```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import f1_score
from sklearn.metrics import classification_report,confusion_matrix
import warnings
import pickle
from scipy import stats
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')
```

```
data=pd.read_csv('/content/Data_Train.csv')
```

```
data.head()
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR ? DEL	22:20	01:10 22 Mar
1	Air India	1/05/2019	Kolkata	Banglore	CCU ? IXR ? BBI ?	05:50	13:15

```
for i in data:
  print(i,data[i].unique())
```

```
Airline ['IndiGo' 'Air India' 'Jet Airways' 'SpiceJet' 'Multiple carriers' 'GoAir'
 'Vistara' 'Air Asia' 'Vistara Premium economy' 'Jet Airways Business'
 'Multiple carriers Premium economy' 'Trujet']
Date_of_Journey ['24/03/2019' '1/05/2019' '9/06/2019' '12/05/2019' '01/03/2019'
 '24/06/2019' '12/03/2019' '27/05/2019' '1/06/2019' '18/04/2019'
 '9/05/2019' '24/04/2019' '3/03/2019' '15/04/2019' '12/06/2019'
 '6/03/2019' '21/03/2019' '3/04/2019' '6/05/2019' '15/05/2019'
 '18/06/2019' '15/06/2019' '6/04/2019' '18/05/2019' '27/06/2019'
 '21/05/2019' '06/03/2019' '3/06/2019' '15/03/2019' '3/05/2019'
 '9/03/2019' '6/06/2019' '24/05/2019' '09/03/2019' '1/04/2019'
 '21/04/2019' '21/06/2019' '27/03/2019' '18/03/2019' '12/04/2019'
 '9/04/2019' '1/03/2019' '03/03/2019' '27/04/2019']
Source ['Banglore' 'Kolkata' 'Delhi' 'Chennai' 'Mumbai']
Destination ['New Delhi' 'Banglore' 'Cochin' 'Kolkata' 'Delhi' 'Hyderabad']
Route ['BLR ? DEL' 'CCU ? IXR ? BBI ? BLR' 'DEL ? LKO ? BOM ? COK'
 'CCU ? NAG ? BLR' 'BLR ? NAG ? DEL' 'CCU ? BLR' 'BLR ? BOM ? DEL'
 'DEL ? BOM ? COK' 'DEL ? BLR ? COK' 'MAA ? CCU' 'CCU ? BOM ? BLR'
 'DEL ? AMD ? BOM ? COK' 'DEL ? PNQ ? COK' 'DEL ? CCU ? BOM ? COK'
 'BLR ? COK ? DEL' 'DEL ? IDR ? BOM ? COK' 'DEL ? LKO ? COK'
 'CCU ? GAU ? DEL ? BLR' 'DEL ? NAG ? BOM ? COK' 'CCU ? MAA ? BLR'
 'DEL ? HYD ? COK' 'CCU ? HYD ? BLR' 'DEL ? COK' 'CCU ? DEL ? BLR'
 'BLR ? BOM ? AMD ? DEL' 'BOM ? DEL ? HYD' 'DEL ? MAA ? COK' 'BOM ? HYD'
 'DEL ? BHO ? BOM ? COK' 'DEL ? JAI ? BOM ? COK' 'DEL ? ATQ ? BOM ? COK'
 'DEL ? JDH ? BOM ? COK' 'CCU ? BBI ? BOM ? BLR' 'BLR ? MAA ? DEL'
 'DEL ? GOI ? BOM ? COK' 'DEL ? BDQ ? BOM ? COK' 'CCU ? JAI ? BOM ? BLR'
 'CCU ? BBI ? BLR' 'BLR ? HYD ? DEL' 'DEL ? TRV ? COK'
 'CCU ? IXR ? DEL ? BLR' 'DEL ? IXU ? BOM ? COK' 'CCU ? IXB ? BLR'
 'BLR ? BOM ? JDH ? DEL' 'DEL ? UDR ? BOM ? COK' 'DEL ? HYD ? MAA ? COK'
 'CCU ? BOM ? COK ? BLR' 'BLR ? CCU ? DEL' 'CCU ? BOM ? GOI ? BLR'
 'DEL ? RPR ? NAG ? BOM ? COK' 'DEL ? HYD ? BOM ? COK'
 'CCU ? DEL ? AMD ? BLR' 'CCU ? PNQ ? BLR' 'BLR ? CCU ? GAU ? DEL'
 'CCU ? DEL ? COK ? BLR' 'BLR ? PNQ ? DEL' 'BOM ? JDH ? DEL ? HYD'
 'BLR ? BOM ? BHO ? DEL' 'DEL ? AMD ? COK' 'BLR ? LKO ? DEL'
 'CCU ? GAU ? BLR' 'BOM ? GOI ? HYD' 'CCU ? BOM ? AMD ? BLR'
 'CCU ? BBI ? IXR ? DEL ? BLR' 'DEL ? DED ? BOM ? COK'
 'DEL ? MAA ? BOM ? COK' 'BLR ? AMD ? DEL' 'BLR ? VGA ? DEL'
 'CCU ? JAI ? DEL ? BLR' 'CCU ? AMD ? BLR' 'CCU ? VNS ? DEL ? BLR'
 'BLR ? BOM ? IDR ? DEL' 'BLR ? BBI ? DEL' 'BLR ? GOI ? DEL'
 'BOM ? AMD ? ISK ? HYD' 'BOM ? DED ? DEL ? HYD' 'DEL ? IXC ? BOM ? COK'
 'CCU ? PAT ? BLR' 'BLR ? CCU ? BBI ? DEL' 'CCU ? BBI ? HYD ? BLR'
 'BLR ? BOM ? NAG ? DEL' 'BLR ? CCU ? BBI ? HYD ? DEL' 'BLR ? GAU ? DEL'
 'BOM ? BHO ? DEL ? HYD' 'BOM ? JLR ? HYD' 'BLR ? HYD ? VGA ? DEL'
 'CCU ? KNU ? BLR' 'CCU ? BOM ? PNQ ? BLR' 'DEL ? BBI ? COK'
```

