In [124]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objs as go
import plotly.offline as py

import matplotlib.ticker as mtick
plt.style.use('fivethirtyeight')
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.tree import ExtraTreeRegressor
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [125]:
```
data=pd.read_csv('zomato.csv')
```

In [126]:
```
data.head()
```

Out[126]:

	url	address	name	online_order	book_table
0	https://www.zomato.com/bangalore/jalsa-banasha...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes
1	https://www.zomato.com/bangalore/spice-elephan...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No
3	https://www.zomato.com/bangalore/addhuri-udupi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No
4	https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No

In [127]:
```
data.shape
```

Out[127]:
```
(51717, 17)
```

In [128]:
```
data.columns
```

Out[128]:
```
Index(['url', 'address', 'name', 'online_order', 'book_table', 'rate', 'vote
s',
       'phone', 'location', 'rest_type', 'dish_liked', 'cuisines',
       'approx_cost(for two people)', 'reviews_list', 'menu_item',
       'listed_in(type)', 'listed_in(city)'],
      dtype='object')
```

In [129]:
```
data.dtypes
```

Out[129]:
```
url                            object
address                        object
name                           object
online_order                   object
book_table                     object
rate                           object
votes                           int64
phone                          object
location                       object
rest_type                      object
dish_liked                     object
cuisines                       object
approx_cost(for two people)    object
reviews_list                   object
menu_item                      object
listed_in(type)                object
listed_in(city)                object
dtype: object
```

In [130]:
```
data.drop(['url','phone'],axis=1,inplace=True)
```

In [131]:
```
data.duplicated().sum()
```

Out[131]:
```
43
```

In [132]:
```
data.drop_duplicates(inplace=True)
```

In [133]:
```
data.duplicated().sum()
```

Out[133]:
```
0
```

In [134]:
```
data.dropna(how='any',inplace=True)
```

In [135]:
```
data.isnull().sum()
```

Out[135]:
```
address                        0
name                           0
online_order                   0
book_table                     0
rate                           0
votes                          0
location                       0
rest_type                      0
dish_liked                     0
cuisines                       0
approx_cost(for two people)    0
reviews_list                   0
menu_item                      0
listed_in(type)                0
listed_in(city)                0
dtype: int64
```

In [136]:
```
data.columns
```

Out[136]:
```
Index(['address', 'name', 'online_order', 'book_table', 'rate', 'votes',
       'location', 'rest_type', 'dish_liked', 'cuisines',
       'approx_cost(for two people)', 'reviews_list', 'menu_item',
       'listed_in(type)', 'listed_in(city)'],
      dtype='object')
```

In [137]:
```
data.rename(columns={'approx_cost(for two people)':'cost','rate':'rating','lis
```

In [138]:
```
data.columns
```

Out[138]:
```
Index(['address', 'name', 'online_order', 'book_table', 'rating', 'votes',
       'location', 'rest_type', 'dish_liked', 'cuisines', 'cost',
       'reviews_list', 'menu_item', 'type', 'city'],
      dtype='object')
```

In [139]:
```
#as we can see there, two values seperated from each other it means that there
#that are reason for change in its datatype from "Int" to "Object" so we have r
#values are removed
data['cost'].unique()
```

Out[139]:
```
array(['800', '300', '600', '700', '550', '500', '450', '650', '400',
       '750', '200', '850', '1,200', '150', '350', '250', '1,500',
       '1,300', '1,000', '100', '900', '1,100', '1,600', '950', '230',
       '1,700', '1,400', '1,350', '2,200', '2,000', '1,800', '1,900',
       '180', '330', '2,500', '2,100', '3,000', '2,800', '3,400', '40',
       '1,250', '3,500', '4,000', '2,400', '1,450', '3,200', '6,000',
       '1,050', '4,100', '2,300', '120', '2,600', '5,000', '3,700',
       '1,650', '2,700', '4,500'], dtype=object)
```

In [140]:
```
data['cost']=data['cost'].apply(lambda x: x.replace(',',''))
data['cost']=data['cost'].astype(float)
```

In [141]:
```
data['cost'].dtypes
```

Out[141]:
```
dtype('float64')
```

In [142]:
```
data['cost'].unique()
```

Out[142]:
```
array([ 800.,  300.,  600.,  700.,  550.,  500.,  450.,  650.,  400.,
        750.,  200.,  850., 1200.,  150.,  350.,  250., 1500., 1300.,
       1000.,  100.,  900., 1100., 1600.,  950.,  230., 1700., 1400.,
       1350., 2200., 2000., 1800., 1900.,  180.,  330., 2500., 2100.,
       3000., 2800., 3400.,   40., 1250., 3500., 4000., 2400., 1450.,
       3200., 6000., 1050., 4100., 2300.,  120., 2600., 5000., 3700.,
       1650., 2700., 4500.])
```

In [143]:
```
data['rating'].unique()
```

Out[143]:
```
array(['4.1/5', '3.8/5', '3.7/5', '4.6/5', '4.0/5', '4.2/5', '3.9/5',
       '3.0/5', '3.6/5', '2.8/5', '4.4/5', '3.1/5', '4.3/5', '2.6/5',
       '3.3/5', '3.5/5', '3.8 /5', '3.2/5', '4.5/5', '2.5/5', '2.9/5',
       '3.4/5', '2.7/5', '4.7/5', 'NEW', '2.4/5', '2.2/5', '2.3/5',
       '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5', '4.1 /5', '2.9 /5',
       '2.7 /5', '2.5 /5', '2.6 /5', '4.5 /5', '4.3 /5', '3.7 /5',
       '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '3.4 /5', '3.6 /5',
       '3.3 /5', '4.6 /5', '4.9 /5', '3.2 /5', '3.0 /5', '2.8 /5',
       '3.5 /5', '3.1 /5', '4.8 /5', '2.3 /5', '4.7 /5', '2.4 /5',
       '2.1 /5', '2.2 /5', '2.0 /5', '1.8 /5'], dtype=object)
```

