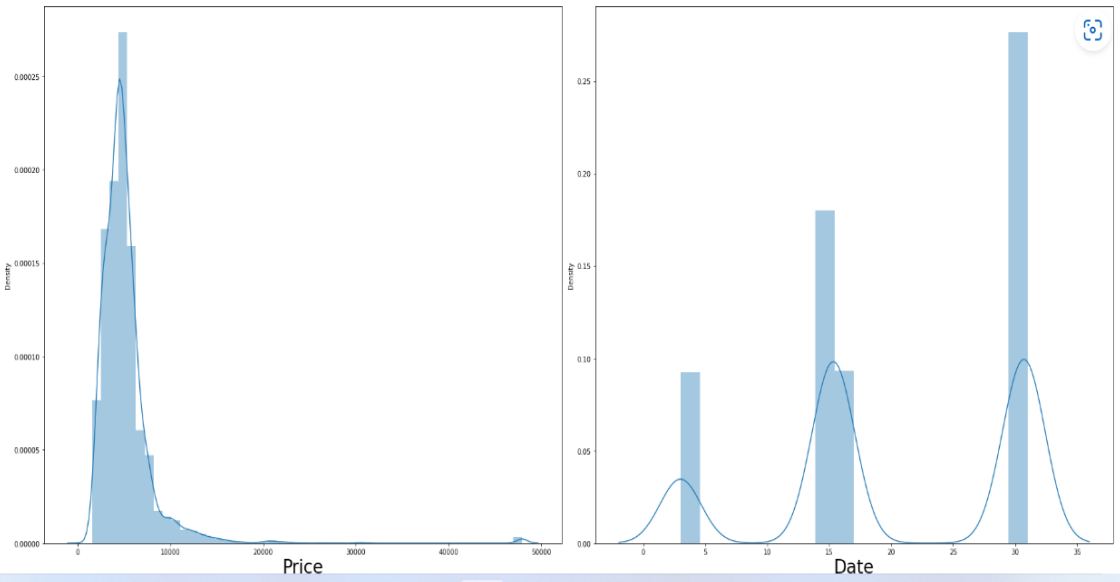
```
IndiGo              2757
GoFirst             1942
Vistara             1039
AirIndia             991
AirAsia              701
SpiceJet             417
IndiGo,GoFirst        23
```

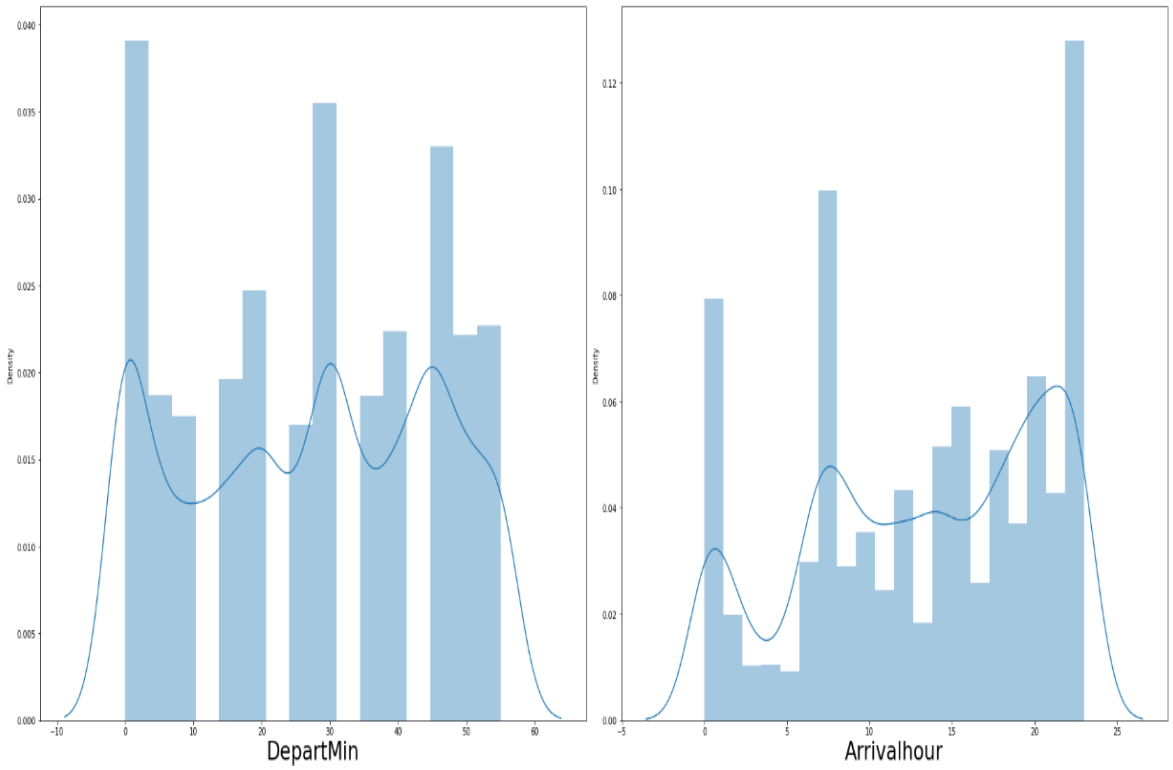
For Unique Values:-

In [18]:
```
# For checking unique values in airline column:-
data.Airlines.unique()
```

Out[18]:
```
array(['AirAsia', 'GoFirst', 'IndiGo', 'SpiceJet', 'AirIndia', 'Vistara',
       'AirIndia,GoFirst', 'IndiGo,GoFirst', 'Vistara,GoFirst', 'StarAir',
       'GoFirst,IndiGo', 'SpiceJet,IndiGo', 'IndiGo,AirAsia',
       'IndiGo,Vistara', 'AirAsia,GoFirst', 'Vistara,IndiGo',
       'AirAsia,IndiGo', 'SpiceJet,GoFirst', 'IndiGo,AirIndia',
       'AirIndia,IndiGo', 'StarAir,IndiGo', 'GoFirst,AirIndia',
       'Vistara,AirAsia', 'AirAsia,SpiceJet', 'SpiceJet,AirAsia',
       'IndiGo,SpiceJet', 'GoFirst,Vistara', 'IndiGo,StarAir',
       'SpiceJet,AirIndia', 'AirAsia,AirIndia', 'Vistara,AirIndia',
       'AirIndia,SpiceJet', 'AirIndia,AirAsia', 'GoFirst,AirAsia'],
      dtype=object)
```

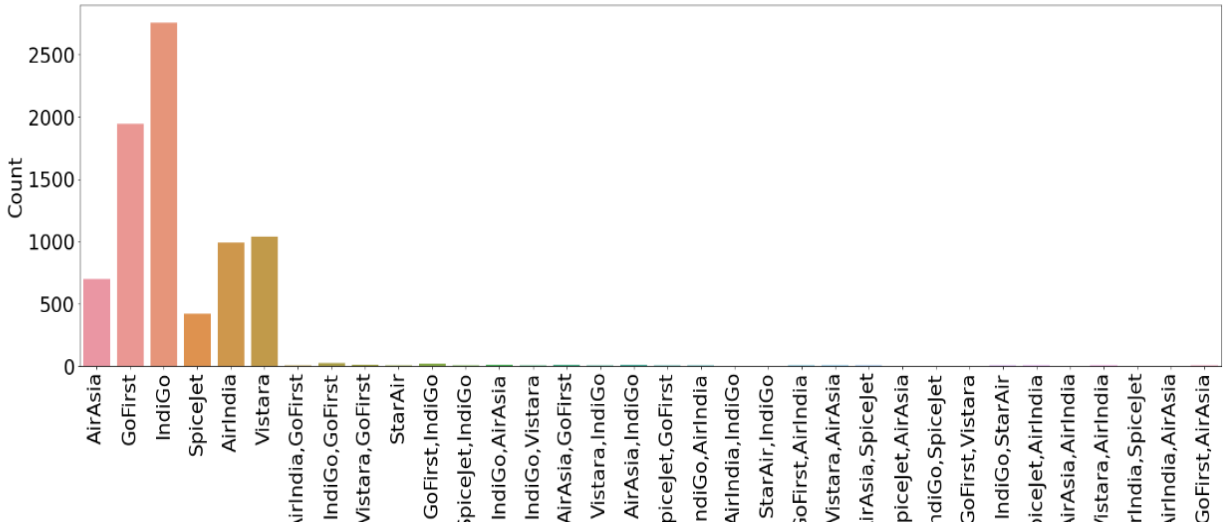
In [39]:
```
# For visualising all numerical columns using distribution plots:-
plt.figure(figsize=(25,45))
no=1
for i in data[integer]:
    if no <=7:
        ax=plt.subplot(4,2,no)
        sns.distplot(data[i])
        plt.xlabel(i,fontsize=27)
    no+=1
plt.tight_layout()
```





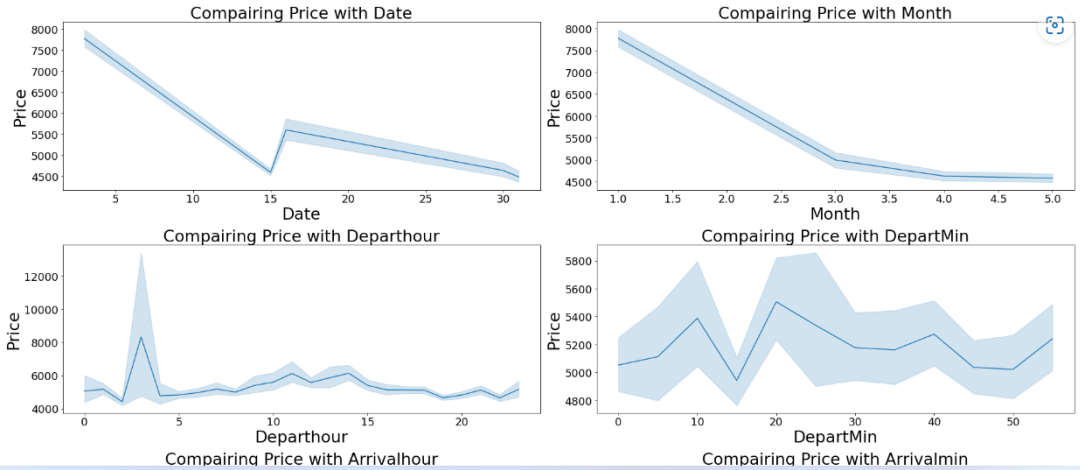
For Visualising Categorical columns:-

In [40]:
```
# For Visualizing all categorical columns using count plot:-
plt.figure(figsize=(25,45))
no=1
for i in data[categorical]:
    if no<=4:
        ax=plt.subplot(4,1,no)
        sns.countplot(data[i])
        plt.xticks(rotation=90, fontsize=27)
        plt.yticks(fontsize=27)
        plt.xlabel(i,fontsize=27)
        plt.ylabel("Count",fontsize=27)
    no+=1
plt.tight_layout()
```



For Comparing Date with Target column:-

In [48]:
```
# For comparing Date with our target column Price:-
plt.figure(figsize=(25,20))
for i in range(len(integercol)):
    plt.subplot(4,2,i+1)
    sns.lineplot(x=data[integercol[i]], y=data['Price'])
    plt.title(f"Compairing Price with {integercol[i]}", fontsize=27)
    plt.xticks(fontsize=18)
    plt.yticks(fontsize=18)
    plt.xlabel(integercol[i],fontsize=27)
    plt.ylabel('Price', fontsize=27)
    plt.tight_layout()
```



Correlation:-

In [52]:
```
# For checking correlation among dataset:-
corr=data.corr()
```

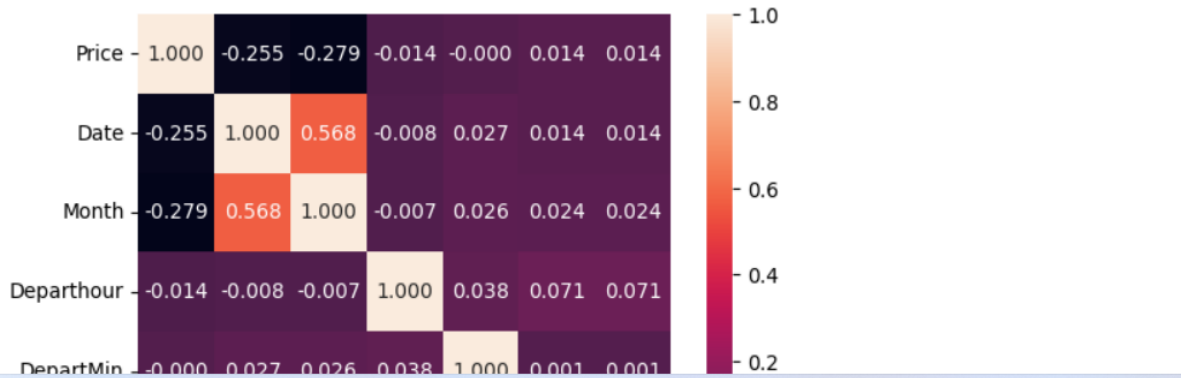
In [53]:
```
corr
```

Out[53]:

	Price	Date	Month	Departhour	DepartMin	Arrivalhour	Arrivalmin
Price	1.000000	-0.254872	-0.278998	-0.014346	-0.000409	0.014264	0.014264
Date	-0.254872	1.000000	0.568062	-0.008190	0.026602	0.013613	0.013613
Month	-0.278998	0.568062	1.000000	-0.007432	0.026319	0.024015	0.024015
Departhour	-0.014346	-0.008190	-0.007432	1.000000	0.037625	0.071158	0.071158
DepartMin	-0.000409	0.026602	0.026319	0.037625	1.000000	0.001410	0.001410
Arrivalhour	0.014264	0.013613	0.024015	0.071158	0.001410	1.000000	1.000000
Arrivalmin	0.014264	0.013613	0.024015	0.071158	0.001410	1.000000	1.000000

In [54]:
```
sns.heatmap(data.corr(),annot=True, square=True, fmt='0.3f')
```

Out[54]: <AxesSubplot:>



Model Building Phase:-

Model Building:-

In [63]:
```
# For assigning values to x and y for training and testing our dataset:-
x=data.drop('Price',axis=1)
y=data['Price']
```

In [64]:
```
# For importing required libraries for scaling data :-
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x1=pd.DataFrame(scaler.fit_transform(x),columns=x.columns)
x1.head()
```

Out[64]:

	Airlines	From	To	Stops	Date	Month	Departhour	DepartMin	Arrivalhour	Arrivalmin
0	-1.661122	0.908435	0.277387	-0.502629	-1.714481	-1.739882	-0.815339	-0.408182	0.103407	0.103407
1	-1.661122	0.908435	0.277387	-0.502629	-1.714481	-1.739882	-0.815339	-0.408182	0.103407	0.103407
2	-1.661122	0.908435	0.277387	-1.055217	-1.714481	-1.739882	1.090648	-1.531946	-1.586403	-1.586403
3	-1.661122	0.908435	0.277387	-1.055217	-1.714481	-1.739882	-0.180010	-0.408182	-1.586403	-1.586403
4	-1.661122	0.908435	0.277387	-1.231040	-1.714481	-1.739882	1.090648	0.996524	-0.882315	-0.882315

Data has been scaled properly.

In [65]:
```
# For finding Best Random state and Accuracy and importing all required libraries for model selection:-
from sklearn.metrics import accuracy_score, r2_score
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

In [66]:
```
Max_acc=0
Max_rs=0
for i in range(0,200):
    x_train,x_test,y_train,y_test=train_test_split(x1,y,test_size=.30,random_state=i)
    Model=RandomForestRegressor()
```

Different Model Scores:-

In [72]:
```
# For importing all required libraries for model selection:-
from sklearn.neighbors import KNeighborsRegressor as KNN
from sklearn.svm import SVR
from sklearn.ensemble import ExtraTreesRegressor, AdaBoostRegressor, GradientBoostingRegressor
from sklearn.linear_model import Lasso,Ridge,ElasticNet, LinearRegression
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
```

In [75]:
```
ModelB=[LinearRegression(),KNN(),SVR(),RandomForestRegressor(),ExtraTreesRegressor(),AdaBoostRegressor(),GradientBoostingRegressc
for i in ModelB:
    i.fit(x_train,y_train)
    pred=i.predict(x_test)
    print("Accuracy Score :",i,"is", i.score(x_train,y_train))
    print("\nError")
    print("Mean Absolute Error: ", mean_absolute_error(y_test,pred))
    print("Root mean Squared Error: ", (mean_squared_error(y_test,pred))**0.5)
    print("r2 Score: ",r2_score(y_test,pred))

    print("----------")
    print("\n\n")
```

```
Accuracy Score : LinearRegression() is 0.1291232020712504

Error
Mean Absolute Error:  1521.8436214282576
Root mean Squared Error:  3191.843739970436
r2 Score:  0.16555433728544744
----------



Accuracy Score : KNeighborsRegressor() is 0.5055308720871929

Error
Mean Absolute Error:  1266.3557315548144
Root mean Squared Error:  3016.0471038494406
```

```
Accuracy Score : SVR() is -0.003201870631363235

Error
Mean Absolute Error:  1692.1421859933428
Root mean Squared Error:  3508.3864964192708
r2 Score:  -0.008160504111505817
----------



Accuracy Score : RandomForestRegressor() is 0.9100061617445917

Error
Mean Absolute Error:  824.0896917068619
Root mean Squared Error:  2122.894659002022
r2 Score:  0.6308764901717837
----------
```

```
Accuracy Score : ExtraTreesRegressor() is 0.9993582531168671

Error
Mean Absolute Error:  716.7669237182159
Root mean Squared Error:  1920.5125399891092
r2 Score:  0.6979011219469474
----------



Accuracy Score : AdaBoostRegressor() is 0.018435109530423488

Error
Mean Absolute Error:  1909.9198317774924
Root mean Squared Error:  3533.5954047583627
r2 Score:  -0.02270048251950274
----------



Accuracy Score : GradientBoostingRegressor() is 0.3413450259299502

Error
Mean Absolute Error:  1194.57165111527
Root mean Squared Error:  2795.2293918538885
r2 Score:  0.36004457286654024
----------
```

Cross Validation Phase:-

Cross Validation Phase:-

In [76]:
```
# For importing required libraries for cross validation:-
from sklearn.model_selection import cross_val_score
for j in ModelB:
    cvs=cross_val_score(j,x_train,y_train,cv=15).mean()
    print("Score of ",j, "is", cvs)
```