```
       'BLR ? VGA ? HYD ? DEL' 'BOM ? JDH ? JAI ? DEL ? HYD'
       'DEL ? GWL ? IDR ? BOM ? COK' 'CCU ? RPR ? HYD ? BLR' 'CCU ? VTZ ? BLR'
       'CCU ? DEL ? VGA ? BLR' 'BLR ? BOM ? IDR ? GWL ? DEL'
       'CCU ? DEL ? COK ? TRV ? BLR' 'BOM ? COK ? MAA ? HYD' 'BOM ? NDC ? HYD'
       'BLR ? BDQ ? DEL' 'CCU ? BOM ? TRV ? BLR' 'CCU ? BOM ? HBX ? BLR'
       'BOM ? BDQ ? DEL ? HYD' 'BOM ? CCU ? HYD' 'BLR ? TRV ? COK ? DEL'
       'BLR ? IDR ? DEL' 'CCU ? IXZ ? MAA ? BLR' 'CCU ? GAU ? IMF ? DEL ? BLR'
       'BOM ? GOI ? PNQ ? HYD' 'BOM ? BLR ? CCU ? BBI ? HYD' 'BOM ? MAA ? HYD'
       'BLR ? BOM ? UDR ? DEL' 'BOM ? UDR ? DEL ? HYD' 'BLR ? VGA ? VTZ ? DEL'
       'BLR ? HBX ? BOM ? BHO ? DEL' 'CCU ? IXA ? BLR' 'BOM ? RPR ? VTZ ? HYD'
       'BLR ? HBX ? BOM ? AMD ? DEL' 'BOM ? IDR ? DEL ? HYD' 'BOM ? BLR ? HYD'
       'BLR ? STV ? DEL' 'CCU ? IXB ? DEL ? BLR' 'BOM ? JAI ? DEL ? HYD'
       'BOM ? VNS ? DEL ? HYD' 'BLR ? HBX ? BOM ? NAG ? DEL' nan
       'BLR ? BOM ? IXC ? DEL' 'BLR ? CCU ? BBI ? HYD ? VGA ? DEL'
       'BOM ? BBI ? HYD']
```

```
data.Date_of_Journey=data.Date_of_Journey.str.split('/')
```

```
data.Date_of_Journey
```

```
0        [24, 03, 2019]
1         [1, 05, 2019]
2         [9, 06, 2019]
3        [12, 05, 2019]
4        [01, 03, 2019]
               ...
10678     [9, 04, 2019]
10679    [27, 04, 2019]
10680    [27, 04, 2019]
10681    [01, 03, 2019]
10682     [9, 05, 2019]
Name: Date_of_Journey, Length: 10683, dtype: object
```

```
data['Date']=data.Date_of_Journey.str[0]
data['Month']=data.Date_of_Journey.str[1]
data['Year']=data.Date_of_Journey.str[2]
```

```
data.Total_Stops.unique()
```

```
array(['non-stop', '2 stops', '1 stop', '3 stops', nan, '4 stops'],
      dtype=object)
```

```
data.Route=data.Route.str.split(' ')
data.Route
```

```
0                       [BLR, ?, DEL]
1        [CCU, ?, IXR, ?, BBI, ?, BLR]
2        [DEL, ?, LKO, ?, BOM, ?, COK]
3                [CCU, ?, NAG, ?, BLR]
4                [BLR, ?, NAG, ?, DEL]
                     ...
10678                   [CCU, ?, BLR]
10679                   [CCU, ?, BLR]
10680                   [BLR, ?, DEL]
10681                   [BLR, ?, DEL]
10682    [DEL, ?, GOI, ?, BOM, ?, COK]
Name: Route, Length: 10683, dtype: object
```

```
data['City1']=data.Route.str[0]
data['City2']=data.Route.str[1]
data['City3']=data.Route.str[2]
data['City4']=data.Route.str[3]
data['City5']=data.Route.str[4]
data['City6']=data.Route.str[5]
```

```
data.Dep_Time=data.Dep_Time.str.split(':')
```

```
data['Dep_Time_hour']=data.Dep_Time.str[0]
data['Dep_Time_Mins']=data.Dep_Time.str[1]
```