In [144]:
```
data=data.loc[data.rating !='NEW']
```

In [145]:
```
data['rating'].unique()
```

Out[145]:
```
array(['4.1/5', '3.8/5', '3.7/5', '4.6/5', '4.0/5', '4.2/5', '3.9/5',
       '3.0/5', '3.6/5', '2.8/5', '4.4/5', '3.1/5', '4.3/5', '2.6/5',
       '3.3/5', '3.5/5', '3.8 /5', '3.2/5', '4.5/5', '2.5/5', '2.9/5',
       '3.4/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5', '4.8/5',
       '3.9 /5', '4.2 /5', '4.0 /5', '4.1 /5', '2.9 /5', '2.7 /5',
       '2.5 /5', '2.6 /5', '4.5 /5', '4.3 /5', '3.7 /5', '4.4 /5',
       '4.9/5', '2.1/5', '2.0/5', '1.8/5', '3.4 /5', '3.6 /5', '3.3 /5',
       '4.6 /5', '4.9 /5', '3.2 /5', '3.0 /5', '2.8 /5', '3.5 /5',
       '3.1 /5', '4.8 /5', '2.3 /5', '4.7 /5', '2.4 /5', '2.1 /5',
       '2.2 /5', '2.0 /5', '1.8 /5'], dtype=object)
```

In [146]:
```
data.isnull().sum()
```

Out[146]:
```
address         0
name            0
online_order    0
book_table      0
rating          0
votes           0
location        0
rest_type       0
dish_liked      0
cuisines        0
cost            0
reviews_list    0
menu_item       0
type            0
city            0
dtype: int64
```

In [147]:
```
data['rating']=data['rating'].apply(lambda x: x.replace('/5',''))
```

In [148]:
```
data['rating'].unique()
```

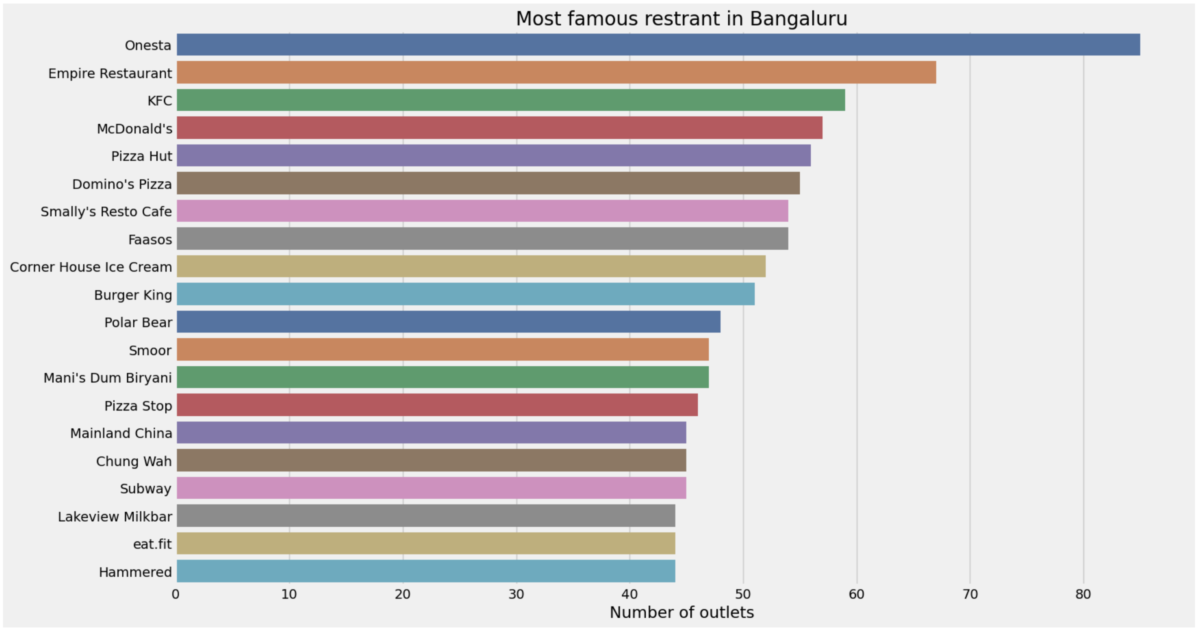
Out[148]:
```
array(['4.1', '3.8', '3.7', '4.6', '4.0', '4.2', '3.9', '3.0', '3.6',
       '2.8', '4.4', '3.1', '4.3', '2.6', '3.3', '3.5', '3.8 ', '3.2',
       '4.5', '2.5', '2.9', '3.4', '2.7', '4.7', '2.4', '2.2', '2.3',
       '4.8', '3.9 ', '4.2 ', '4.0 ', '4.1 ', '2.9 ', '2.7 ', '2.5 ',
       '2.6 ', '4.5 ', '4.3 ', '3.7 ', '4.4 ', '4.9', '2.1', '2.0', '1.8',
       '3.4 ', '3.6 ', '3.3 ', '4.6 ', '4.9 ', '3.2 ', '3.0 ', '2.8 ',
       '3.5 ', '3.1 ', '4.8 ', '2.3 ', '4.7 ', '2.4 ', '2.1 ', '2.2 ',
       '2.0 ', '1.8 '], dtype=object)
```

Visualizations

In [ ]:
```

```

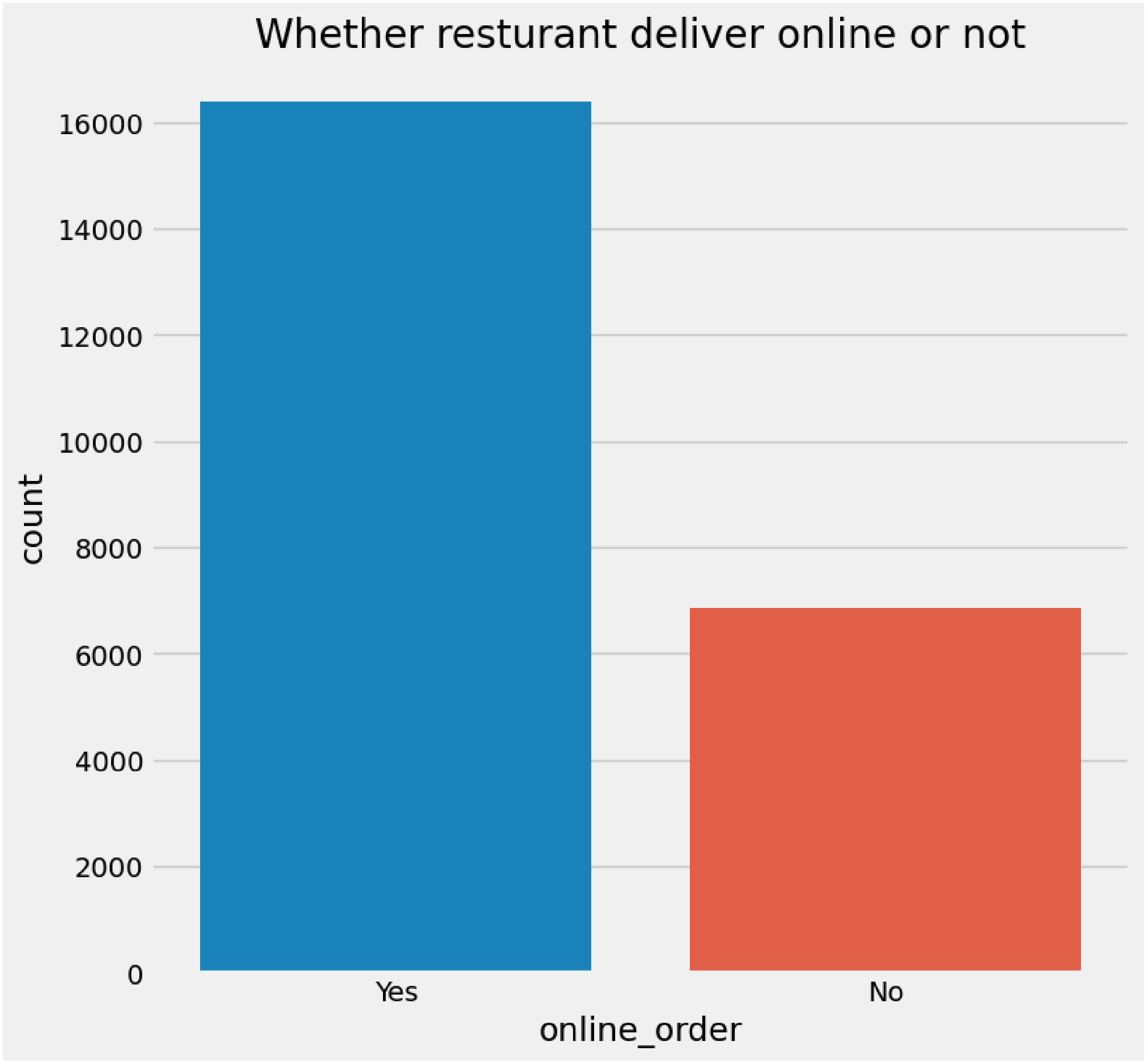
In [149]:
```
#first of all we will check the most famous returant in Bangaluru
```

```
In [150]: plt.figure(figsize=(17,10))
          chains=data['name'].value_counts()[:20]
          sns.barplot(x=chains,y=chains.index,palette='deep')
          plt.title("Most famous restrant in Bangaluru")
          plt.xlabel("Number of outlets")
          plt.show()
```

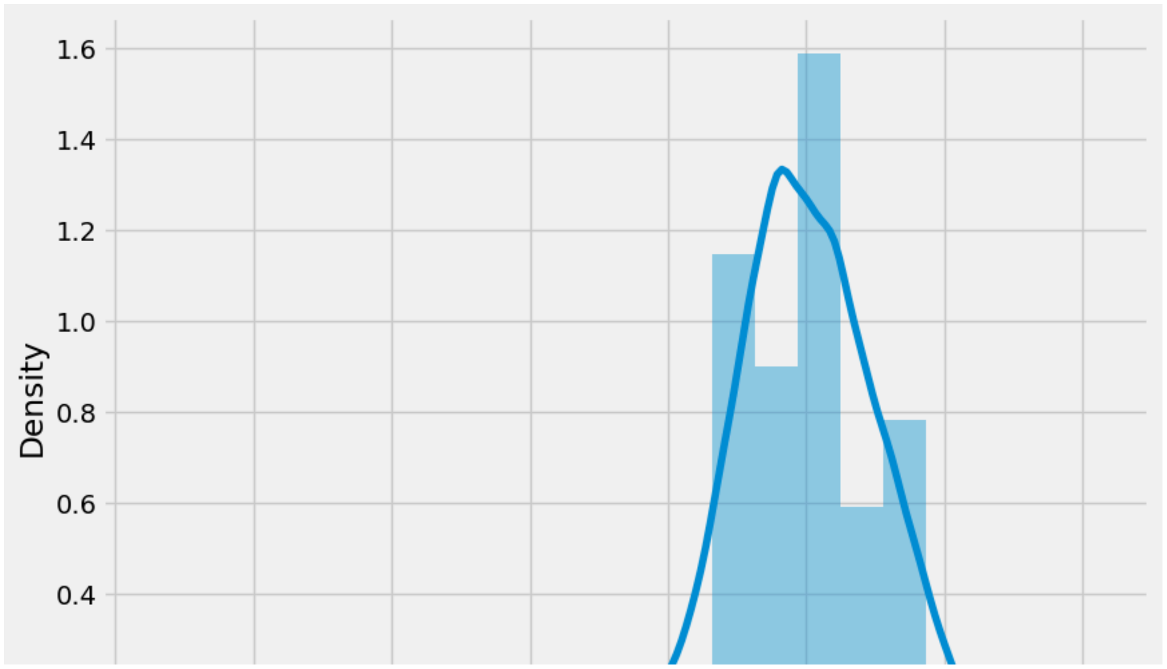


In [151]:
```
import plotly.graph_objs as go
import plotly.offline as py


x=data['book_table'].value_counts()
colors=['#7ffc03','#fcbe03']
trace=go.Pie(labels=x.index,values=x,textinfo='value',
            marker=dict(colors=colors,
                       line=dict(color='#fcbe03',width=2)))
layout=go.Layout(title="Table Booking",width=600,height=600)
fig=go.Figure(data=[trace],layout=layout)
py.iplot(fig,filename='pie_chart_subplot')
```

```
In [152]: sns.countplot(x='online_order',data=data)
          fig=plt.gcf()
          fig.set_size_inches(8,8)
          plt.title("Whether resturant deliver online or not")
          plt.show()
```



In [153]:
```
plt.figure(figsize=(9,7))
sns.distplot(data['rating'],bins=20)
plt.show()
```



In [154]:
```
data['rating'].min()
```

Out[154]:
```
'1.8'
```

In [155]:
```
data['rating'].max()
```

Out[155]:
```
'4.9 '
```

In [156]:
```
data['rating']=data['rating'].astype(float)
```

In [157]:
```
data['rating'].dtypes
```

Out[157]:
```
dtype('float64')
```

In [158]:
```
((data['rating'] >= 1) & (data['rating'] < 2)).sum()
```

Out[158]:
```
5
```

In [159]:
```
((data['rating']>=2) & (data['rating']<3)).sum()
```

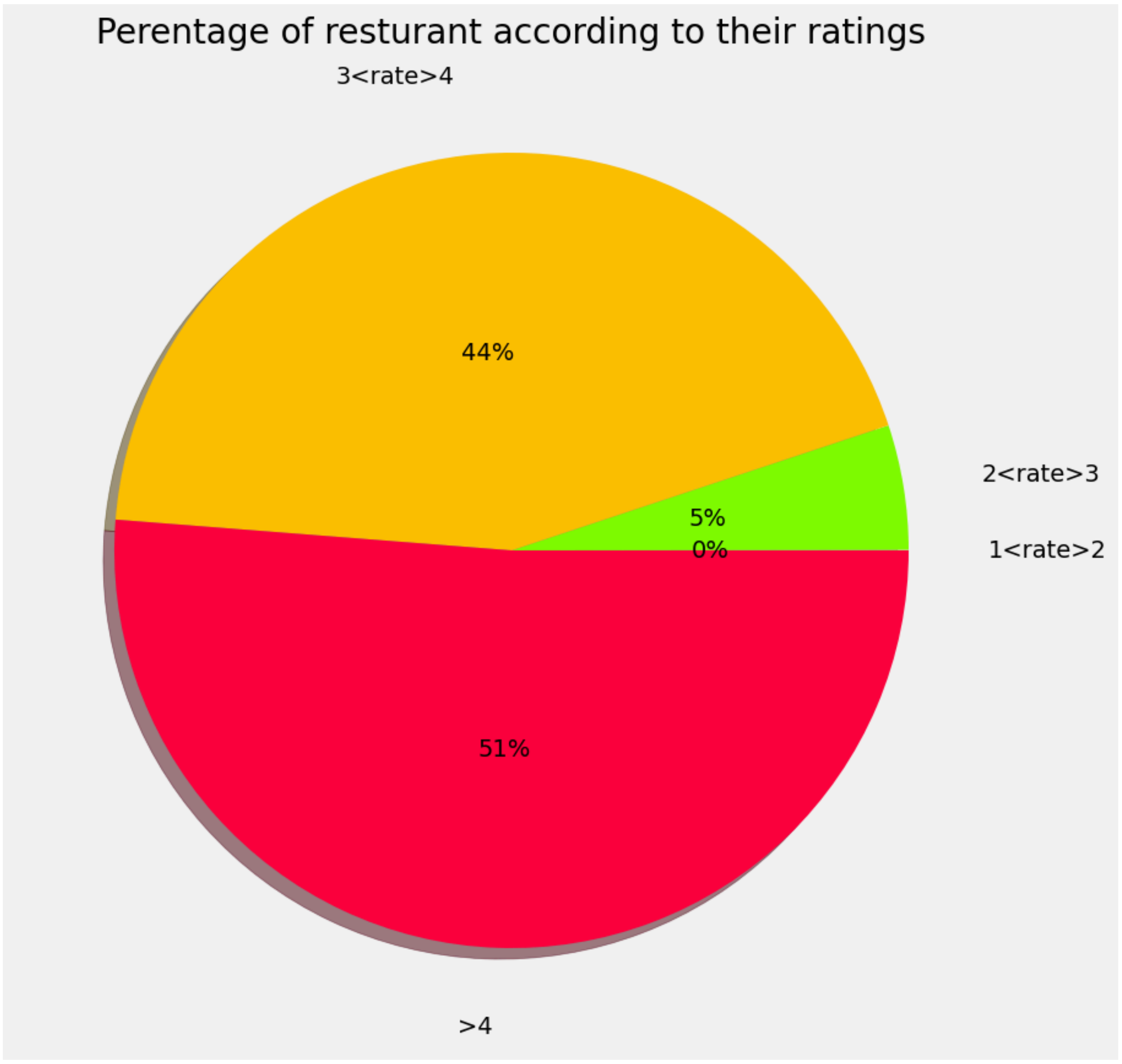
Out[159]:
```
1179
```

In [160]:
```
((data['rating']>=3)& (data['rating']<4)).sum()
```

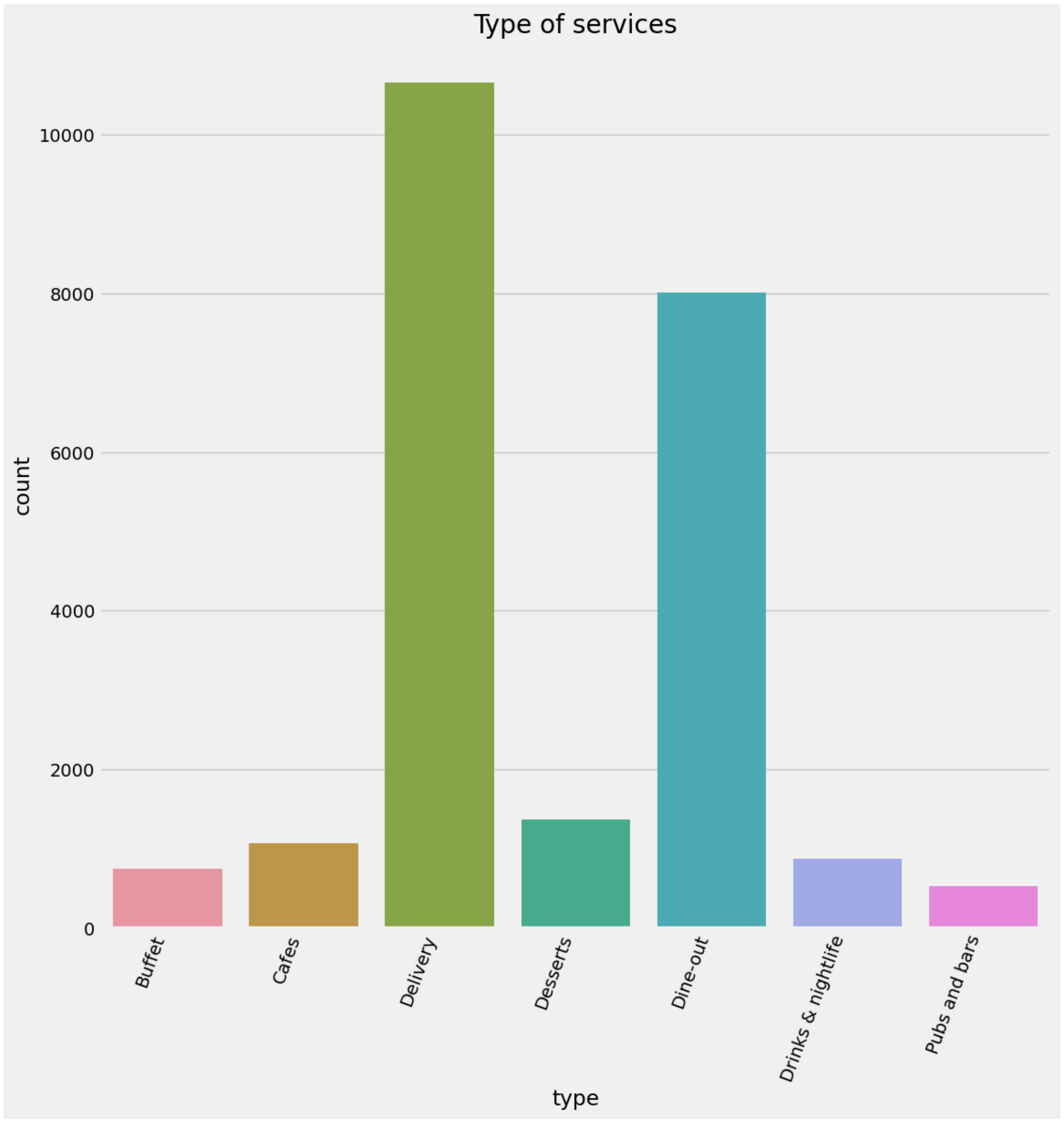
Out[160]:
```
10153
```

```
In [161]: ((data['rating']>=4) & (data['rating']<5)).sum()

Out[161]: 11911
```

```
In [162]: slices=[((data['rating'] >= 1) & (data['rating'] < 2)).sum(),
                  ((data['rating']>=2) & (data['rating']<3)).sum(),
                  ((data['rating']>=3)& (data['rating']<4)).sum(),
                  ((data['rating']>=4) & (data['rating']<5)).sum()
          ]

          labels=['1<rate>2','2<rate>3','3<rate>4','>4']
          colors=['#fba2fc','#7ffc03','#fcbe03','#fc033d']
          plt.pie(slices,colors=colors,labels=labels,autopct='%1.0f%%',pctdistance=.5,lal
          fig=plt.gcf()
          plt.title('Perentage of resturant according to their ratings')
          fig.set_size_inches(10,10)
          plt.show()
```



