untitled-checkpoint

May 24, 2024

1 Flight Price Prediction

1.0.1 Importing Libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

[2]:
```
! pip install openpyxl
```

```
Requirement already satisfied: openpyxl in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (3.1.2)
Requirement already satisfied: et-xmlfile in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
openpyxl) (1.1.0)
```

1.0.2 Loading Dataset

[3]:
```
df = pd.read_excel("Data_Train.xlsx")
```

1.0.3 Data Pre-processing

[4]:
```
df.head()
```

[4]:

	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
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK						
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR						
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL						

```
2    09:25  04:25 10 Jun      19h     2 stops         No info  13882
3    18:05         23:30   5h 25m      1 stop         No info   6218
4    16:50         21:35   4h 45m      1 stop         No info  13302
```

[5]:
```
df.tail()
```

[5]:
```
           Airline Date_of_Journey    Source Destination
10678     Air Asia       9/04/2019   Kolkata    Banglore  \
10679    Air India      27/04/2019   Kolkata    Banglore
10680  Jet Airways      27/04/2019  Banglore       Delhi
10681      Vistara      01/03/2019  Banglore   New Delhi
10682    Air India       9/05/2019     Delhi      Cochin

                     Route Dep_Time Arrival_Time Duration Total_Stops
10678              CCU → BLR    19:55        22:25   2h 30m    non-stop  \
10679              CCU → BLR    20:45        23:20   2h 35m    non-stop
10680              BLR → DEL    08:20        11:20       3h    non-stop
10681              BLR → DEL    11:30        14:10   2h 40m    non-stop
10682  DEL → GOI → BOM → COK    10:55        19:15   8h 20m     2 stops

      Additional_Info  Price
10678         No info   4107
10679         No info   4145
10680         No info   7229
10681         No info  12648
10682         No info  11753
```

[6]:
```
df.drop('Date_of_Journey',axis=1,inplace=True)
```

1.0.4 Checking null values

[7]:
```
df.isnull().sum()
```

[7]:
```
Airline           0
Source            0
Destination       0
Route             1
Dep_Time          0
Arrival_Time      0
Duration          0
Total_Stops       1
Additional_Info   0
Price             0
dtype: int64
```

[8]:
```
df.dropna(inplace=True)
```

[9]:
```
df.columns
```

[9]:
```
Index(['Airline', 'Source', 'Destination', 'Route', 'Dep_Time', 'Arrival_Time',
       'Duration', 'Total_Stops', 'Additional_Info', 'Price'],
      dtype='object')
```

[10]:
```
df.isnull().any()
```

[10]:
```
Airline            False
Source             False
Destination        False
Route              False
Dep_Time           False
Arrival_Time       False
Duration           False
Total_Stops        False
Additional_Info    False
Price              False
dtype: bool
```

1.0.5 Extracting hours and minutes from departure time

[11]:
```
df[['Dep_Hours','Dep_Mins']] = df['Dep_Time'].str.split(':',expand=True)
```

[12]:
```
df.head()
```

[12]:

	Airline	Source	Destination	Route	Dep_Time	\
0	IndiGo	Banglore	New Delhi	BLR → DEL	22:20	\
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	18:05	
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	16:50	

	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Dep_Hours	Dep_Mins
0	01:10 22 Mar	2h 50m	non-stop	No info	3897	22	20
1	13:15	7h 25m	2 stops	No info	7662	05	50
2	04:25 10 Jun	19h	2 stops	No info	13882	09	25
3	23:30	5h 25m	1 stop	No info	6218	18	05
4	21:35	4h 45m	1 stop	No info	13302	16	50

[13]:
```
df.drop('Dep_Time', axis=1,inplace=True)
```

[14]:
```
df.head()
```

[14]:

	Airline	Source	Destination	Route	Arrival_Time	\
0	IndiGo	Banglore	New Delhi	BLR → DEL	01:10 22 Mar	\
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	13:15	

```
2  Jet Airways     Delhi      Cochin  DEL → LKO → BOM → COK  04:25 10 Jun
3       IndiGo   Kolkata    Banglore        CCU → NAG → BLR         23:30
4       IndiGo  Banglore   New Delhi        BLR → NAG → DEL         21:35

  Duration Total_Stops Additional_Info  Price Dep_Hours Dep_Mins
0   2h 50m    non-stop         No info   3897        22       20
1   7h 25m     2 stops         No info   7662        05       50
2      19h     2 stops         No info  13882        09       25
3   5h 25m      1 stop         No info   6218        18       05
4   4h 45m      1 stop         No info  13302        16       50
```

1.0.6 Extracting Arrival hours and mins from arrival time

[15]:
```
df['Arrival_Hour']=pd.to_datetime(df['Arrival_Time']).dt.hour
df['Arrival_min']=pd.to_datetime(df['Arrival_Time']).dt.minute
```

[16]:
```
df.drop('Arrival_Time',axis=1,inplace=True)
```

[17]:
```
duration=list(df['Duration'])
```

1.0.7 Making Separate Column for hour and min and extracting it from duration

[18]:
```
duration_hours = [int(x.split('h')[0].strip()) if 'h' in x else 0  for x in␣
 ↪df['Duration']]
duration_mins = [int(x.split('m')[0].strip()[-1].strip()) if 'm' in x else 0␣
 ↪for x in df['Duration']]
```

[19]:
```
df['Duration_hours'] = duration_hours
df['Duration_mins'] = duration_mins
```

[20]:
```
df.head()
```

[20]:
```
       Airline    Source Destination                  Route Duration  \
0       IndiGo  Banglore   New Delhi              BLR → DEL   2h 50m   \
1    Air India   Kolkata    Banglore  CCU → IXR → BBI → BLR   7h 25m
2  Jet Airways     Delhi      Cochin  DEL → LKO → BOM → COK      19h
3       IndiGo   Kolkata    Banglore        CCU → NAG → BLR   5h 25m
4       IndiGo  Banglore   New Delhi        BLR → NAG → DEL   4h 45m

  Total_Stops Additional_Info  Price Dep_Hours Dep_Mins  Arrival_Hour  \
0    non-stop         No info   3897        22       20             1   \
1     2 stops         No info   7662        05       50            13
2     2 stops         No info  13882        09       25             4
3      1 stop         No info   6218        18       05            23
4      1 stop         No info  13302        16       50            21
```

```
   Arrival_min  Duration_hours  Duration_mins
0           10               2              0
1           15               7              5
2           25              19              0
3           30               5              5
4           35               4              5
```

[21]:
```
df.drop('Duration',axis=1,inplace=True)
df.head()
```

[21]:
```
        Airline    Source Destination                  Route Total_Stops  \
0        IndiGo  Banglore   New Delhi              BLR → DEL    non-stop
1    Air India   Kolkata    Banglore  CCU → IXR → BBI → BLR     2 stops
2  Jet Airways     Delhi      Cochin  DEL → LKO → BOM → COK     2 stops
3        IndiGo   Kolkata    Banglore        CCU → NAG → BLR      1 stop
4        IndiGo  Banglore   New Delhi        BLR → NAG → DEL      1 stop

  Additional_Info  Price Dep_Hours Dep_Mins  Arrival_Hour  Arrival_min  \
0         No info   3897        22       20             1           10
1         No info   7662        05       50            13           15
2         No info  13882        09       25             4           25
3         No info   6218        18       05            23           30
4         No info  13302        16       50            21           35

   Duration_hours  Duration_mins
0               2              0
1               7              5
2              19              0
3               5              5
4               4              5
```

[22]:
```
df['Airline'].value_counts()
```

[22]:
```
Airline
Jet Airways                          3849
IndiGo                               2053
Air India                            1751
Multiple carriers                    1196
SpiceJet                              818
Vistara                               479
Air Asia                              319
GoAir                                 194
Multiple carriers Premium economy      13
Jet Airways Business                    6
Vistara Premium economy                 3
Trujet                                  1
Name: count, dtype: int64
```

1.0.8 Handling Categorical Data

[23]:
```
Airline=df[['Airline']]
Airline=pd.get_dummies(Airline,drop_first=True)
```

1.0.9 One hot encoding on source

[24]:
```
df['Source'].value_counts()
```

[24]:
```
Source
Delhi       4536
Kolkata     2871
Banglore    2197
Mumbai       697
Chennai      381
Name: count, dtype: int64
```

[25]:
```
Source=df[['Source']]
Source=pd.get_dummies(Source,drop_first=True)
Source.head()
```

[25]:

	Source_Chennai	Source_Delhi	Source_Kolkata	Source_Mumbai
0	False	False	False	False
1	False	False	True	False
2	False	True	False	False
3	False	False	True	False
4	False	False	False	False

[26]:
```
Destination=df[['Destination']]
Destination=pd.get_dummies(Destination,drop_first=True)
```

[27]:
```
df.drop(['Route','Additional_Info'],axis=1,inplace=True)
```

[28]:
```
df['Total_Stops'].value_counts()
```

[28]:
```
Total_Stops
1 stop      5625
non-stop    3491
2 stops     1520
3 stops       45
4 stops        1
Name: count, dtype: int64
```

1.0.10 Label Encoding on ordinal stops data

[29]:
```
df.replace({'non-stop':0, '1 stop':1, '2 stops':2, '3 stops':3, '4 stops':
 ↪4},inplace=True)
```

[30]:
```
df.head()
```

[30]:
```
       Airline    Source Destination  Total_Stops  Price Dep_Hours Dep_Mins  \
0       IndiGo  Banglore   New Delhi            0   3897        22       20
1    Air India   Kolkata    Banglore            2   7662        05       50
2  Jet Airways     Delhi      Cochin            2  13882        09       25
3       IndiGo   Kolkata    Banglore            1   6218        18       05
4       IndiGo  Banglore   New Delhi            1  13302        16       50

   Arrival_Hour  Arrival_min  Duration_hours  Duration_mins
0             1           10               2              0
1            13           15               7              5
2             4           25              19              0
3            23           30               5              5
4            21           35               4              5
```

[31]:
```
df_final=pd.concat([df,Airline,Source,Destination],axis=1)
df.head()
```

[31]:
```
       Airline    Source Destination  Total_Stops  Price Dep_Hours Dep_Mins  \
0       IndiGo  Banglore   New Delhi            0   3897        22       20
1    Air India   Kolkata    Banglore            2   7662        05       50
2  Jet Airways     Delhi      Cochin            2  13882        09       25
3       IndiGo   Kolkata    Banglore            1   6218        18       05
4       IndiGo  Banglore   New Delhi            1  13302        16       50

   Arrival_Hour  Arrival_min  Duration_hours  Duration_mins
0             1           10               2              0
1            13           15               7              5
2             4           25              19              0
3            23           30               5              5
4            21           35               4              5
```

[32]:
```
df_final.shape
```

[32]:
```
(10682, 31)
```

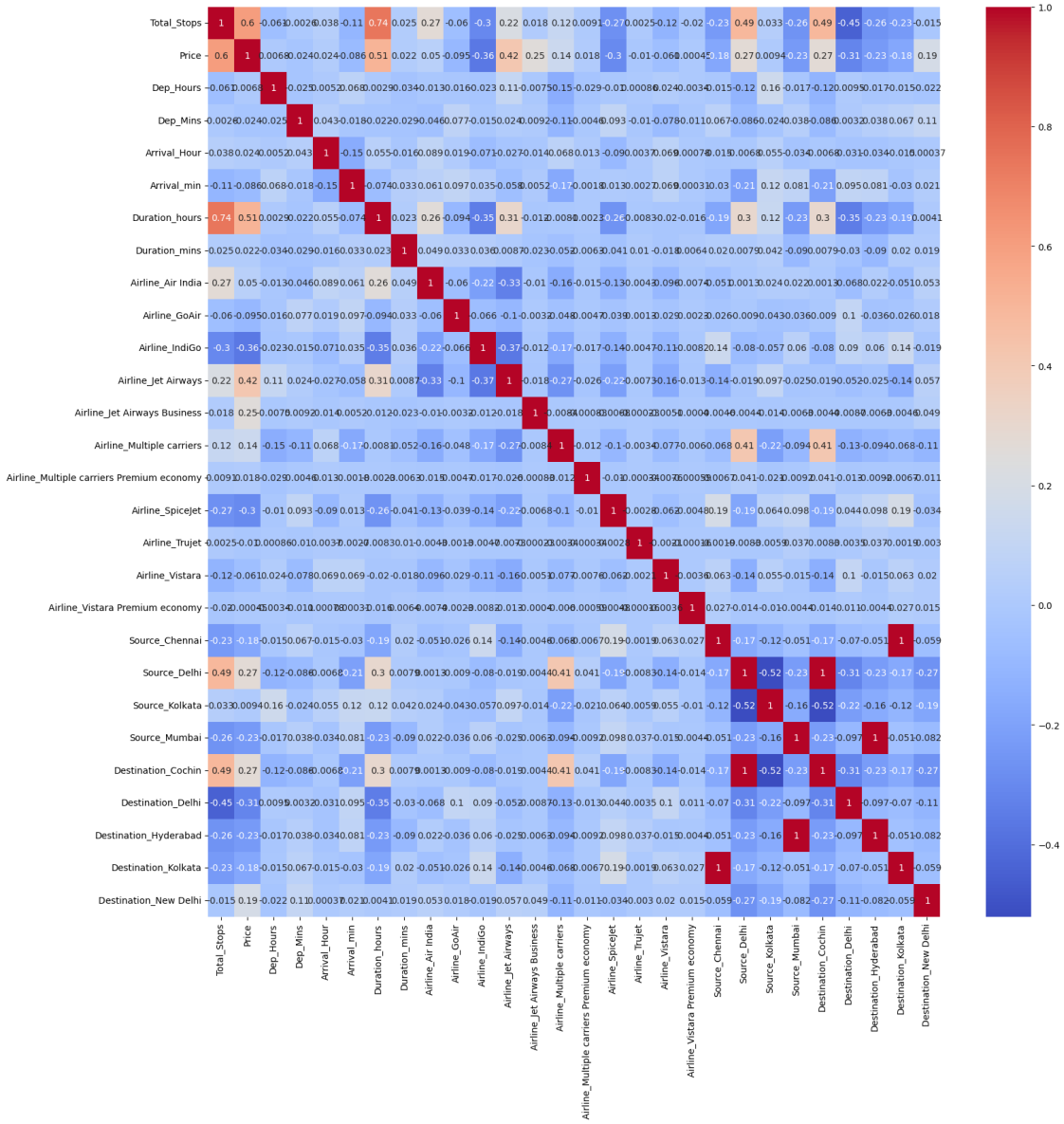
[33]:
```
df_final.drop(['Airline','Source','Destination'],axis=1,inplace=True)
df_final.head()
```