```
data.Arrival_Time=data.Arrival_Time.str.split(' ')
```

```
data['Arrival_Date']=data.Arrival_Time.str[1]
data['Time_of_Arrival']=data.Arrival_Time.str[0]


data['Time_of_Arrival']=data.Time_of_Arrival.str.split(':')


data['Arrival_Time_Hours']=data.Time_of_Arrival.str[0]
data['Arrival_Time_Mins']=data.Time_of_Arrival.str[1]


data.Duration=data.Duration.str.split(' ')



data['Travel_Hours']=data.Duration.str[0]
data['Travel_Hours']=data['Travel_Hours'].str.split('h')
data['Travel_Hours']=data['Travel_Hours'].str[0]
data.Travel_Hours=data.Travel_Hours
data['Travel_Mins']=data.Duration.str[1]
data.Travel_Mins=data.Travel_Mins.str.split('m')
data.Travel_Mins=data.Travel_Mins.str[0]



data.Total_Stops.replace('non_stops',0,inplace=True)
data.Total_Stops=data.Total_Stops.str.split(' ')
data.Total_Stops=data.Total_Stops.str[0]



data.Total_Stops.replace('non_stop',0,inplace=True)
data.Total_Stops=data.Total_Stops.str.split(' ')
data.Total_Stops=data.Total_Stops.str[0]




data.Additional_Info.unique()
```

```
array(['No info', 'In-flight meal not included',
       'No check-in baggage included', '1 Short layover', 'No Info',
       '1 Long layover', 'Change airports', 'Business class',
       'Red-eye flight', '2 Long layover'], dtype=object)
```

```
data.Additional_Info.replace('No Info','No info',inplace=True)


data.isnull().sum()
```

```
Airline                 0
Date_of_Journey         0
Source                  0
Destination             0
Route                   1
Dep_Time                0
Arrival_Time            0
Duration                0
Total_Stops             1
Additional_Info         0
Price                   0
Date                    0
Month                   0
Year                    0
City1                   1
City2                   1
City3                   1
City4                3492
City5                3492
City6                9117
Dep_Time_hour           0
Dep_Time_Mins           0
Arrival_Date         6348
Time_of_Arrival         0
Arrival_Time_Hours      0
Arrival_Time_Mins       0
Travel_Hours            0
Travel_Mins          1032
dtype: int64
```

```
data.drop(['City4','City5','City6'],axis=1,inplace=True)



data.drop(['Date_of_Journey','Route','Dep_Time','Arrival_Time','Duration'],axis=1,inplace=True)
data.drop(['Time_of_Arrival'],axis=1,inplace=True)


data.isnull().sum()
```

```
Airline                 0
Source                  0
Destination             0
Total_Stops             1
Additional_Info         0
Price                   0
Date                    0
Month                   0
Year                    0
City1                   1
City2                   1
City3                   1
Dep_Time_hour           0
Dep_Time_Mins           0
Arrival_Date         6348
Arrival_Time_Hours      0
Arrival_Time_Mins       0
Travel_Hours            0
Travel_Mins          1032
dtype: int64
```

```
data['City3'].fillna('None',inplace=True)


data['Arrival_Date'].fillna(data['Date'],inplace=True)



data['Travel_Mins'].fillna(0,inplace=True)


data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 20 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Airline             10683 non-null  object
 1   Source              10683 non-null  object
 2   Destination         10683 non-null  object
 3   Total_Stops         10682 non-null  object
 4   Additional_Info     10683 non-null  object
 5   Price               10683 non-null  int64 
 6   Date                10683 non-null  object
 7   Month               10683 non-null  object
 8   Year                10683 non-null  object
 9   City1               10682 non-null  object
 10  City2               10682 non-null  object
 11  City3               10683 non-null  object
 12  Dep_Time_hour       10683 non-null  int64 
 13  Dep_Time_Mins       10683 non-null  int64 
 14  Arrival_Date        10683 non-null  int64 
 15  Arrival_Time_Hours  10683 non-null  int64 
 16  Arrival_Time_Mins   10683 non-null  int64 
 17  Travel_Hours        10683 non-null  object
 18  Travel_Mins         10683 non-null  int64 
 19  date                10683 non-null  int64 
dtypes: int64(8), object(12)
memory usage: 1.6+ MB
```

```
data['date']=data.Date.astype('int64')
data['Month'].astype(str).astype(int, errors='ignore')
data['Year'].astype(str).astype(int, errors='ignore')

data['Dep_Time_hour']=data.Dep_Time_hour.astype('int64')
data['Dep_Time_hour']=data.Dep_Time_hour.astype('int64')
data['Dep_Time_Mins']=data.Dep_Time_Mins.astype('int64')
data['Arrival_Date']=data.Arrival_Date.astype('int64')
data['Arrival_Time_Hours']=data.Arrival_Time_Hours.astype('int64')
```

```
data['Arrival_Time_Mins']=data.Arrival_Time_Mins.astype('int64')
data.Travel_Mins=data.Travel_Mins.astype('int64')
```

```
data[data['Travel_Hours']=='5m']
```

	Airline	Source	Destination	Total_Stops	Additional_Info	Price	Date	Month	Year	City1	City2	City3	Dep_Time_hour	Dep_Time_
6474	Air India	Mumbai	Hyderabad	2	No info	17327	6	03	2019	BOM	?	GOI	16	

```
categorical=['Airline','Source','Destination','Additional_Info','City1','month','year']
numerical=['Total_Stops','Date','Dep_Time_Hour','Dep_Time_Mins','Arrival_Date','Arrival_Time_Hours','Arrival_Time_Mins','Travel_Hours','Trave
```

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
data.Airli=le.fit_transform(data.Airline)
data.Source=le.fit_transform(data.Source)
data.Destination=le.fit_transform(data.Destination)
data.Total_Stops=le.fit_transform(data.Total_Stops)
data.City1  =le.fit_transform(data.City1  )
data.City2=le.fit_transform(data.City2)
data.City3=le.fit_transform(data.City3)
data.Additional_Info=le.fit_transform(data.Additional_Info)
data.head()
```

	Airline	Source	Destination	Total_Stops	Additional_Info	Price	Date	Month	Year	City1	City2	City3	Dep_Time_hour	Dep_Time_Min
0	IndiGo	0	5	4	7	3897	24	03	2019	0	0	10	22	2
1	Air India	3	0	1	7	7662	1	05	2019	2	0	20	5	5
2	Jet Airways	2	1	1	7	13882	9	06	2019	3	0	27	9	2
3	IndiGo	3	0	0	7	6218	12	05	2019	2	0	29	18	
4	IndiGo	0	5	0	7	13302	01	03	2019	0	0	29	16	5

```
data.head()
```

	Airline	Source	Destination	Total_Stops	Additional_Info	Price	Date	Month	Year	City1	City2	City3	Dep_Time_hour	Dep_Time_Min
0	IndiGo	0	5	4	7	3897	24	03	2019	0	0	10	22	2
1	Air India	3	0	1	7	7662	1	05	2019	2	0	20	5	5
2	Jet Airways	2	1	1	7	13882	9	06	2019	3	0	27	9	2
3	IndiGo	3	0	0	7	6218	12	05	2019	2	0	29	18	
4	IndiGo	0	5	0	7	13302	01	03	2019	0	0	29	16	5

```
data.head()
```

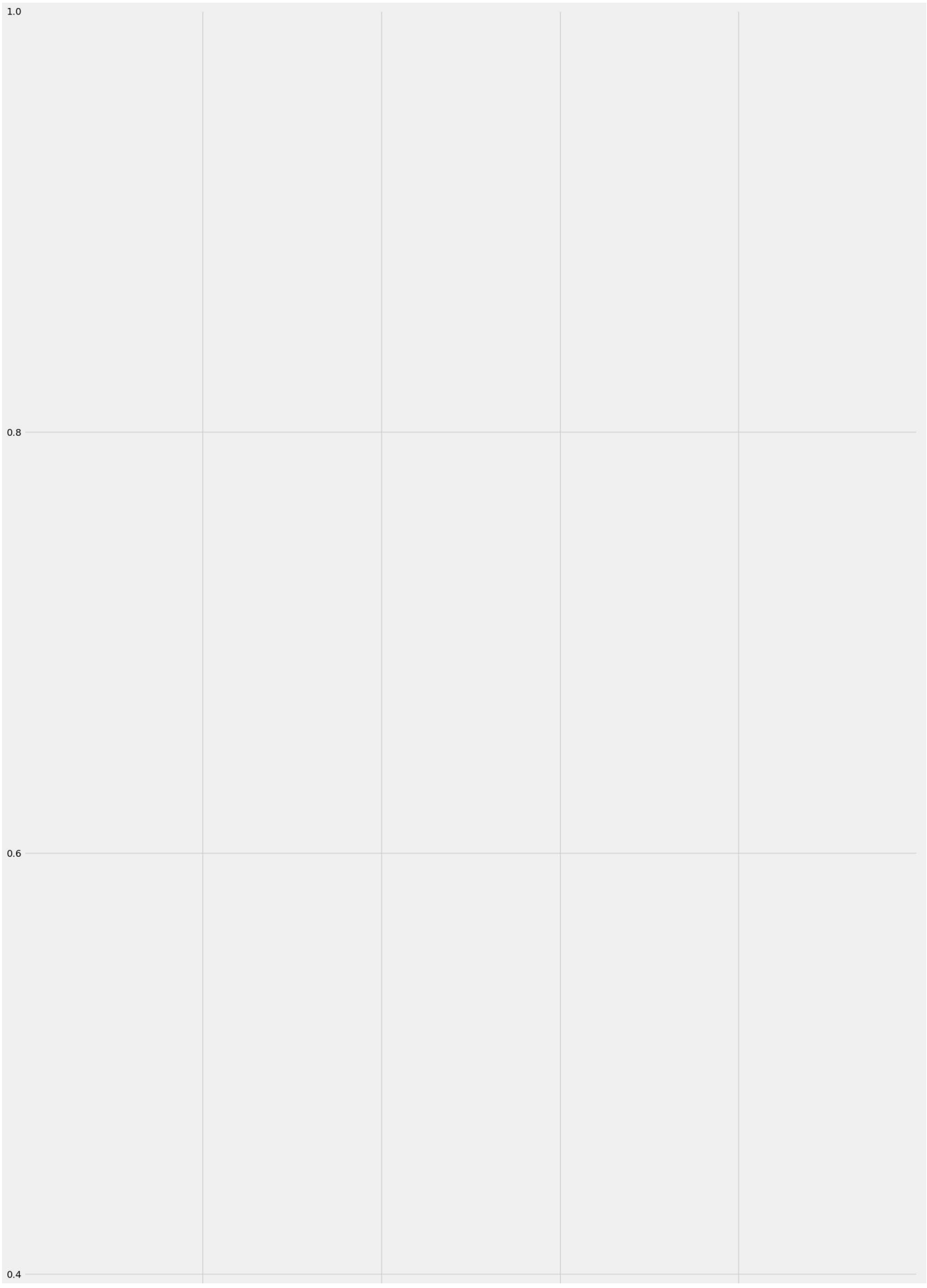
	Airline	Source	Destination	Total_Stops	Additional_Info	Price	Date	Month	Year	City1	City2	City3	Dep_Time_hour	Dep_Time_Min
0	IndiGo	0	5	4	7	3897	24	03	2019	0	0	10	22	2
1	Air India	3	0	1	7	7662	1	05	2019	2	0	20	5	5
2	Jet Airways	2	1	1	7	13882	9	06	2019	3	0	27	9	2
3	IndiGo	3	0	0	7	6218	12	05	2019	2	0	29	18	
4	IndiGo	0	5	0	7	13302	01	03	2019	0	0	29	16	5

```
data.describe()
```