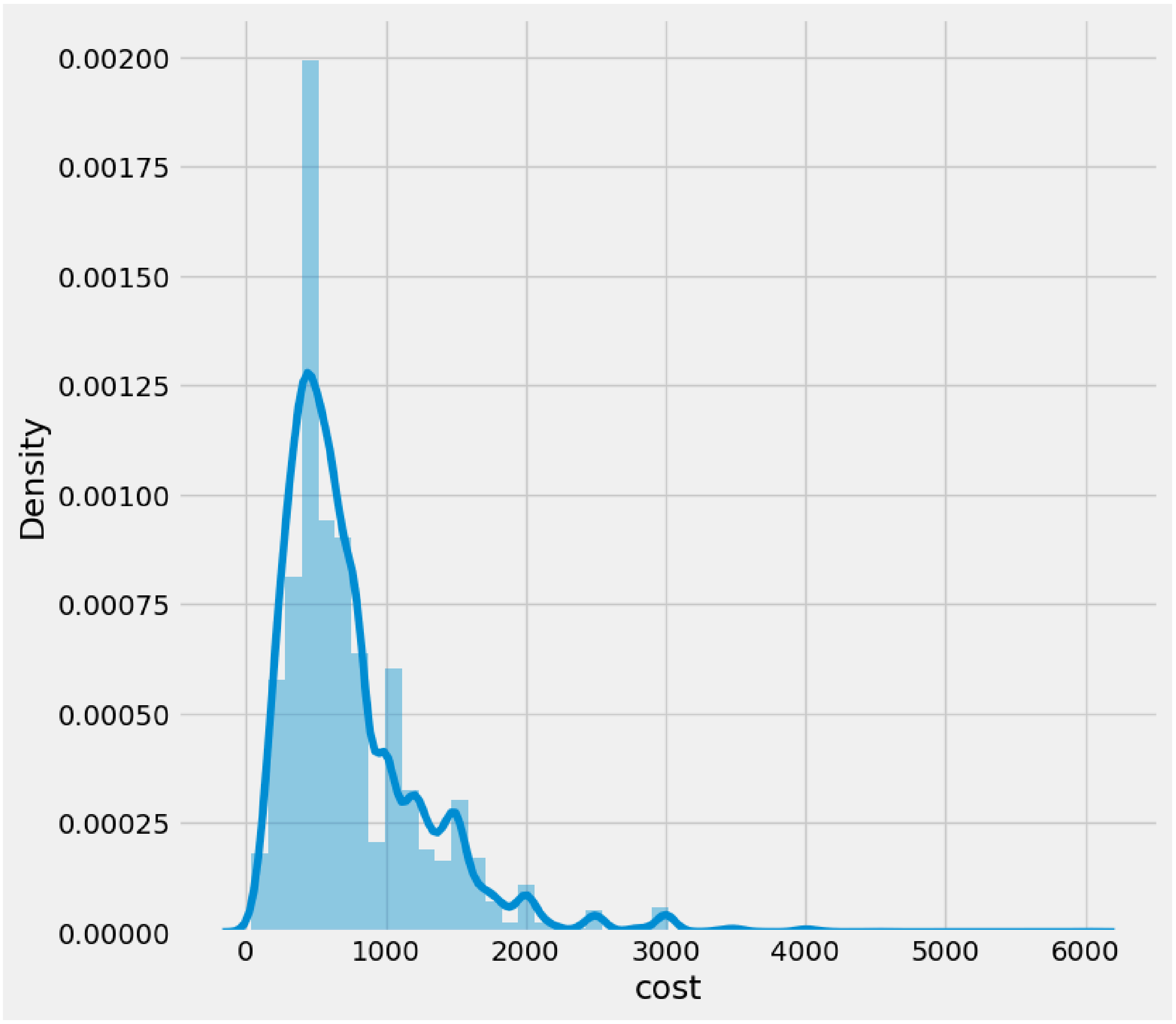
In [163]:
```
ax = sns.countplot(x='type', data=data)

# Rotate the x-tick labels
ax.set_xticklabels(ax.get_xticklabels(), rotation=70, ha='right')

# Set the figure size
fig = plt.gcf()
fig.set_size_inches(12, 12)

# Set the title
plt.title('Type of services')

# Show the plot
plt.show()
```



In [164]:
```
#Distribution of cost of food for two people
```

In [165]:
```
trace0=go.Box(y=data['cost'],name='accepting online orders',marker=dict(color=
df=[trace0]
layout=go.Layout(title='Box plot of approximate cost',width=800,height=800,yax
fig=go.Figure(data=df,layout=layout)
py.iplot(fig)
```

In [169]:
```
plt.figure(figsize=(8,8))
sns.distplot(data['cost'])
plt.show()
```



In [171]:
```
import re

data.index=range(data.shape[0])
likes=[]
for i in range(data.shape[0]):
    array_split=re.split(',',data['dish_liked'][i])
    for item in array_split:
        likes.append(item)
```

In [172]:
```
data.index=range(data.shape[0])
```

In [173]:
```
data.index
```

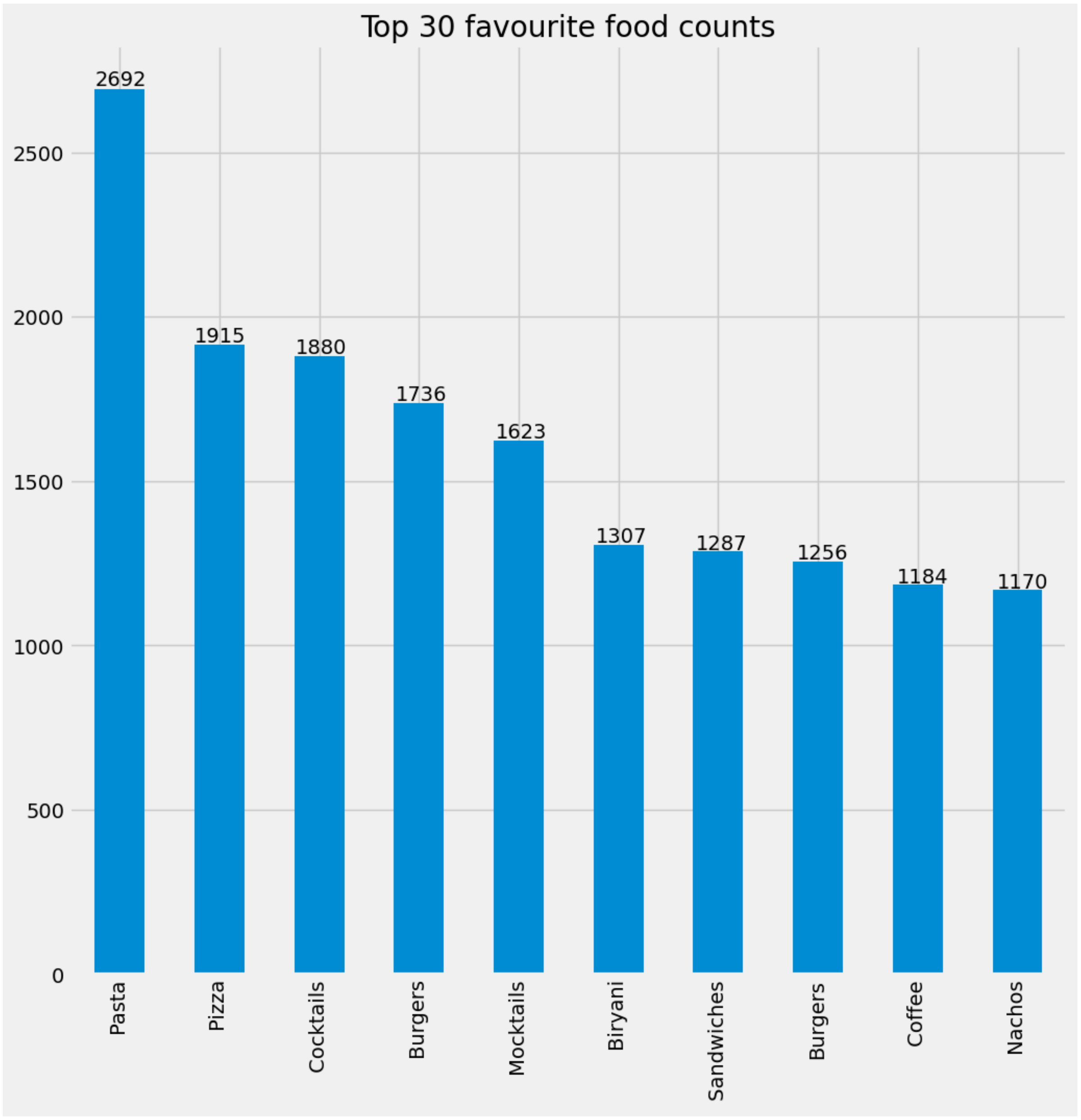
Out[173]:
```
RangeIndex(start=0, stop=23248, step=1)
```

In [179]:
```
print("Count the most liked dishes in Bangaluru")
fav_food=pd.Series(likes).value_counts()
fav_food.head(30)
```

```
Count the most liked dishes in Bangaluru
```

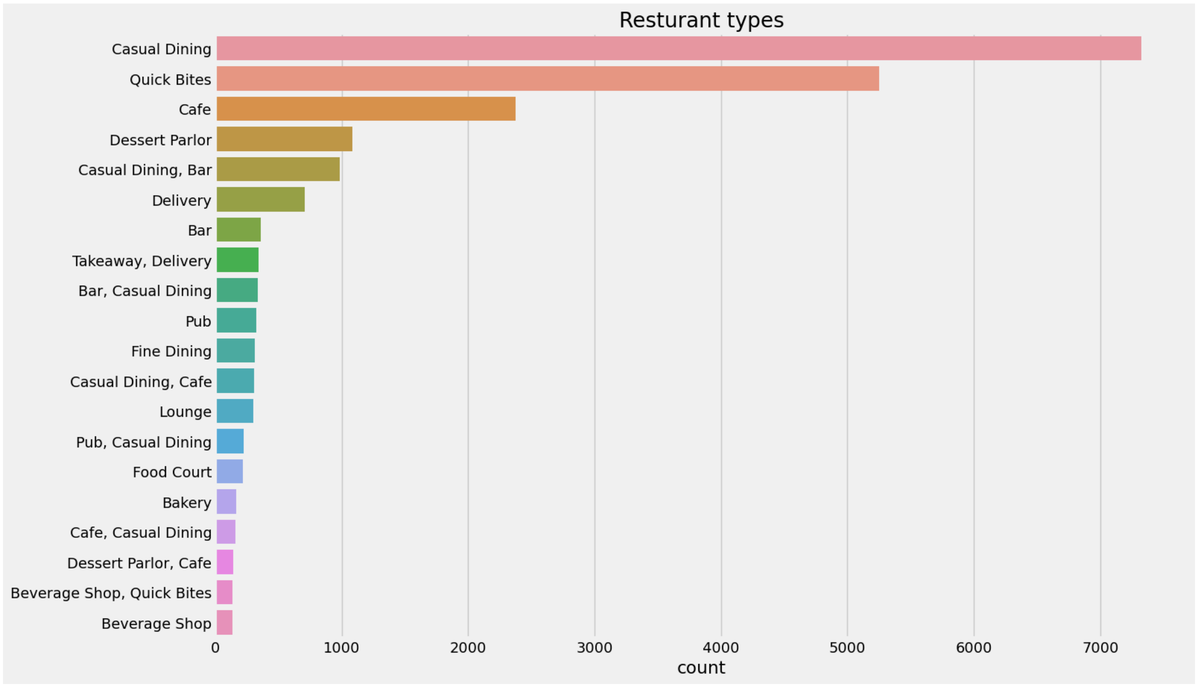
Out[179]:
```
 Pasta               2692
 Pizza               1915
 Cocktails           1880
 Burgers             1736
 Mocktails           1623
 Biryani             1307
 Sandwiches          1287
Burgers              1256
 Coffee              1184
 Nachos              1170
 Fish                1116
 Paratha             1107
 Salads              1055
 Chicken Biryani     1004
Cocktails             891
 Fries                876
 Noodles              854
 Beer                 835
 Mutton Biryani       832
 Tea                  819
Coffee                801
 Sandwich             788
 Butter Chicken       782
 Thali                770
Biryani               749
Pizza                 747
 Roti                 729
 Brownie              726
 Salad                677
 Hot Chocolate        672
dtype: int64
```