```
Score of  LinearRegression() is 0.13305879841739204
Score of  KNeighborsRegressor() is 0.22093116706389848
Score of  SVR() is -0.005302407812317745
Score of  RandomForestRegressor() is 0.36021118885038017
Score of  ExtraTreesRegressor() is 0.4425417741307501
Score of  AdaBoostRegressor() is -0.16336972815306158
Score of  GradientBoostingRegressor() is 0.2813503638859759
Score of  Lasso() is 0.1331567665519984
Score of  Ridge() is 0.1331272865715434
Score of  ElasticNet() is 0.12192180646303434
```

So, Based on R2 score and cross validation score, ExtraTreesRegressor is giving least difference,So, ExtraTreesRegressor is our best model and will hypertune it for best accuracy.

Hyper Parameter Tuning:-

Hyper Parameter Tuning:-

In [77]:
```
# For importing all required libraries:-
from sklearn.model_selection import GridSearchCV
```

In [78]:
```
ExtraT=ExtraTreesRegressor()
parameter={'n_estimators':[10,50,100],'max_depth':[2,8,16],'criterion':['mse','mae'],'max_features':['auto','sqrt'],'random_state
search=GridSearchCV(ExtraT,parameter)
search.fit(x_train,y_train)
```

Out[78]:
```
GridSearchCV(estimator=ExtraTreesRegressor(),
             param_grid={'criterion': ['mse', 'mae'], 'max_depth': [2, 8, 16],
                         'max_features': ['auto', 'sqrt'],
                         'n_estimators': [10, 50, 100], 'random_state': [223]})
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [79]:
```
print(search.best_params_)
```

```
{'criterion': 'mae', 'max_depth': 16, 'max_features': 'auto', 'n_estimators': 100, 'random_state': 223}
```

In [81]:
```
ExtraTR=ExtraTreesRegressor(n_estimators=100, max_depth=16, criterion='mae',max_features='auto',random_state=223)
ExtraTR.fit(x_train,y_train)
ExtraTR.score(x_train,y_train)
pred=ExtraTR.predict(x_test)
print("Accuracy Score: ",r2_score(y_test,pred))
print("Mean Absolute Error: ",mean_absolute_error(y_test,pred))
print("Root Mean Squared Error: ",(mean_squared_error(y_test,pred)**0.5))
```

```
Accuracy Score:  0.7074952211639535
Mean Absolute Error:  688.4475510629428
Root Mean Squared Error:  1889.7705380588582
```

Saving and Loading Predictions:-

Saving the Model:- ¶

In [82]:
```
# For importing librariesfor saving model:-
import pickle
Name="FlightPricePredictions.pkl"
pickle.dump(ExtraTR,open(Name,'wb'))
```

Loading the Model:-

In [83]:
```
# For loading model:-
Models=pickle.load(open(Name,'rb'))
```

In [84]:
```
# For displaying predictions:-
prediction=Models.predict(x_test)
prediction
```

Out[84]:
```
array([6924.78 , 4723.44 , 3945.925, ..., 2531.155, 2003.12 , 3011.77 ])
```

In [86]:
```
pd.DataFrame([Models.predict(x_test)[:],y_test[:]],index=["Predicted","Actual"])
```

Out[86]:

	0	1	2	3	4	5	6	7	8	9	...	2389	2390	2391	2392	2393
Predicted	6924.78	4723.44	3945.925	5979.135	6098.555	3453.015	9268.04	3727.645	4384.095	3413.56	...	3000.98	4495.865	6352.44	6295.345	3344.305
Actual	5953.00	4266.00	3861.000	5966.000	4413.000	3441.000	11721.00	3228.000	4052.000	3000.00	...	3001.00	4500.000	6894.00	7081.000	3540.000

2 rows × 2399 columns

Interpretation of the Results:-

- I have used visualization tools such as dist Plot, Count Plot for categorical data and line plot to understand the data in a better way.
- I have done the model building process with several algorithms and the best model is Extra Trees Regressor with an accuracy score of 71% after Hyper Parameter Tuning.

CONCLUSION:-

The overall survey for the dynamic price changes in the flight tickets is presented. This gives the information about the ups and downs in the airfares according to the days, weekend and time of the day that is morning, evening and night. Also, the machine learning models in the computational intelligence field that are evaluated before on different datasets are studied. Their accuracy and performances are evaluated and compared in order to get better result. For the prediction of the ticket prices perfectly different prediction models are tested for the better prediction accuracy. As the pricing models of the company are developed in order to maximize the revenue management, so to get the result with maximum accuracy regression analysis is used. From the studies , the feature that influences the prices of the ticket are to be considered. In future, the details about number of available seats can improve the performance of the model.