	Source	Destination	Total_Stops	Additional_Info	Price	City1	City2	City3	Dep_Time_hour	De
count	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	1
mean	1.952261	1.436113	1.458579	6.582140	9087.064121	2.019657	0.000094	9.683890	12.490686	
std	1.177221	1.474782	1.806560	0.838073	4611.359167	1.206527	0.009675	6.567734	5.748650	
min	0.000000	0.000000	0.000000	0.000000	1759.000000	0.000000	0.000000	0.000000	0.000000	
25%	2.000000	0.000000	0.000000	7.000000	5277.000000	1.000000	0.000000	6.000000	8.000000	
50%	2.000000	1.000000	0.000000	7.000000	8372.000000	2.000000	0.000000	7.000000	11.000000	
75%	3.000000	2.000000	4.000000	7.000000	12373.000000	3.000000	0.000000	10.000000	18.000000	
max	4.000000	5.000000	5.000000	8.000000	79512.000000	5.000000	1.000000	40.000000	23.000000	

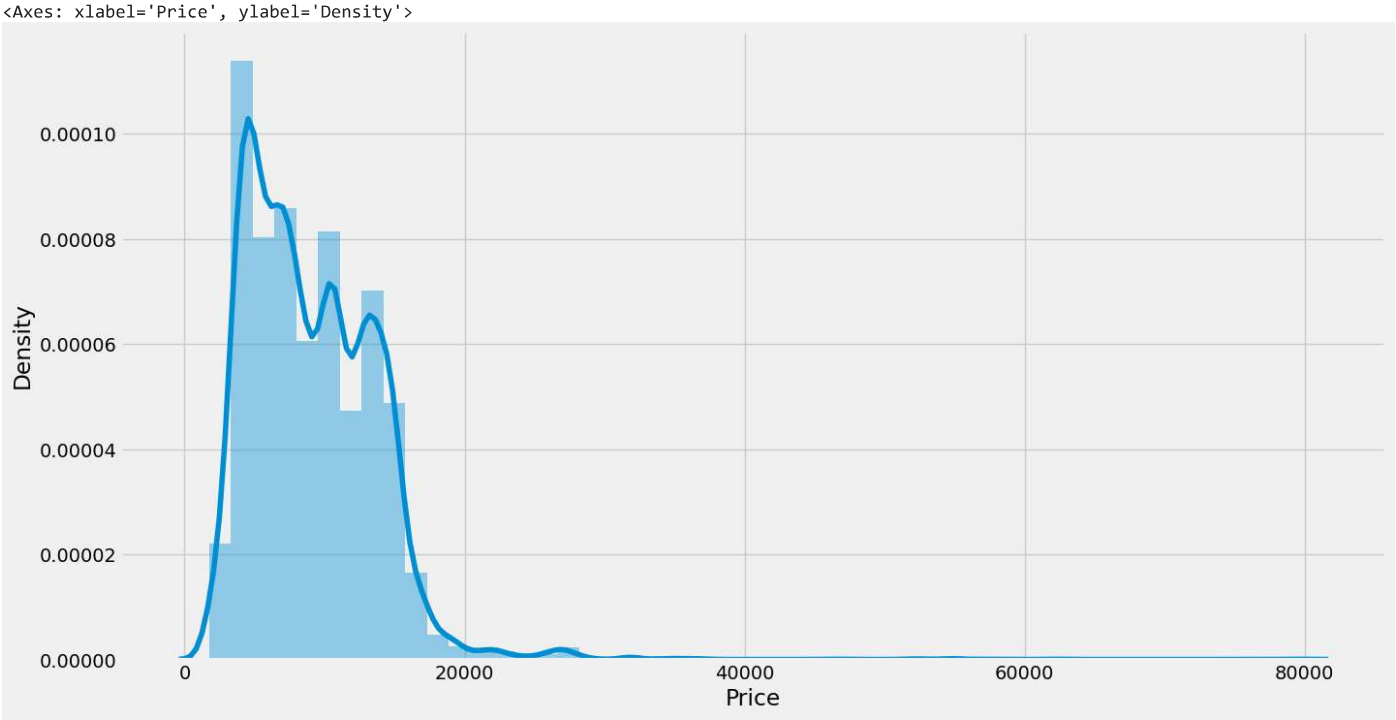
```
import seaborn as sns
c=1
plt.figure(figsize=(20,45))

for i in data:

  #sns.countplot(data(i))
  plt.xticks(rotation=90)
  plt.tight_layout(pad=3.0)
  c=c+1
  print(type (i))
  plt.show()
```

```
<class 'str'>
```



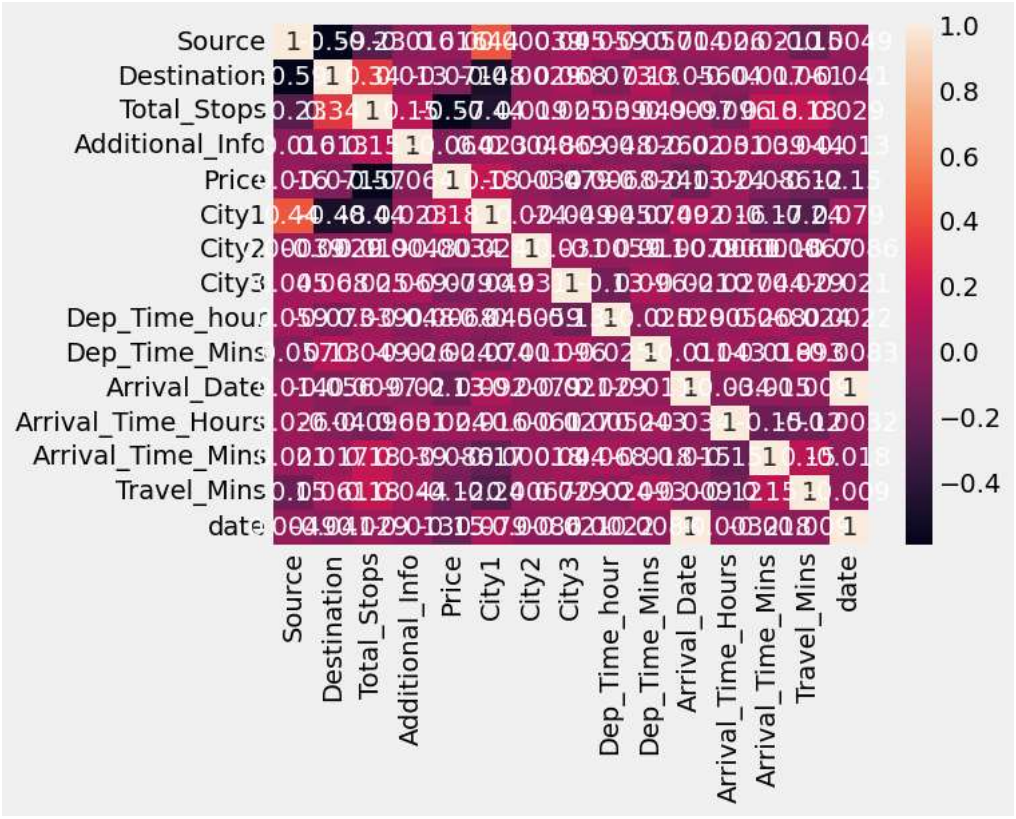
```
plt.figure(figsize=(15,8))
sns.distplot(data.Price)
```

```
<Axes: xlabel='Price', ylabel='Density'>
```



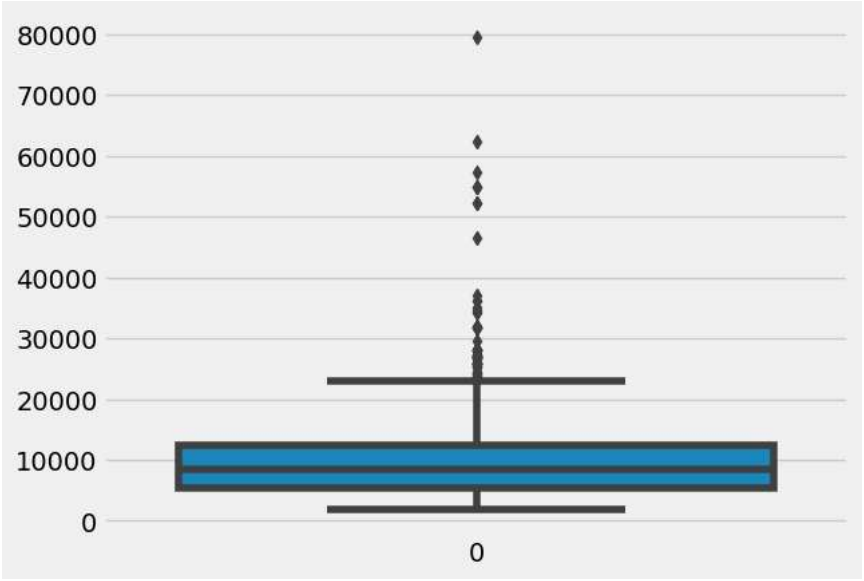

```
sns.heatmap(data.corr(),annot=True)
```

```
<Axes: >
```



```
import seaborn as sns
sns.boxplot(data['Price'])
```

```
<Axes: >
```




```
y=data['Price']
x=data.drop(columns=['Price'],axis=1)
```

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
```

```
x_scaled = ss.fit_transform(x)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-69-ca9912d0bdd8> in <cell line: 1>()
----> 1 x_scaled = ss.fit_transform(x)

 7 frames 

/usr/local/lib/python3.9/dist-packages/pandas/core/generic.py in __array__(self, dtype)
   2062
   2063     def __array__(self, dtype: npt.DTypeLike | None = None) -> np.ndarray:
-> 2064         return np.asarray(self._values, dtype=dtype)
   2065
   2066     def __array_wrap__(

ValueError: could not convert string to float: 'IndiGo'

SEARCH STACK OVERFLOW
```

```
x_scaled = pd.DataFrame(x_scaled,columns=x.columns)
x_scaled.head()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-70-d1fc7ecb22a9> in <cell line: 1>()
----> 1 x_scaled = pd.DataFrame(x_scaled,columns=x.columns)
      2 x_scaled.head()

NameError: name 'x_scaled' is not defined

SEARCH STACK OVERFLOW
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
x_train.head
```

```
<bound method NDFrame.head of                Airline  Source  Destination  Total_Stops  Additional_Info  \
8990        Jet Airways       4            3            1               7
3684        Jet Airways       2            1            0               5
```

```
1034           SpiceJet       2            1            0               7
3909  Multiple carriers       2            1            0               7
3088          Air India       2            1            1               7
...                 ...     ...          ...          ...             ...
5734        Jet Airways       2            1            0               7
5191        Jet Airways       3            0            0               5
5390  Multiple carriers       2            1            0               7
860              IndiGo       0            5            4               7
7270        Jet Airways       2            1            0               5

     Date Month  Year  City1  City2  City3  Dep_Time_hour  Dep_Time_Mins  \
8990   12    03  2019      1      0     39              6             30
3684    9    05  2019      3      0      6             11             30
1034   24    04  2019      3      0     28             15             45
3909   21    03  2019      3      0      6             12             50
3088   24    06  2019      3      0     15             17             15
...   ...   ...   ...    ...    ...    ...            ...            ...
5734   27    03  2019      3      0      6              9              0
5191    9    05  2019      2      0      6             14              5
5390   15    05  2019      3      0      6             12             50
860    03    03  2019      0      0     10              0             40
7270    1    06  2019      3      0      6             13              0

      Arrival_Date  Arrival_Time_Hours  Arrival_Time_Mins Travel_Hours  \
8990            12                  16                 35           10
3684            10                  12                 35           25
1034            24                  22                  5            6
3909            22                   1                 35           12
3088            25                  19                 15           26
...            ...                 ...                ...          ...
5734            28                   4                 25           19
5191             9                  20                 45            6
5390            16                   1                 30           12
860              3                   3                 25            2
7270             2                   4                 25           15

      Travel_Mins  date
8990            5    12
3684            5     9
1034           20    24
3909           45    21
3088            0    24
...           ...   ...
5734           25    27
5191           40     9
5390           40    15
860            45     3
7270           25     1

[8546 rows x 19 columns]>
```

```
from sklearn.ensemble import RandomForestRegressor,GradientBoostingRegressor,AdaBoostRegressor
rfr=RandomForestRegressor()
gb=GradientBoostingRegressor()
ad=AdaBoostRegressor()
```

0.6

```
from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error

for i in [rfr,gb,ad]:
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  test_score=r2_score(y_test,y_pred)
  train_score=r2_score(y_train,i.predict(x_train.csv))
  if abs(train_score-test_score)<=0.2:
    print(i)

    print("R2 score is",r2_score(y_test,y_pred))
    print("R2 for train data",r2_score(y_train,i.predict(x_train)))
    print("Mean Absolute Error is",mean_absolute_error(y_pred,y_test))
    print("Mean Squared Error is",mean_squared_error(y_pred,y_test))
    print("Root Mean Sqaured Error is",(mean_squared_error(y_pred,y_test,squared=False)))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-67-0771d970f43d> in <cell line: 3>()
      2
      3 for i in [rfr,gb,ad]:
```

```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor

from sklearn.metrics import r2_score,mean_absolute_error,mean_squard_error

knn=KNeighborsRegressor()
svr=SVR()
dt=DecisionTreeRegressor()

for i in [knn,svr,dt]:
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  test_score=r2_score(y_test,y_pred)
  train_score=r2_score(y_train,i.predict(x_train))
  if abs(train_score_test_score)<=0.1:
    print(i)
    print('R2 Score is ',r2_score(y_test,y_pred))
    print('R2 Score for train data',r2_score(y_train,i.predict(x_train)))
    print('Mean Absolute Error is',mean_absolute_error(y_test,y_pred))
    print('Mean Squared Error is',mean_squared_error(y_test,y_pred))
    print('Root Mean Squared Error is',(mean_squared_error(y_test,y_pred,squared=False)))
```

```
---------------------------------------------------------------------------
ImportError                               Traceback (most recent call last)
<ipython-input-59-8313b8364350> in <cell line: 5>()
      3 from sklearn.tree import DecisionTreeRegressor
      4
----> 5 from sklearn.metrics import r2_score,mean_absolute_error,mean_squard_error
      6
      7 knn=KNeighborsRegressor()

ImportError: cannot import name 'mean_squard_error' from 'sklearn.metrics' (/usr/local/lib/python3.9/dist-
packages/sklearn/metrics/__init__.py)

---------------------------------------------------------------------------
NOTE: If your import is failing due to a missing package, you can
manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the
"Open Examples" button below.
---------------------------------------------------------------------------
```

SEARCH STACK OVERFLOW

```
from sklearn.model_selection import cross_val_score
for i in range(2,5):
    cv=cross_val_score(rfr,x,y,cv=i)
    print(rfr,cv.mean())
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-4-d41dda11af5c> in <cell line: 2>()
      1 from sklearn.model_selection import cross_val_score
      2 for i in range(2,5):
----> 3     cv=cross_val_score(rfr,x,y,cv=i)
      4     print(rfr,cv.mean())

NameError: name 'rfr' is not defined
```

SEARCH STACK OVERFLOW

```
from sklearn.model_selection import RandomizedSearchCV
```

```
param_grid={'n_estimators':[10,30,50,70,100],'max_depth':[None,1,2,3],'max_features':['auto','sqrt']}
rfr=RandomForestRegressor()
rf_res=RandomizedSearchCV(estimator=rfr,param_distributions=param_grid,cv=3,verbose=2,n_jobs=-1)
rf_res.fit(x_train,y_train)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-58-0623a328d564> in <cell line: 4>()
      2 rfr=RandomForestRegressor()
      3 rf_res=RandomizedSearchCV(estimator=rfr,param_distributions=param_grid,cv=3,verbose=2,n_jobs=-1)
----> 4 rf_res.fit(x_train,y_train)

NameError: name 'x_train' is not defined
```

SEARCH STACK OVERFLOW

```
gb=GradientBoostingRegressor()
gb_res=RandomizedSearchCV(estimator=gb,param_distributions=param_grid,cv=3,verbose=2,n_jobs=-1)
gb_res.fit(x_train,y_train)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-8-2771cb4aefe0> in <cell line: 1>()
----> 1 gb=GradientBoostingRegressor()
      2 gb_res=RandomizedSearchCV(estimator=gb,param_distributions=param_grid,cv=3,verbose=2,n_jobs=-1)
      3 gb_res.fit(x_train,y_train)

NameError: name 'GradientBoostingRegressor' is not defined
```

SEARCH STACK OVERFLOW

```
rfr=RandomForestRegressor(n_estimators=10,max_features='sqrt',max_depth=None)
rfr.fit(x_train,y_train)
y_train_pred=rfr.predict(x_train)
y_test_pred=rfr.predict(x_test)
print("train accuracy",r2_score(y_train_pred,y_train))
print("test accuracy",r2_score(y_test_pred,y_test))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-9-b7c361fe3953> in <cell line: 1>()
----> 1 rfr=RandomForestRegressor(n_estimators=10,max_features='sqrt',max_depth=None)
      2 rfr.fit(x_train,y_train)
      3 y_train_pred=rfr.predict(x_train)
      4 y_test_pred=rfr.predict(x_test)
      5 print("train accuracy",r2_score(y_train_pred,y_train))

NameError: name 'RandomForestRegressor' is not defined
```

SEARCH STACK OVERFLOW

```
knn=KNeighborsRegressor(n_neighbors=2,algorithm='auto',metric_parmas=None,n_jobs=-1)
knn.fit(x_train,y_train)
y_train_pred=knn.predict(x_train)
y_test_pred=knn.predict(x_test)
print("train accuracy",r2_score(y_train_pred,y_train))
print("test accuracy",r2_score(y_test_pred,y_test))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-10-88375d41a60f> in <cell line: 1>()
----> 1 knn=KNeighborsRegressor(n_neighbors=2,algorithm='auto',metric_parmas=None,n_jobs=-1)
      2 knn.fit(x_train,y_train)
      3 y_train_pred=knn.predict(x_train)
      4 y_test_pred=knn.predict(x_test)
      5 print("train accuracy",r2_score(y_train_pred,y_train))

TypeError: __init__() got an unexpected keyword argument 'metric_parmas'
```

SEARCH STACK OVERFLOW

```
rfr=RandomForestRegressor(n_estimators=10,max_features='sqrt',max_depth=None)
rfr.fit(x_train,y_train)
y_train_pred=rfr.predict(x_train)
y_test_pred=rfr.predict(x_test)
print("train accuraacy",r2_score(y_train_pred,y_train))
print("train accuracy",r2_score(y_test_pred,y_test))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-11-b685a4716a2b> in <cell line: 1>()
----> 1 rfr=RandomForestRegressor(n_estimators=10,max_features='sqrt',max_depth=None)
      2 rfr.fit(x_train,y_train)
      3 y_train_pred=rfr.predict(x_train)
      4 y_test_pred=rfr.predict(x_test)
      5 print("train accuraacy",r2_score(y_train_pred,y_train))

NameError: name 'RandomForestRegressor' is not defined
```

SEARCH STACK OVERFLOW

```
price_list=pd.DataFrame({'price':prices})
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-23-fe8faf4ab565> in <cell line: 1>()
----> 1 price_list=pd.DataFrame({'price':prices})

NameError: name 'pd' is not defined
```

SEARCH STACK OVERFLOW

```
price_list
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-22-8c800825bae0> in <cell line: 1>()
----> 1 price_list

NameError: name 'price_list' is not defined
```

SEARCH STACK OVERFLOW

```
import pickle
pickle.dump(rfr,open('model1.pk1','wb'))
```

Double-click (or enter) to edit

```
import pickle
pickle.dump(rfr,open('model1.pk1','wb'))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-12-134067c7e297> in <cell line: 2>()
      1 import pickle
----> 2 pickle.dump(rfr,open('model1.pk1','wb'))

NameError: name 'rfr' is not defined
```

SEARCH STACK OVERFLOW

```
from flask import Flask,render_template,request
import numpy as np
import pickle
```

```
model=pickle.load(open(r"model1.pkl",'rb'))
```

```
@app.route("/home")
def home():
  return render_template('home.html')


@app.route("/predict")
def home1():
  return render_template('predict.html')
  @app.route("pred",methods=['POST','GET'])
  def predict():
    x=[[int(x)for x in request.form.values()]]
    print(x)
    x=np.array(x)
    print(x.shape)
```

```
        print(x)
        pred=model.predict(x)
        print(pred)
        return render_template('submit.html',prediction_text=pred)


if__name__=="__main__":
  app.run(debug=False)
```