In [182]:
```
ax=fav_food.nlargest(n=10,keep='first').plot(kind='bar',figsize=(11,11),title=

for i in ax.patches:
    ax.annotate(str(i.get_height()),(i.get_x()*1.005,i.get_height()*1.005))
```



In [183]:
```
#Resturants and there counts
```

In [189]:
```
plt.figure(figsize=(15,10))
rest=data['rest_type'].value_counts()[:20]
sns.barplot(y=rest.index,x=rest,orient='h')
plt.title('Resturant types')
plt.xlabel("count")
plt.show()
```



In [ ]:

In [190]:
```
#Now for model building we have to prepare the data
```

In [191]:
```python
data.head()
```

Out[191]:

	address	name	online_order	book_table	rating	votes	location	rest_type	dish
0	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1	775	Banashankari	Casual Dining	I I I
1	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1	787	Banashankari	Casual Dining	M Chc N Tl
2	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8	918	Banashankari	Cafe, Casual Dining	Cl Canı Mine Sou (
3	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7	88	Banashankari	Quick Bites	I
4	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No	3.8	166	Basavanagudi	Casual Dining	Pa Gol

In [192]:
```python
data.online_order[data.online_order=='Yes']=1
data.online_order[data.online_order=='No']=0
```

In [193]:
```python
data['online_order'].value_counts()
```

Out[193]:
```
1    16378
0     6870
Name: online_order, dtype: int64
```

In [194]:
```python
data.book_table[data.book_table=='Yes']=1
data.book_table[data.book_table=='No']=0
```

In [195]:
```python
data['book_table'].value_counts()
```

Out[195]:
```
0    17191
1     6057
Name: book_table, dtype: int64
```

In [197]:
```python
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [198]:
```
data['location']=le.fit_transform(data['location'])
data['rest_type']=le.fit_transform(data['rest_type'])
data['cuisines']=le.fit_transform(data['cuisines'])
data['menu_item']=le.fit_transform(data['menu_item'])
```

In [199]:
```
data.head()
```

Out[199]:

	address	name	online_order	book_table	rating	votes	location	rest_type	dish_like
0	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	1	1	4.1	775	1	20	Pasta Lunc Buffe Masal Papac Panee Laja.
1	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	1	0	4.1	787	1	20	Momos Lunc Buffe Chocolat Nirvana Thai G.
2	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	1	0	3.8	918	1	16	Churros Cannellon Minestron Soup, Hc Choc.
3	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	0	0	3.7	88	1	62	Masal Dos
4	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	0	0	3.8	166	4	20	Panipur Gol Gapp

In [200]:
```
my_data=data.iloc[:,[2,3,4,5,6,7,9,10,12]]
my_data.to_csv('zomato_df.csv')
```

In [201]:
```
x=data.iloc[:,[2,3,5,6,7,9,10,12]]
x.head()
```

Out[201]:

	online_order	book_table	votes	location	rest_type	cuisines	cost	menu_item
0	1	1	775	1	20	1386	800.0	5047
1	1	0	787	1	20	594	800.0	5047
2	1	0	918	1	16	484	800.0	5047
3	0	0	88	1	62	1587	300.0	5047
4	0	0	166	4	20	1406	600.0	5047

In [204]:
```
y=data['rating']
```

In [205]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state
```

LinearRegression

In [206]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[206]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [207]:
```
from sklearn.metrics import r2_score
y_pred=lr.predict(x_test)
r2_score(y_test,y_pred)
```

Out[207]:
```
0.22762342262807467
```

RandomForest

In [209]:
```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor(n_estimators=650,random_state=120,min_samples_leaf=0.
rf.fit(x_train,y_train)
y_pred=rf.predict(x_test)
r2_score(y_test,y_pred)
```

Out[209]:
```
0.8920067629047455
```

ExtraTreesRegressor

In [211]:
```
from sklearn.ensemble import ExtraTreesRegressor
et=ExtraTreesRegressor(n_estimators=120)
et.fit(x_train,y_train)
y_pred=et.predict(x_test)
r2_score(y_pred,y_test)
```

Out[211]:
```
0.9299835257201194
```

Pickle

#Use pickle to save our model so that we can use it later

In [212]:
```
import pickle
pickle.dump(et,open('model.pkl','wb'))
model=pickle.load(open('model.pkl','rb'))
```