[33]:
```
   Total_Stops  Price Dep_Hours Dep_Mins  Arrival_Hour  Arrival_min  \
0            0   3897        22       20             1           10  \
```

```
1            2   7662        05       50            13           15
2            2  13882        09       25             4           25
3            1   6218        18       05            23           30
4            1  13302        16       50            21           35

   Duration_hours  Duration_mins  Airline_Air India  Airline_GoAir  …
0               2              0              False          False  …  \
1               7              5               True          False  …
2              19              0              False          False  …
3               5              5              False          False  …
4               4              5              False          False  …

   Airline_Vistara Premium economy  Source_Chennai  Source_Delhi
0                            False           False         False  \
1                            False           False         False
2                            False           False          True
3                            False           False         False
4                            False           False         False

   Source_Kolkata  Source_Mumbai  Destination_Cochin  Destination_Delhi
0           False          False               False              False  \
1            True          False               False              False
2           False          False                True              False
3            True          False               False              False
4           False          False               False              False

   Destination_Hyderabad  Destination_Kolkata  Destination_New Delhi
0                  False                False                   True
1                  False                False                  False
2                  False                False                  False
3                  False                False                  False
4                  False                False                   True

[5 rows x 28 columns]
```

[34]:
```
X=df_final.drop('Price',axis=1)
y=df_final['Price']
```

[35]:
```
plt.figure(figsize=(18,18))
sns.heatmap(df_final.corr(),annot=True,cmap='coolwarm')
plt.show()
```



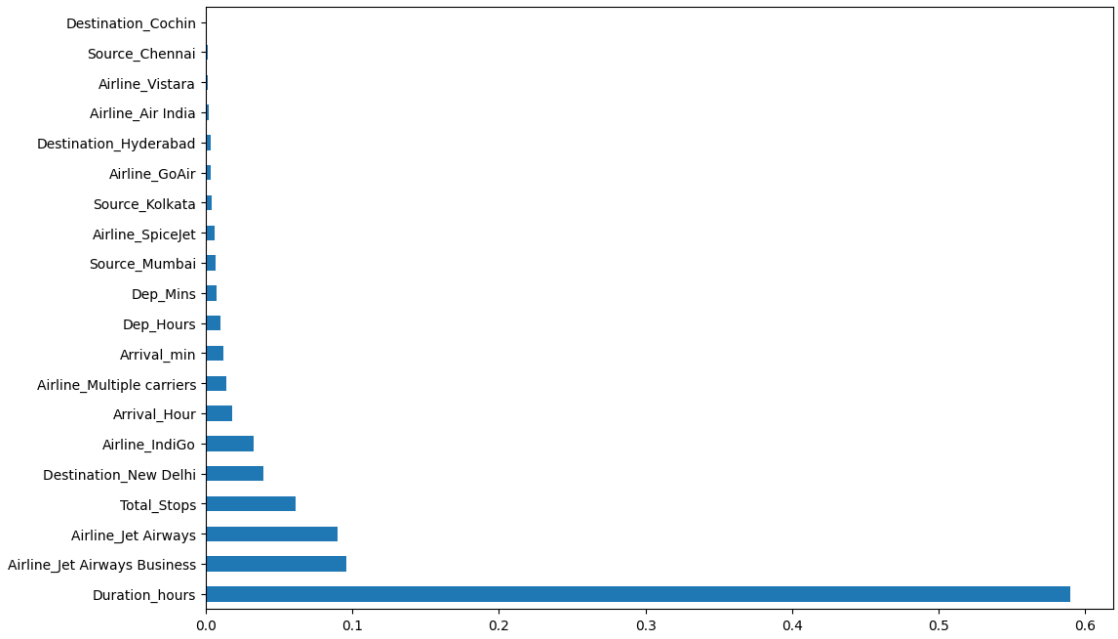
1.0.11 Feature selecting using Gradient Boosting Regressor

[36]:
```
from sklearn.ensemble import GradientBoostingRegressor

model = GradientBoostingRegressor()
model.fit(X,y)
print(model.feature_importances_)
```

```
[6.14089582e-02 9.89991478e-03 7.66735617e-03 1.81453976e-02
 1.19638461e-02 5.89615543e-01 2.30576853e-04 2.13798390e-03
 3.50713998e-03 3.23705034e-02 8.99598195e-02 9.59546936e-02
```

```
1.41918208e-02 2.29511993e-04 6.19076628e-03 0.00000000e+00
1.54167694e-03 6.07738927e-05 1.23067726e-03 1.59639401e-04
4.10299414e-03 6.63564054e-03 3.00584310e-04 8.11220373e-05
3.38033997e-03 0.00000000e+00 3.90327189e-02]
```

[37]:
```
plt.figure(figsize=(12,8))
model=pd.Series(model.feature_importances_,index=X.columns)
model.nlargest(20).plot(kind='barh')
plt.show()
```



1.0.12 Model using Random Forest

[38]:
```
from sklearn.model_selection import train_test_split
```

[39]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
↪2,random_state=42)
```

[40]:
```
from sklearn.ensemble import RandomForestRegressor
model=RandomForestRegressor()
model.fit(X_train,y_train)
```

[40]:
```
RandomForestRegressor()
```

[41]:
```
y_pred=model.predict(X_test)
```

[42]:
```
model.score(X_train,y_train)
```

[42]:
```
0.7885520553007651
```

[43]:
```
model.score(X_test,y_test)
```

[43]:
```
0.5720384275052324
```

[44]:
```
n_estimators=[int(x) for x in np.linspace(start=100, stop=1200, num=12)]
max_features=['auto','sqrt']
max_depth=[int(x) for x in np.linspace(5,30,num=6)]
min_samples_split=[2,5,10,15,100]
min_samples_leaf=[1,2,5,10]
```

[45]:
```
random_grid={
    'n_estimators':n_estimators,
    'max_features':max_features,
    'max_depth':max_depth,
    'min_samples_split':min_samples_split,
    'min_samples_leaf':min_samples_leaf
}
```

[46]:
```
from sklearn.model_selection import RandomizedSearchCV
```

[47]:
```
model2=RandomizedSearchCV(estimator=model, param_distributions=random_grid,␣
↪scoring='neg_mean_squared_error',n_iter=10, cv=4,random_state=42,n_jobs=1)
```

[48]:
```
model2.fit(X_train,y_train)
```

[48]:
```
RandomizedSearchCV(cv=4, estimator=RandomForestRegressor(), n_jobs=1,
                   param_distributions={'max_depth': [5, 10, 15, 20, 25, 30],
                                        'max_features': ['auto', 'sqrt'],
                                        'min_samples_leaf': [1, 2, 5, 10],
                                        'min_samples_split': [2, 5, 10, 15,
                                                              100],
                                        'n_estimators': [100, 200, 300, 400,
                                                         500, 600, 700, 800,
                                                         900, 1000, 1100,
                                                         1200]},
                   random_state=42, scoring='neg_mean_squared_error')
```

[49]:
```
model2.best_params_
```

[49]:
```
{'n_estimators': 300,
 'min_samples_split': 15,
 'min_samples_leaf': 1,
 'max_features': 'sqrt',
 'max_depth': 15}
```

[50]:
```
prediction=model.predict(X_test)
```

[51]:
```
from sklearn import metrics
print(metrics.r2_score(y_test,prediction))
```

```
0.5720384275052324
```

[ ]:
```

```