Flight price prediction

Submitted by:

Dumpala Aditya

ACKNOWLEDGMENT

References which helps me in completion of this project:

- W3schools
- Analyticalvidya
- Geeksforweeks
- Google

INTRODUCTION

- Business Problem Framing

  Anyone who has booked a flight ticket knows how unexpectedly the prices vary. The cheapest available ticket on a given flight gets more and less expensive over time. This usually happens as an attempt to maximize revenue based on - 1. Time of purchase patterns (making sure last-minute purchases are expensive) 2. Keeping the flight as full as they want it (raising prices on a flight which is filling up in order to reduce sales and hold back inventory for those expensive last-minute expensive purchases)

- Conceptual Background of the Domain Problem

  Basically it is completely related to flights and their and also it has some features which are more important to predict the price of the flight they are

  - Flight name
  - Source(starting point)
  - Destination(end point)
  - Travel time
  - No of stops
  - Start time
  - End time
  - Class
  - Date of journey
  - price

- Review of Literature

This is a regression problem I scrape the data from various sites they are easemytrip,tripodeal etc following are the steps which I follow while building this project,

- Scraping the data by using selenium
- Data cleaning
- Data preprocessing
- Scaling the data
- Applying different types of models
- Choosing best model
- Tunning the selected model
- Testing the model by test data
- Save the model

- Motivation for the Problem Undertaken

  Anyone who has booked a flight ticket knows how unexpectedly the prices vary. The cheapest available ticket on a given flight gets more and less expensive over time. To over come this problem I decided that to build the machine learning model

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

Out put of this data set is continuous in the nature so we have to use regression models so I am applying regression models only,

Price is skewed right side because since some of the flight ticket price is very high some are not so it is skewed right side and travel time also skewed since no of stops increases time will also increases and also outliers in the features are high but we cannot consider those data is belongs to outliers because it is alsdo an important data and also it makes sense so I am not remove any outliers from the data,but some features are highly correlated I drop those features from the data set

- Data Sources and their formats

I am scraping the data from various sites like easemytrip,tripodeal etc.

All the features are in the same format irrespective of their data i.e object so I convert those features into their respective data type like price is in the form of object type but it should be in the form of numeric type so I convert it into float because it is in the form of continuous and also I change the remaining features also.

- Data Preprocessing Done

Steps which I followed while data preprocessing

- Checking null values
  Result:- no null values are present

- Checking data types
  Result:- all the features are in the same type i.e object type so I change the data types according to data which is present in the features
- Extracting day and month from date feature for better understanding
- I extract hours and minutes from start time and end time for analysis purpose

- Checking skewness, duration hour and price are skewed data but price is our target variable we cannot change anything in the target variable so I am not modifying the target variable but I remove skewness in duration hours feature.
- Checking outliers, and outliers are makes sense so iam not going to remove those outliers
- Encoding the data by using
- Checking correlation and remove highly correlated features
- Dividing the features and label as x and y
- Scaling the data by using minmax scaler

- Data Inputs- Logic- Output Relationships

  Describe the relationship behind the data input, its format, the logic in between and the output. Describe how the input affects the output.

  Relation with price and remaining features

  - In the flight name Vistara has high price.
  - The ticket price is high in the months of 6, 9, 11
  - In the class business class high price
  - The price is high in the no of stops is 1 and 2 and 0 stops is less price.
  - Source Delhi, Goa, Bengulore and kolkata has high price
  - Price is high in destination in the cities of Hyderabad, bengulore and goa
  - If duration hours increases price will also increases

- Hardware and Software Requirements and Tools Used

  In this project hardware and software requirements are below

  Processor:-i5

  Ram:-32gb

  Operating system:-Windows10

  Software packages that I used are

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
pd.set_option('display.max_columns',None)
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error
from sklearn.ensemble import RandomForestRegressor,GradientBoostingRegressor, GradientBoostingRegressor
from xgboost import XGBRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Lasso
import joblib
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  ➔ Our target is continuous so regression model is suitable for this problem so I am checking with all the regression models and pick up the best model from all the models

  ➔ some of the features are skewed right side we have to convert them into normally distributed so I can apply log transformation to convert into normally distributed
- Testing of Identified Approaches (Algorithms)

  ➔LinearRegression

  ➔RandomForestRegressor

  ➔GradientBoostingRegressor

  ➔ BaggingRegressor

  ➔ DecisionTreeRegressor

  ➔ XGBRegressor

- Run and Evaluate selected models

  Describe all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.

  Linear regression model:-

```
#linear regression
lr=LinearRegression()
lr.fit(x_train,y_train)

print_score(lr,x_train,x_test,y_train,y_test,train=True)
print_score(lr,x_train,x_test,y_train,y_test,train=False)

kfolds(lr,'linearRegression')
```

```
=======train results===========


accuracy score:88.156620%


=======test results=========


r2score is:92.209911%


linearRegression score on cross validation: 3.5684500133564443%
```

  2)Random forest model:-

```
#random Forest
from sklearn.ensemble import RandomForestRegressor
rfc=RandomForestRegressor()
rfc.fit(x_train,y_train)

print_score(rfc,x_train,x_test,y_train,y_test,train=True)
print_score(rfc,x_train,x_test,y_train,y_test,train=False)

kfolds(rfc,'RandomForestRegressor')
```

```
=======train results===========


accuracy score:99.430319%


=======test results=========


r2score is:99.690018%


RandomForestRegressor score on cross validation: 42.42442492709513%
```

3)gradient boosting regressor

```
#GradientBoostingRegressor
from sklearn.ensemble import GradientBoostingRegressor
gbr=GradientBoostingRegressor()
gbr.fit(x_train,y_train)

print_score(gbr,x_train,x_test,y_train,y_test,train=True)
print_score(gbr,x_train,x_test,y_train,y_test,train=False)

kfolds(gbr,'GradientBoostingRegressor')
```

```
=======train results===========


accuracy score:95.361775%


=======test results=========


r2score is:97.289242%


GradientBoostingRegressor score on cross validation: 55.45127553504513%
```

4)Bagging regressor

```
#BaggingRegressor
from sklearn.ensemble import BaggingRegressor
bgr=BaggingRegressor()
bgr.fit(x_train,y_train)


print_score(bgr,x_train,x_test,y_train,y_test,train=True)
print_score(bgr,x_train,x_test,y_train,y_test,train=False)

kfolds(bgr,'BaggingRegressor')
```

```
=======train results===========


accuracy score:99.276332%


=======test results=========


r2score is:99.449350%


BaggingRegressor score on cross validation: 42.67548011376949%
```

5)Decision tree regressor

```
#DecisionTreeRegressor
from sklearn.tree import DecisionTreeRegressor
dtr=DecisionTreeRegressor()
dtr.fit(x_train,y_train)

print_score(dtr,x_train,x_test,y_train,y_test,train=True)
print_score(dtr,x_train,x_test,y_train,y_test,train=False)

kfolds(dtr,'DecisionTreeRegressor')
```

```
=======train results===========


accuracy score:99.980453%


=======test results=========


r2score is:100.000000%


DecisionTreeRegressor score on cross validation: 0.9756739783514999%
```

6)xgb regressor

```
from xgboost import XGBRegressor
xgbr=XGBRegressor()
xgbr.fit(x_train,y_train)

print_score(dtr,x_train,x_test,y_train,y_test,train=True)
print_score(dtr,x_train,x_test,y_train,y_test,train=False)

kfolds(xgbr,'XGBRegressor')
```

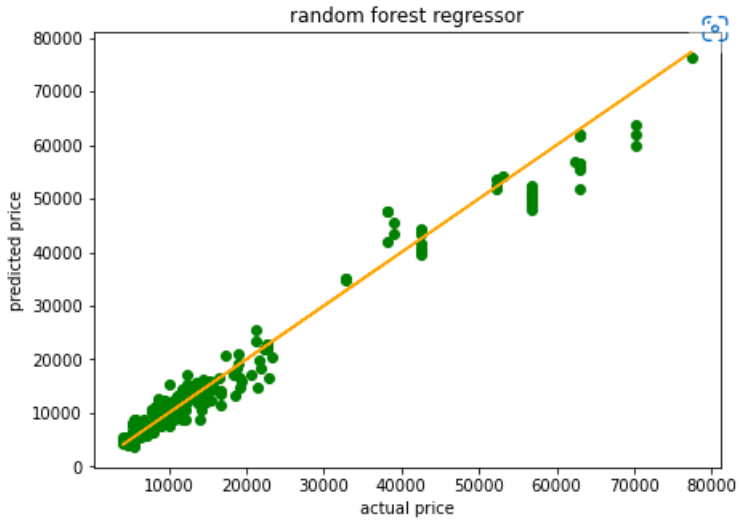
```
=======train results===========


accuracy score:99.980453%


=======test results=========


r2score is:100.000000%


XGBRegressor score on cross validation: 53.32995463202626%
```

Apart from all the models random forest and gradient boosting regressor are giving best results so we have to select the best one between them so I can check error rate from those two selected models

- random forest

```
#random forest
y_pred=gbr.predict(x_train)
pred=gbr.predict(x_test)

from sklearn.metrics import mean_absolute_error,mean_squared_error
print('mean absolute error is:',mean_absolute_error(y_test,pred))
print('\n')
print('mean squared error is:',mean_squared_error(y_test,pred))
print('\n')
print('Root mean suared error is:',np.sqrt(mean_squared_error(y_test,pred)))

plt.figure(figsize=(7,5))
plt.scatter(x=y_test,y=pred,color='green')
plt.plot(y_test,y_test,color='orange')
plt.xlabel('actual price')
plt.ylabel('predicted price')
plt.title('random forest regressor')
plt.show()
```

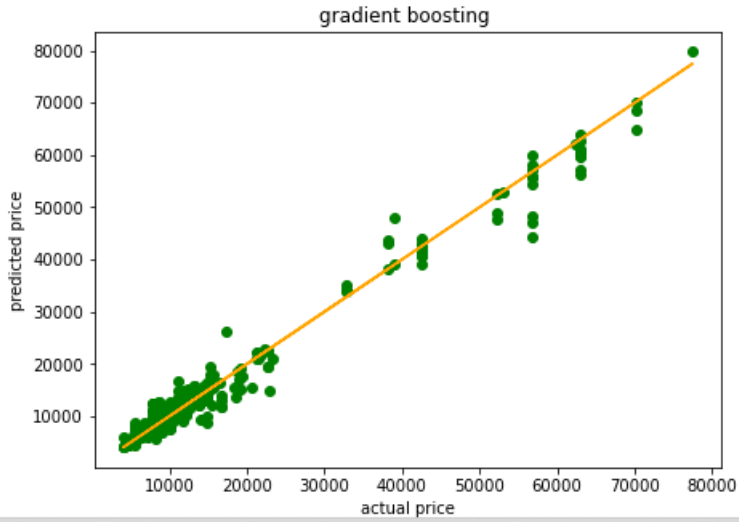
```
mean absolute error is: 1609.0346192476538


mean squared error is: 6132883.394202896


Root mean suared error is: 2476.4659081446885
```



- gradient boosting regressor

```
y_pred=xgbr.predict(x_train)
pred=xgbr.predict(x_test)

from sklearn.metrics import mean_absolute_error,mean_squared_error
print('mean absolute error is:',mean_absolute_error(y_test,pred))
print('\n')
print('mean squared error is:',mean_squared_error(y_test,pred))
print('\n')
print('Root mean suared error is:',np.sqrt(mean_squared_error(y_test,pred)))

plt.figure(figsize=(7,5))
plt.scatter(x=y_test,y=pred,color='green')
plt.plot(y_test,y_test,color='orange')
plt.xlabel('actual price')
plt.ylabel('predicted price')
plt.title('gradient boosting ')
plt.show()
```

```
mean absolute error is: 1313.5673888324059


mean squared error is: 4416368.301902109


Root mean suared error is: 2101.5157153592995
```



From these two gradient boosting is given best results so I can select gradient boosting regressor as the final model

- Key Metrics for success in solving problem under consideration

  What were the key metrics used along with justification for using it?
  You may also include statistical metrics used if any.
  Key metrics in this problem are

  - Accuracy score

- R2score
- Cross validation score
- Mean absolute error
- Mean squared error
- Root mean squared error
- Hyperparameter tunning

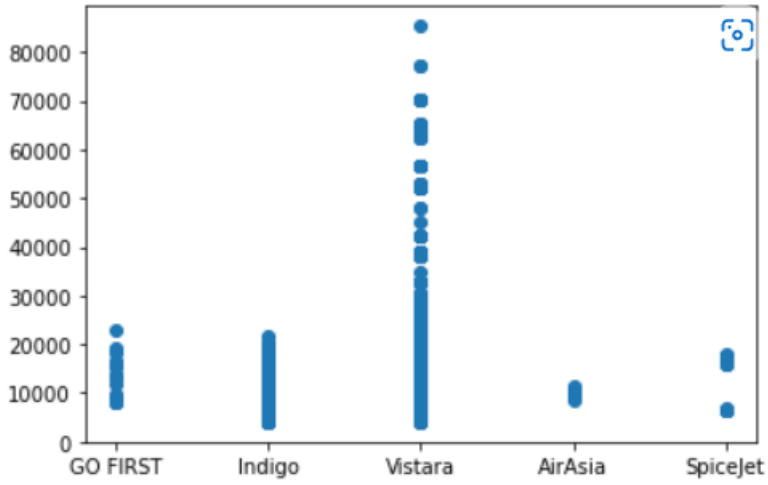
- Visualizations

  Mention all the plots made along with their pictures and what were the inferences and observations obtained from those. Describe them in detail.

  If different platforms were used, mention that as well.

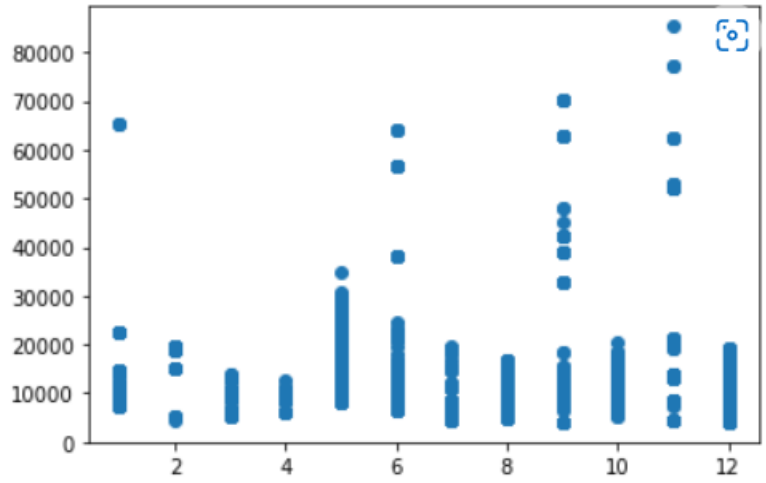
```
plt.scatter(x='flight_name',y='price',data=data)
```

```
<matplotlib.collections.PathCollection at 0x29f73af8c10>
```



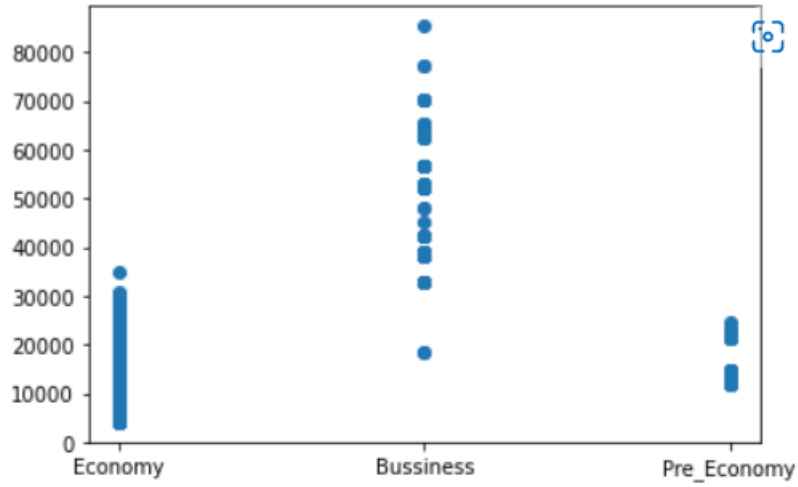
```
plt.scatter(x='month',y='price',data=data)
```

```
<matplotlib.collections.PathCollection at 0x29f73a74760>
```



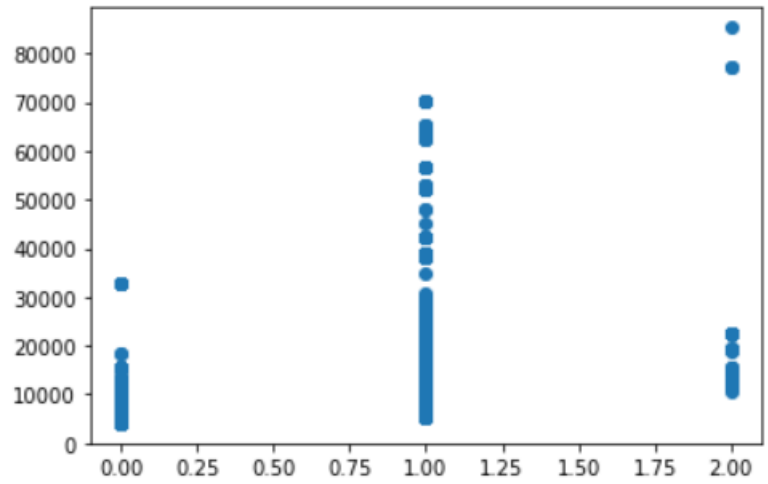

```
plt.scatter(x='class',y='price',data=data)
```

```
<matplotlib.collections.PathCollection at 0x29f74b95af0>
```



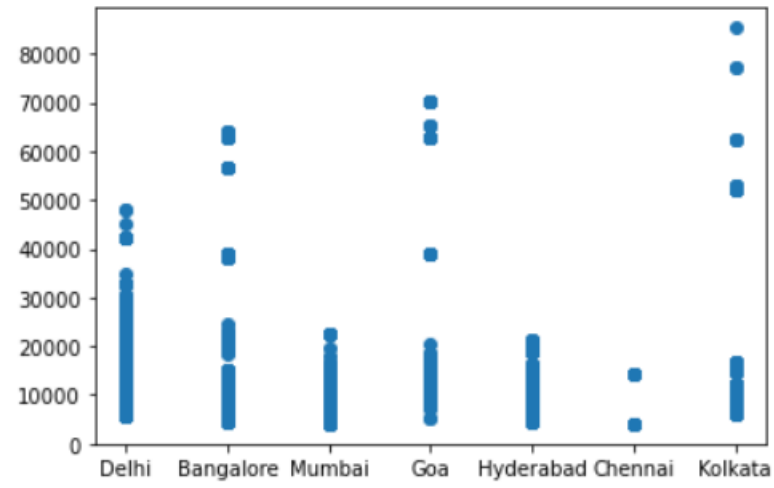
```
plt.scatter(x='no_of_stops',y='price',data=data)
```

```
<matplotlib.collections.PathCollection at 0x29f74be4c10>
```



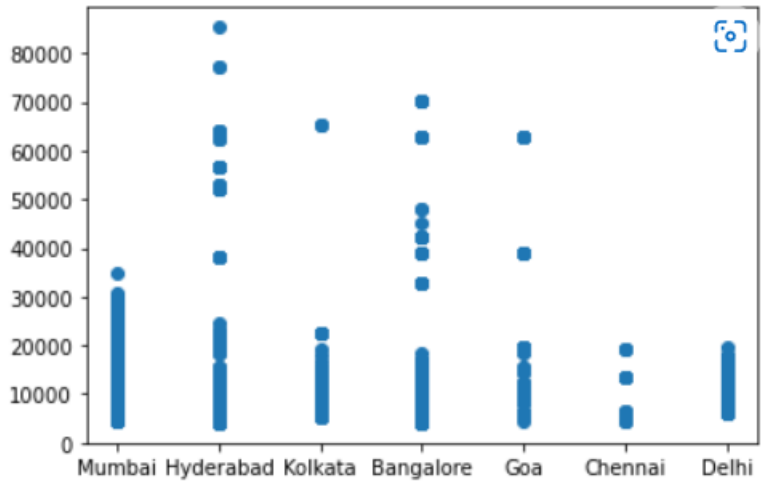

```
plt.scatter(x='source',y='price',data=data)
```

```
<matplotlib.collections.PathCollection at 0x29f74c50310>
```



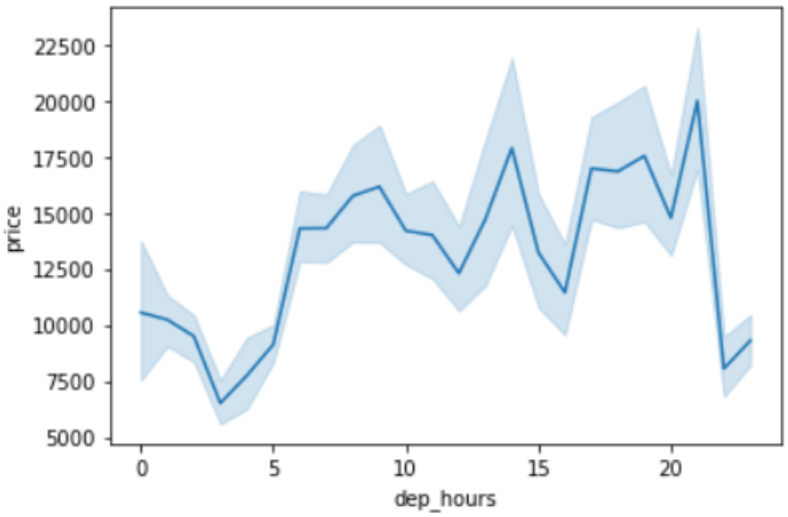
```
plt.scatter(x='destination',y='price',data=data)
```

```
<matplotlib.collections.PathCollection at 0x29f74ca9dc0>
```



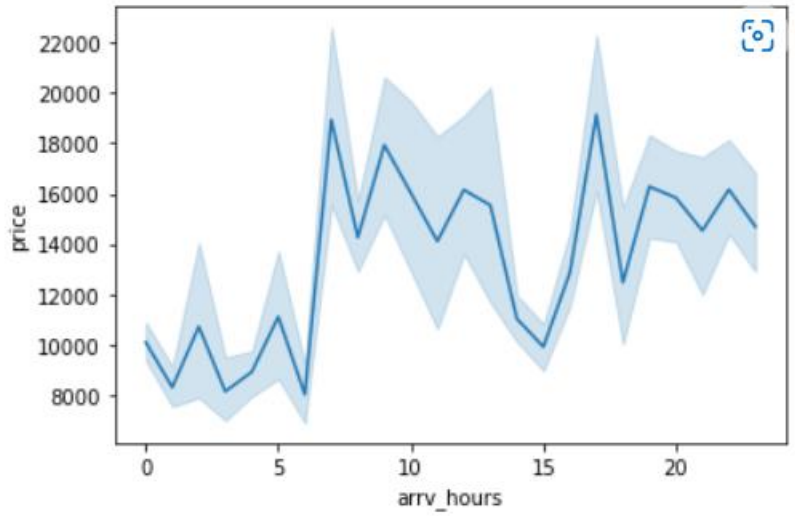

```
sns.lineplot(x='dep_hours',y='price',data=data)
```

```
<AxesSubplot:xlabel='dep_hours', ylabel='price'>
```



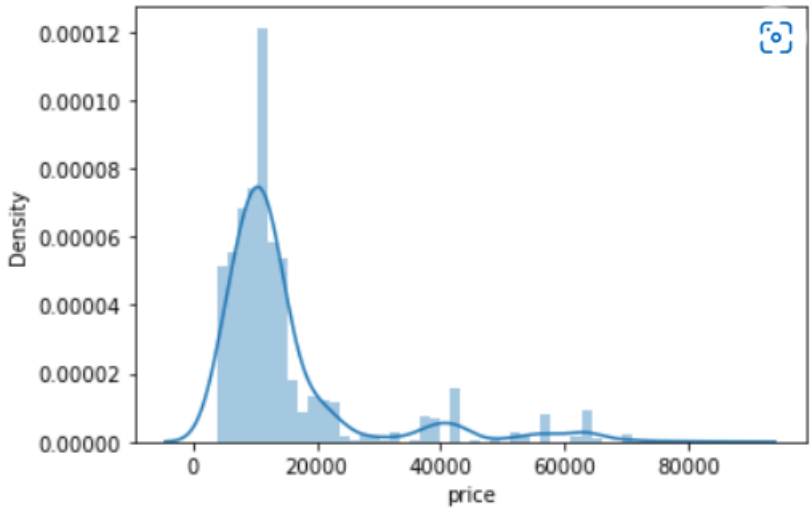
```
sns.lineplot(x='arrv_hours',y='price',data=data)
```

```
<AxesSubplot:xlabel='arrv_hours', ylabel='price'>
```




```
sns.distplot(data['price'])
```

```
<AxesSubplot:xlabel='price', ylabel='Density'>
```



- Interpretation of the Results

  The above plots all are correlates with the target variable

CONCLUSION

- Key Findings and Conclusions of the Study

Key findings in this problem are

- In the flight name feature there are 5 categories are there apart from those 5 one category is nearly 70% of the total data so there is a chance to over fitting problem
- And also cross validation score is also very low it indicates that there is a overfitting problem exists

- Learning Outcomes of the Study in respect of Data Science

In this problem I've learn so many things they are

- How to clean data which is in different format with different special characters
- Analysing the features and how they are related to target variable
- Dealing with skewness and outliers
- Checking correlation within the features
- Scaling the data by using minmax scaler
- Applying different models for testing the accuracy score and r2 score
- Regularising the selected model
- Hyperparameter tunning the model
- Save the selected model

- Limitations of this work and Scope for Future Work

What are the limitations of this solution provided, the future scope? What all steps/techniques can be followed to further extend this study and improve the results.

Limitations of this work is:-

- Prices are not same in all the time they varying day by day week by week and month by month so we cannot sure about prices but we are able to predict the price almost nearly
- In the flight ticket booking websites there is no past data is available but past data is also useful for better prediction

- Due to most of the data is belongs to only one category this model cv score is not that much good and also there is chance to improve score by giving more range in hyperparameter tunning