Name : Annapoornima

Roll no: 225229101

Project: Hotel Reservation

Import Dataset

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
df=pd.read_csv("Hotel Reservations.csv")
df.head()
```

Out[2]:

	Booking_ID	no_of_adults	no_of_children	no_of_weekend_nights	no_of_week_nights	type_of_
0	INN00001	2	0	1	2	I
1	INN00002	2	0	2	3	N
2	INN00003	1	0	2	1	I
3	INN00004	2	0	0	2	I
4	INN00005	2	0	1	1	N

****Cleaning the data make it numberic value****

In [3]:
```
data = df.replace({"type_of_meal_plan":{"Not Selected":0,"Meal Plan 1":1,"Meal
                   "room_type_reserved":{"Room_Type 1":1,"Room_Type 2":2,
                   "market_segment_type":{"Offline":0,"Online":1, "Corpor
                   "booking_status":{"Canceled":0,"Not_Canceled":1}})
```

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36275 entries, 0 to 36274
Data columns (total 19 columns):
 #   Column                                Non-Null Count  Dtype  
---  ------                                --------------  -----  
 0   Booking_ID                            36275 non-null  object 
 1   no_of_adults                          36275 non-null  int64  
 2   no_of_children                        36275 non-null  int64  
 3   no_of_weekend_nights                  36275 non-null  int64  
 4   no_of_week_nights                     36275 non-null  int64  
 5   type_of_meal_plan                     36275 non-null  int64  
 6   required_car_parking_space            36275 non-null  int64  
 7   room_type_reserved                    36275 non-null  int64  
 8   lead_time                             36275 non-null  int64  
 9   arrival_year                          36275 non-null  int64  
 10  arrival_month                         36275 non-null  int64  
 11  arrival_date                          36275 non-null  int64  
 12  market_segment_type                   36275 non-null  int64  
 13  repeated_guest                        36275 non-null  int64  
 14  no_of_previous_cancellations          36275 non-null  int64  
 15  no_of_previous_bookings_not_canceled  36275 non-null  int64  
 16  avg_price_per_room                    36275 non-null  float64
 17  no_of_special_requests                36275 non-null  int64  
 18  booking_status                        36275 non-null  int64  
dtypes: float64(1), int64(17), object(1)
memory usage: 5.3+ MB
```

In [5]:
```
data.describe()
```

Out[5]:

	no_of_adults	no_of_children	no_of_weekend_nights	no_of_week_nights	type_of_meal_pla
count	36275.000000	36275.000000	36275.000000	36275.000000	36275.0000(
mean	1.844962	0.105279	0.810724	2.204300	0.9499(
std	0.518715	0.402648	0.870644	1.410905	0.4801(
min	0.000000	0.000000	0.000000	0.000000	0.0000(
25%	2.000000	0.000000	0.000000	1.000000	1.0000(
50%	2.000000	0.000000	1.000000	2.000000	1.0000(
75%	2.000000	0.000000	2.000000	3.000000	1.0000(
max	4.000000	10.000000	7.000000	17.000000	3.0000(

In [6]:
```
data.drop(["booking_status"],axis=1).corrwith(data["booking_status"])
```

Out[6]:
```
no_of_adults                           -0.086920
no_of_children                         -0.033078
no_of_weekend_nights                   -0.061563
no_of_week_nights                      -0.092996
type_of_meal_plan                      -0.049374
required_car_parking_space              0.086185
room_type_reserved                     -0.022986
lead_time                              -0.438538
arrival_year                           -0.179529
arrival_month                           0.011233
arrival_date                           -0.010629
market_segment_type                     0.077877
repeated_guest                          0.107287
no_of_previous_cancellations            0.033728
no_of_previous_bookings_not_canceled    0.060179
avg_price_per_room                     -0.142569
no_of_special_requests                  0.253070
dtype: float64
```

Build the data training and Test Set

In [7]:
```
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
X = data.drop(["Booking_ID","arrival_month","arrival_date", "booking_status"],
y = data["booking_status"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, rand
```

Make model pipeline

In [8]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB

model_pipeline = []
model_pipeline.append(LogisticRegression(solver='liblinear'))
model_pipeline.append(SVC())
model_pipeline.append(KNeighborsClassifier())
model_pipeline.append(DecisionTreeClassifier())
model_pipeline.append(RandomForestClassifier())
model_pipeline.append(GaussianNB())
```

In [9]:
```
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix

model_list = ["Logistic Regression", "SVM", "KNN", "Decision Tree", "Random Fo
acc_list =[]
auc_list = []
cm_list = []

for model in model_pipeline:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    acc_list.append(metrics.accuracy_score(y_test,y_pred))
    fpr, tpr ,_tresholds = metrics.roc_curve(y_test,y_pred)
    auc_list.append(round(metrics.auc(fpr,tpr),2))
    cm_list.append(confusion_matrix(y_test,y_pred))
```

Make heatmap result from test data set

Finding the Best to find the best model

In [10]:
```
result_df = pd.DataFrame({"Model":model_list,"Accuracy" :acc_list, "AUC":auc_l
result_df
```

Out[10]:

	Model	Accuracy	AUC
0	Logistic Regression	0.789324	0.73
1	SVM	0.762509	0.67
2	KNN	0.805530	0.76
3	Decision Tree	0.858157	0.84
4	Random Forest	0.886726	0.86
5	Naive Bayes	0.451508	0.58

the result say that Random Forest are the best model for the data set to make predicion

Make a prediction

In [11]:
```
from sklearn.ensemble import RandomForestClassifier

clf=RandomForestClassifier(n_estimators=100)

clf.fit(X_train,y_train)

y_pred=clf.predict(X_test)
```

In [12]:
```
#checking the accuracy pf the model
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.8859744382257121
```

****Preparing the data, here I use data from the updateed version of the original data****

In [13]:
```
guest = 36273 # can change by the guest Booking ID
data1=data.iloc[[guest]].drop(["Booking_ID","arrival_month","arrival_date", "b
#limiting the input data cause random forest just eccept 15 feature. so I elim
data1.values.tolist()
```

Out[13]:
```
[[2.0,
  0.0,
  0.0,
  3.0,
  0.0,
  0.0,
  1.0,
  63.0,
  2018.0,
  1.0,
  0.0,
  0.0,
  0.0,
  94.5,
  0.0]]
```

In [15]:
```
code = clf.predict(data1)
if code == 0:
    print("Not Cancel")
else:
    print("cancel")
```

```
Not Cancel
```

In [ ]: