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

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

In [2]:
```python
data=pd.read_csv('zomato.csv',encoding='latin-1')
```

In [3]:
```python
data.head()
```

Out[3]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...	Botswana Pula(P)
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...	Botswana Pula(P)
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...	Botswana Pula(P)
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...	Botswana Pula(P)
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...	Botswana Pula(P)

5 rows × 21 columns

In [4]:
```python
data.tail()
```

Out[4]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency
9546	5915730	NamlÛ± Gurme	208	ÛÁstanbul	Kemanke□ô Karamustafa Pa□ôa Mahallesi, RÛ±htÛ±...	Karakí_y	Karakí_y, ÛÁstanbul	28.977392	41.022793	Turkish	...	Turkish Lira(TL)
9547	5908749	Ceviz AÛôacÛ±	208	ÛÁstanbul	Ko□ôuyolu Mahallesi, Muhittin íistí_ndaÛô Cadd...	Ko□ôuyolu	Ko□ôuyolu, ÛÁstanbul	29.041297	41.009847	World Cuisine, Patisserie, Cafe	...	Turkish Lira(TL)
9548	5915807	Huqqa	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.034640	41.055817	Italian, World Cuisine	...	Turkish Lira(TL)
9549	5916112	A□ô□ôk Kahve	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.036019	41.057979	Restaurant Cafe	...	Turkish Lira(TL)
9550	5927402	Walter's Coffee Roastery	208	ÛÁstanbul	CafeaÛôa Mahallesi, BademaltÛ± Sokak, No 21/B,...	Moda	Moda, ÛÁstanbul	29.026016	40.984776	Cafe	...	Turkish Lira(TL)

5 rows × 21 columns

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

Out[5]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Votes
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370	156.909748
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	430.169145
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.000000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	5.000000
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	31.000000
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	131.000000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.000000

In [6]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 21 columns):
Restaurant ID           9551 non-null int64
Restaurant Name         9551 non-null object
Country Code            9551 non-null int64
City                    9551 non-null object
Address                 9551 non-null object
Locality                9551 non-null object
Locality Verbose        9551 non-null object
Longitude               9551 non-null float64
Latitude                9551 non-null float64
Cuisines                9542 non-null object
Average Cost for two    9551 non-null int64
Currency                9551 non-null object
Has Table booking       9551 non-null object
Has Online delivery     9551 non-null object
Is delivering now       9551 non-null object
Switch to order menu    9551 non-null object
Price range             9551 non-null int64
Aggregate rating        9551 non-null float64
Rating color            9551 non-null object
Rating text             9551 non-null object
Votes                   9551 non-null int64
dtypes: float64(3), int64(5), object(13)
memory usage: 1.1+ MB
```

In [8]:
```python
data.drop(columns=['Longitude','Latitude'],inplace=True)
```

In [9]:
```python
data.nunique()
```

Out[9]:
```
Restaurant ID           9551
Restaurant Name         7446
Country Code              15
City                     141
Address                 8918
Locality                1208
Locality Verbose        1265
Cuisines                1825
Average Cost for two     140
Currency                  12
Has Table booking          2
Has Online delivery        2
Is delivering now          2
Switch to order menu       1
Price range                4
Aggregate rating          33
Rating color               6
Rating text                6
Votes                   1012
dtype: int64
```

In [10]:
```python
data.drop(columns=['Restaurant ID','Country Code','Address','Locality','Locality Verbose'],inplace=True)
```

In [11]:
```python
from sklearn.preprocessing import LabelEncoder
```

In [12]:
```python
w={'Orange':0,'White':1,'Yellow':2,'Green':3,'Dark Green':4,'Red':5}
data['Rating color']=data['Rating color'].map(w)
```

In [13]:
```python
d = {'Poor':1,'Good':3,'Very Good':4,'Excellent':5,'Average':2,'Not rated':0}
data['Rating text']=data['Rating text'].map(d)
```

In [14]:
```python
data['Has Table booking']=data['Has Table booking'].map({'No':0,'Yes':1})
```

In [15]:
```python
data['Has Online delivery']=data['Has Online delivery'].map({'No':0,'Yes':1})
```

In [16]:
```python
data['Is delivering now']=data['Is delivering now'].map({'No':0,'Yes':1})
```

In [17]:
```python
data['Switch to order menu']=data['Switch to order menu'].map({'No':0,'Yes':1})
```

In [18]:
```python
data.head()
```

Out[18]:

	Restaurant Name	City	Cuisines	Average Cost for two	Currency	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	Aggregate rating	Rating color	Rating text	Votes
0	Le Petit Souffle	Makati City	French, Japanese, Desserts	1100	Botswana Pula(P)	1	0	0	0	3	4.8	4	5	314
1	Izakaya Kikufuji	Makati City	Japanese	1200	Botswana Pula(P)	1	0	0	0	3	4.5	4	5	591
2	Heat - Edsa Shangri-La	Mandaluyong City	Seafood, Asian, Filipino, Indian	4000	Botswana Pula(P)	1	0	0	0	4	4.4	3	4	270
3	Ooma	Mandaluyong City	Japanese, Sushi	1500	Botswana Pula(P)	0	0	0	0	4	4.9	4	5	365
4	Sambo Kojin	Mandaluyong City	Japanese, Korean	1500	Botswana Pula(P)	1	0	0	0	4	4.8	4	5	229

In [19]:
```python
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import VarianceThreshold
from sklearn.decomposition import PCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
```

In [21]:
```python
x = data.drop(columns=['Restaurant Name','City','Cuisines','Currency'])
y = data['Rating text']
```

In [22]:
```python
x.shape,y.shape
```

Out[22]: ((9551, 10), (9551,))

In [23]:
```python
x_train,x_test,y_train,y_test = train_test_split(x,y,train_size=0.75,random_state=42,stratify=y)
```

In [24]:
```python
x_train.shape,x_test.shape
```

Out[24]: ((7163, 10), (2388, 10))

In [25]:
```python
x_tresh = VarianceThreshold(threshold=0.01)
x_train_unique = x_tresh.fit_transform(x_train)
x_test_unique =x_tresh.transform(x_test)
```

In [26]:
```python
x_train_unique.shape,x_test_unique.shape
```

Out[26]: ((7163, 8), (2388, 8))

In [28]:
```python
x_train_unique = pd.DataFrame(x_train_unique)
x_test_unique = pd.DataFrame(x_test_unique)
```

In [29]:
```python
def correlation(data,thresh):
    corrmat = data.corr()
    corr_col=set()
    for i in range(len(corrmat.columns)):
        for j in range(i):
            if abs(corrmat.iloc[i,j]>thresh):
                col = corrmat.columns[i]
                corr_col.add(col)
    return corr_col
```

In [30]:
```python
corr_feature = correlation(x_train_unique,0.80)
```

In [31]:
```python
x_train_uncorr = x_train_unique.drop(columns=corr_feature)
x_test_uncorr =x_test_unique.drop(columns=corr_feature)
```

In [32]:
```python
x_train_uncorr.shape,x_test_uncorr.shape
```

Out[32]: ((7163, 7), (2388, 7))

In [34]:
```python
from sklearn.metrics import accuracy_score
```

In [35]:
```python
def RandomForest(x_train,x_test,y_train,y_test):
    clf = RandomForestClassifier(random_state=42,n_estimators=100)
    clf.fit(x_train,y_train)
    y_pred = clf.predict(x_test)
    print('Accuarcy :',accuracy_score(y_test,y_pred))
```

In [36]:
```python
RandomForest(x_train_uncorr,x_test_uncorr,y_train,y_test)
```

```
Accuarcy : 1.0
```

LDA

In [38]:
```python
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
```

In [42]:
```python
lda = LDA(n_components=4)
```

In [43]:
```python
x_train_lda = lda.fit_transform(x_train_uncorr,y_train)
```

In [44]:
```python
x_train_lda.shape
```

Out[44]: (7163, 4)

In [45]:
```python
x_train_uncorr.shape
```

Out[45]: (7163, 7)

In [46]:
```python
x_test_lda = lda.transform(x_test_uncorr)
```

In [47]:
```python
x_test_lda.shape
```

Out[47]: (2388, 4)

In [48]:
```python
x_test_uncorr.shape
```

Out[48]: (2388, 7)

In [49]:
```python
x_train_lda.shape,x_test_lda.shape
```

Out[49]: ((7163, 4), (2388, 4))

In [50]:
```python
RandomForest(x_train_lda,x_test_lda,y_train,y_test)
```

```
Accuarcy : 1.0
```

PCA

In [51]:
```python
from sklearn.decomposition import PCA
```

In [52]:
```python
pca = PCA(n_components=3)
```

In [53]:
```python
x_train_pca= pca.fit_transform(x_train_uncorr,y_train)
```

In [55]:
```python
x_train_pca.shape
```

Out[55]: (7163, 3)

In [56]:
```python
x_test_pca = pca.transform(x_test_uncorr)
```

In [57]:
```python
x_test_pca.shape
```

Out[57]: (2388, 3)

In [58]:
```python
RandomForest(x_train_pca,x_test_pca,y_train,y_test)
```

```
Accuarcy : 0.9551926298157454
```

In [60]:
```python
for i in range(1,8):
    pca = PCA(n_components=i,random_state=42)
    x_train_pca = pca.fit_transform(x_train_uncorr)
    x_test_pca = pca.transform(x_test_uncorr)
    print('Accuarcy at component',i)
    RandomForest(x_train_pca,x_test_pca,y_train,y_test)
    print()
```

```
Accuarcy at component 1
Accuarcy : 0.6695979899497487

Accuarcy at component 2
Accuarcy : 0.7018425460636516

Accuarcy at component 3
Accuarcy : 0.9551926298157454

Accuarcy at component 4
Accuarcy : 0.9962311557788944

Accuarcy at component 5
Accuarcy : 0.9970686767169179

Accuarcy at component 6
Accuarcy : 0.9962311557788944

Accuarcy at component 7
Accuarcy : 0.9962311557788944
```

In [ ]: