Data Analysis

In [ ]:
```
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt # plotting
import numpy as np # linear algebra
import os # accessing directory structure
import pandas as pd
```

Loading the dataset

In [11]:
```
df = pd.read_csv('zomato.csv', error_bad_lines = False, engine='python')
```

```
Skipping line 1130: field larger than field limit (131072)
Skipping line 2046: field larger than field limit (131072)
Skipping line 2588: field larger than field limit (131072)
Skipping line 4133: field larger than field limit (131072)
Skipping line 4244: field larger than field limit (131072)
Skipping line 4804: field larger than field limit (131072)
Skipping line 4806: field larger than field limit (131072)
Skipping line 4810: field larger than field limit (131072)
Skipping line 4812: field larger than field limit (131072)
Skipping line 4946: field larger than field limit (131072)
Skipping line 4947: field larger than field limit (131072)
Skipping line 5033: field larger than field limit (131072)
Skipping line 5176: field larger than field limit (131072)
Skipping line 5300: field larger than field limit (131072)
Skipping line 5328: field larger than field limit (131072)
Skipping line 5497: field larger than field limit (131072)
Skipping line 5711: field larger than field limit (131072)
Skipping line 5932: field larger than field limit (131072)
Skipping line 6048: field larger than field limit (131072)
```

In [12]:
```
df.dataframeName = 'zomato.csv'
```

In [13]:
```
nRow, nCol = df.shape
print(f'There are {nRow} rows and {nCol} columns')
```

```
There are 51155 rows and 17 columns
```

In [14]:
```
df.index
```

Out[14]:
```
RangeIndex(start=0, stop=51155, step=1)
```

In [15]:
```
df.loc[1]
```

...

Knowing the dataset

In [16]:
```
1 df.columns
```

Out[16]:
```
Index(['url', 'address', 'name', 'online_order', 'book_table', 'rate', 'votes',
       '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 [17]:
```
1 df['listed_in(city)'].unique()
```

Out[17]:
```
array(['Banashankari', 'Bannerghatta Road', 'Basavanagudi', 'Bellandur',
       'Brigade Road', 'Brookefield', 'BTM', 'Church Street',
       'Electronic City', 'Frazer Town', 'HSR', 'Indiranagar',
       'Jayanagar', 'JP Nagar', 'Kalyan Nagar', 'Kammanahalli',
       'Koramangala 4th Block', 'Koramangala 5th Block',
       'Koramangala 6th Block', 'Koramangala 7th Block', 'Lavelle Road',
       'Malleshwaram', 'Marathahalli', 'MG Road', 'New BEL Road',
       'Old Airport Road', 'Rajajinagar', 'Residency Road',
       'Sarjapur Road', 'Whitefield'], dtype=object)
```

Getting the Unique

In [18]:
```
1 df['location'].unique()
2
```

Out[18]:
```
array(['Banashankari', 'Basavanagudi', 'Mysore Road', 'Jayanagar',
       'Kumaraswamy Layout', 'Rajarajeshwari Nagar', 'Vijay Nagar',
       'Uttarahalli', 'JP Nagar', 'South Bangalore', 'City Market',
       'Nagarbhavi', 'Bannerghatta Road', 'BTM', 'Kanakapura Road',
       'Bommanahalli', nan, 'CV Raman Nagar', 'Electronic City', 'HSR',
       'Marathahalli', 'Sarjapur Road', 'Wilson Garden', 'Shanti Nagar',
       'Koramangala 5th Block', 'Koramangala 8th Block', 'Richmond Road',
       'Koramangala 7th Block', 'Jalahalli', 'Koramangala 4th Block',
       'Bellandur', 'Whitefield', 'East Bangalore', 'Old Airport Road',
       'Indiranagar', 'Koramangala 1st Block', 'Frazer Town', 'RT Nagar',
       'MG Road', 'Brigade Road', 'Lavelle Road', 'Church Street',
       'Ulsoor', 'Residency Road', 'Shivajinagar', 'Infantry Road',
       'St. Marks Road', 'Cunningham Road', 'Race Course Road',
       'Commercial Street', 'Vasanth Nagar', 'HBR Layout', 'Domlur',
       'Ejipura', 'Jeevan Bhima Nagar', 'Old Madras Road', 'Malleshwaram',
       'Seshadripuram', 'Kammanahalli', 'Koramangala 6th Block',
       'Majestic', 'Langford Town', 'Central Bangalore', 'Sanjay Nagar',
       'Brookefield', 'ITPL Main Road, Whitefield',
       'Varthur Main Road, Whitefield', 'KR Puram',
       'Koramangala 2nd Block', 'Koramangala 3rd Block', 'Koramangala',
       'Hosur Road', 'Rajajinagar', 'Banaswadi', 'North Bangalore',
       'Nagawara', 'Hennur', 'Kalyan Nagar', 'New BEL Road', 'Jakkur',
       'Rammurthy Nagar', 'Thippasandra', 'Kaggadasapura', 'Hebbal',
       'Kengeri', 'Sankey Road', 'Sadashiv Nagar', 'Basaveshwara Nagar',
       'Yeshwantpur', 'West Bangalore', 'Magadi Road', 'Yelahanka',
       'Sahakara Nagar', 'Peenya'], dtype=object)
```

In [19]:
```
1 df.loc[:,'rate']
```

Out[19]:
```
0        4.1/5
1        4.1/5
2        3.8/5
3        3.7/5
4        3.8/5
         ...
51150    3.6 /5
51151       NaN
51152       NaN
51153    4.3 /5
51154    3.4 /5
Name: rate, Length: 51155, dtype: object
```

In [20]:
```
1 df.head(5)
```

Out[20]:

	url	address	name	online_order	book_table	rat
0	https://www.zomato.com/bangalore/jalsa-banasha...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1
1	https://www.zomato.com/bangalore/spice-elephan...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8
3	https://www.zomato.com/bangalore/addhuri-udupi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7
4	https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No	3.8

Dropping the cells

In [21]:
```
1 df.drop(['url', 'address', 'phone', 'listed_in(city)'], axis = 1, inplace =
```

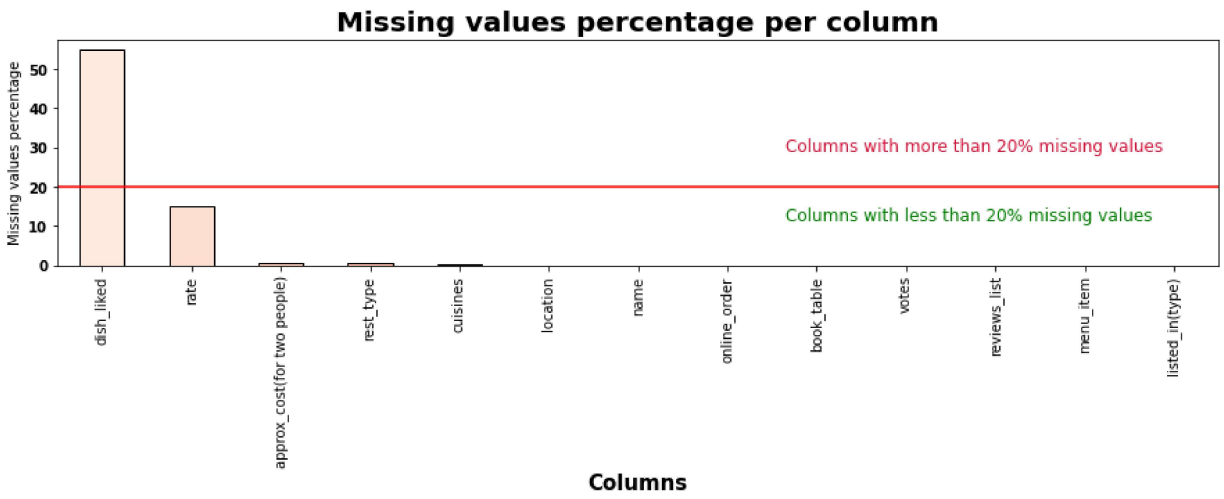
In [22]:
```
1 df.isnull().sum()
```

Out[22]:
```
name                                0
online_order                        0
book_table                          0
rate                             7775
votes                               0
location                           21
rest_type                         224
dish_liked                      28074
cuisines                           45
approx_cost(for two people)       344
reviews_list                        0
menu_item                           0
listed_in(type)                     0
dtype: int64
```

Graph for missing values as percentage

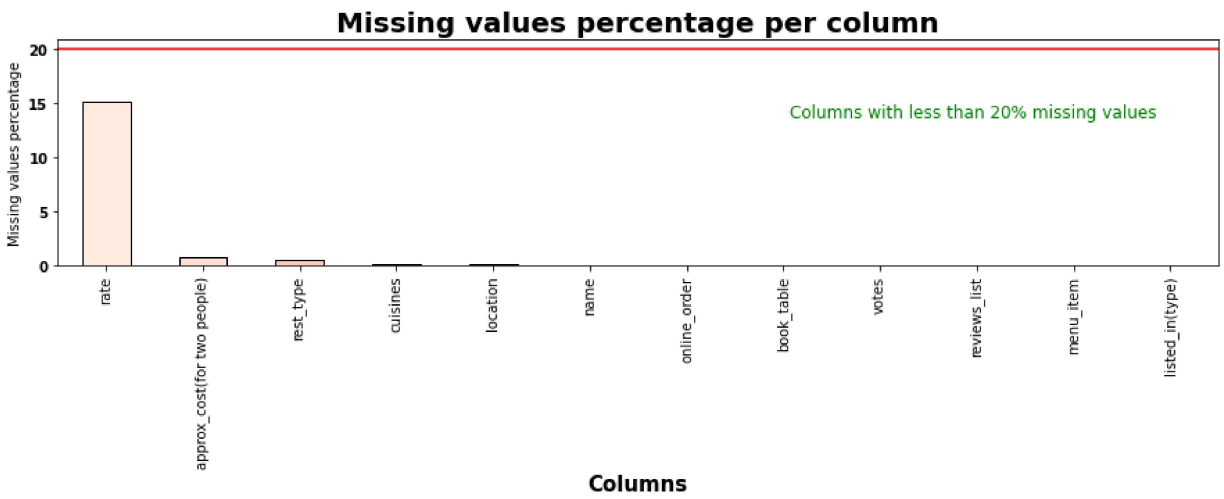
In [23]:
```
1  import matplotlib.pyplot as plt
2  import seaborn as sns
3  %matplotlib inline
4
5  def missing_values(data, thresh = 20, color = 'black', edgecolor = 'black',
6
7      plt.figure(figsize = (width, height))
8      percentage = (data.isnull().mean()) * 100
9      percentage.sort_values(ascending = False).plot.bar(color = color, edgeco
10     plt.axhline(y = thresh, color = 'r', linestyle = '-')
11
12     plt.title('Missing values percentage per column', fontsize = 20, weight
13
14     plt.text(len(data.isnull().sum()/len(data))/1.7, thresh + 12.5, f'Column
15          ha = 'left' ,va = 'top')
16     plt.text(len(data.isnull().sum()/len(data))/1.7, thresh - 5, f'Columns w
17          ha = 'left' ,va = 'top')
18     plt.xlabel('Columns', size = 15, weight = 'bold')
19     plt.ylabel('Missing values percentage')
20     plt.yticks(weight = 'bold')
21
22     return plt.show()
23
```

In [24]:
```
1 missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))
```



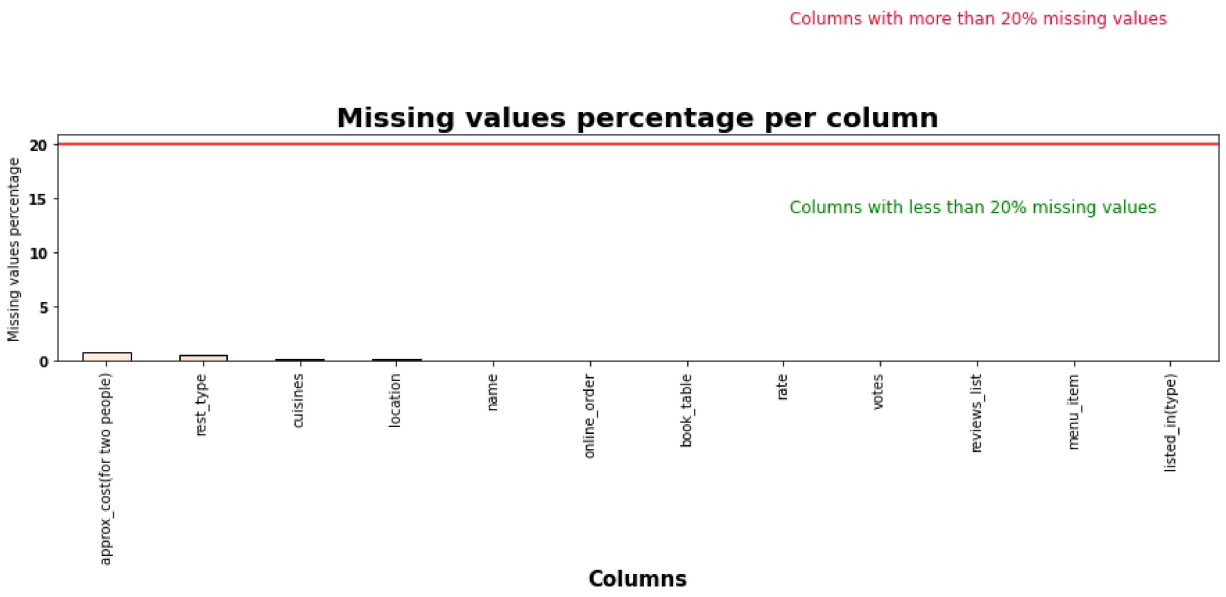
In [25]:
```
1 df.drop(['dish_liked'], axis = 1, inplace = True)
```

In [26]:
```
1 missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))
```



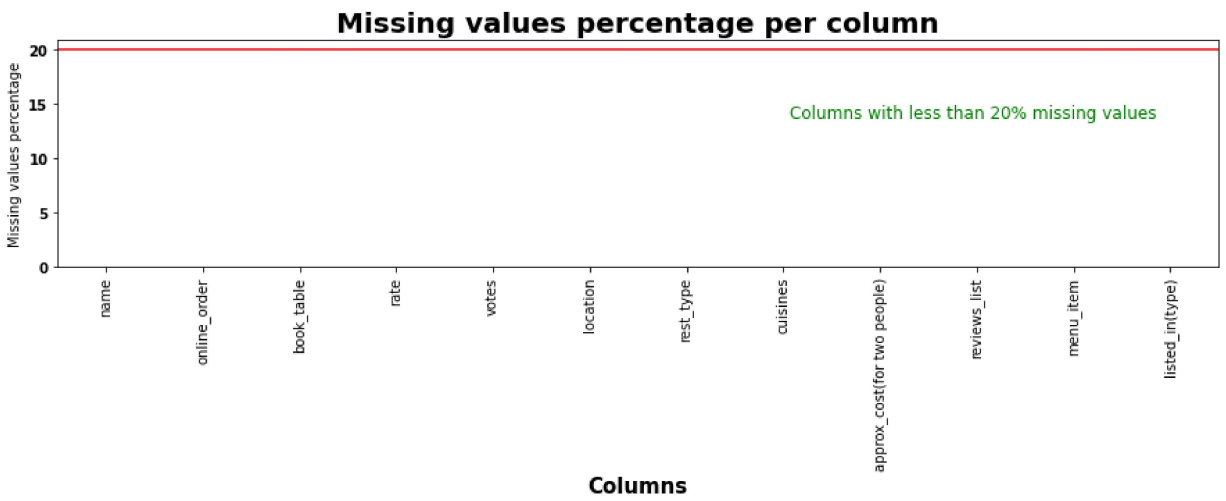
In [27]:
```
1 df['rate'] = df['rate'].fillna(df['rate'].mode()[0])
```

In [28]:
```
1 missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))
```



In [29]:
```
1 previous_df = df
2 df.dropna(inplace = True)
3
```

In [30]:
```
1 missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))
```



Data cleaning

In [31]:
```
1 df.isnull().sum().sum()
```

Out[31]: 0

In [32]:
```
print("Number of rows present in the dataset are: ", df.shape[0])
print("Number of columns present in the dataset are: ", df.shape[1])
```

```
Number of rows present in the dataset are:  50591
Number of columns present in the dataset are:  12
```

In [33]:
```
df['rate'].value_counts()
```

Out[33]:
```
NEW       9818
3.9/5     2086
3.7/5     2003
3.8/5     1994
3.9 /5    1852
          ...
2.2 /5       7
2.1 /5       7
2.0/5        4
1.8/5        2
1.8 /5       1
Name: rate, Length: 64, dtype: int64
```

In [34]:
```
print("rate column type: ", type(df['rate'][0]))
```

```
rate column type:  <class 'str'>
```

In [35]:
```
df['rate'].replace({"NEW" : "2.5 /5"}, inplace = True)
```

In [36]:
```
df['rate'].unique()

```

Out[36]:
```
array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
       '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
       '4.3/5', '2.5 /5', '2.9/5', '3.5/5', '2.6/5', '3.8 /5', '3.4/5',
       '4.5/5', '2.5/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5',
       '3.4 /5', '-', '3.6 /5', '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5',
       '4.1 /5', '3.7 /5', '3.1 /5', '2.9 /5', '3.3 /5', '2.8 /5',
       '3.5 /5', '2.7 /5', '3.2 /5', '2.6 /5', '4.5 /5', '4.3 /5',
       '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '4.6 /5', '4.9 /5',
       '3.0 /5', '4.8 /5', '2.3 /5', '2.4 /5', '4.7 /5', '2.2 /5',
       '2.0 /5', '2.1 /5', '1.8 /5'], dtype=object)
```

In [37]:
```
1 df.loc[df['rate'] == '-'].head(5)
```

Out[37]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	approx_two
3065	House of Tasty Food	No	No	-	0	Wilson Garden	Quick Bites	North Indian	
3370	Right Pizza	Yes	No	-	0	Basavanagudi	Quick Bites	Pizza	
3375	Mezban Family Restaurant	Yes	No	-	0	Basavanagudi	Quick Bites	Chinese, North Indian	
3384	Mota Bawarchi	No	No	-	0	Basavanagudi	Quick Bites	North Indian, Biryani, Fast Food	
3393	Aahar Cafe	No	No	-	0	Basavanagudi	Quick Bites	South Indian	

In [38]:
```
1 a = df.loc[df['rate'] == '-'].index
2 df.drop(a, axis = 0, inplace = True)
3
4 df['rate'].unique()
5
```

Out[38]:
```
array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
       '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
       '4.3/5', '2.5 /5', '2.9/5', '3.5/5', '2.6/5', '3.8 /5', '3.4/5',
       '4.5/5', '2.5/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5',
       '3.4 /5', '3.6 /5', '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5',
       '4.1 /5', '3.7 /5', '3.1 /5', '2.9 /5', '3.3 /5', '2.8 /5',
       '3.5 /5', '2.7 /5', '3.2 /5', '2.6 /5', '4.5 /5', '4.3 /5',
       '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '4.6 /5', '4.9 /5',
       '3.0 /5', '4.8 /5', '2.3 /5', '2.4 /5', '4.7 /5', '2.2 /5',
       '2.0 /5', '2.1 /5', '1.8 /5'], dtype=object)
```

In [39]:
```
1 df['rate'] = df['rate'].apply(lambda x : x.split('/')[0])
```

In [40]:
```
1 df['rate'].unique()
```

Out[40]:
```
array(['4.1', '3.8', '3.7', '3.6', '4.6', '4.0', '4.2', '3.9', '3.1',
       '3.0', '3.2', '3.3', '2.8', '4.4', '4.3', '2.5 ', '2.9', '3.5',
       '2.6', '3.8 ', '3.4', '4.5', '2.5', '2.7', '4.7', '2.4', '2.2',
       '2.3', '3.4 ', '3.6 ', '4.8', '3.9 ', '4.2 ', '4.0 ', '4.1 ',
       '3.7 ', '3.1 ', '2.9 ', '3.3 ', '2.8 ', '3.5 ', '2.7 ', '3.2 ',
       '2.6 ', '4.5 ', '4.3 ', '4.4 ', '4.9', '2.1', '2.0', '1.8', '4.6 ',
       '4.9 ', '3.0 ', '4.8 ', '2.3 ', '2.4 ', '4.7 ', '2.2 ', '2.0 ',
       '2.1 ', '1.8 '], dtype=object)
```

In [41]:
```
1 df['rate'] = df['rate'].apply(lambda x : x.split(' ')[0])
```

In [42]:
```
1 df['rate'].unique()
```

Out[42]:
```
array(['4.1', '3.8', '3.7', '3.6', '4.6', '4.0', '4.2', '3.9', '3.1',
       '3.0', '3.2', '3.3', '2.8', '4.4', '4.3', '2.5', '2.9', '3.5',
       '2.6', '3.4', '4.5', '2.7', '4.7', '2.4', '2.2', '2.3', '4.8',
       '4.9', '2.1', '2.0', '1.8'], dtype=object)
```

In [43]:
```
1 df = df.astype({'rate' : float})
2 print(df['rate'].dtype)
```

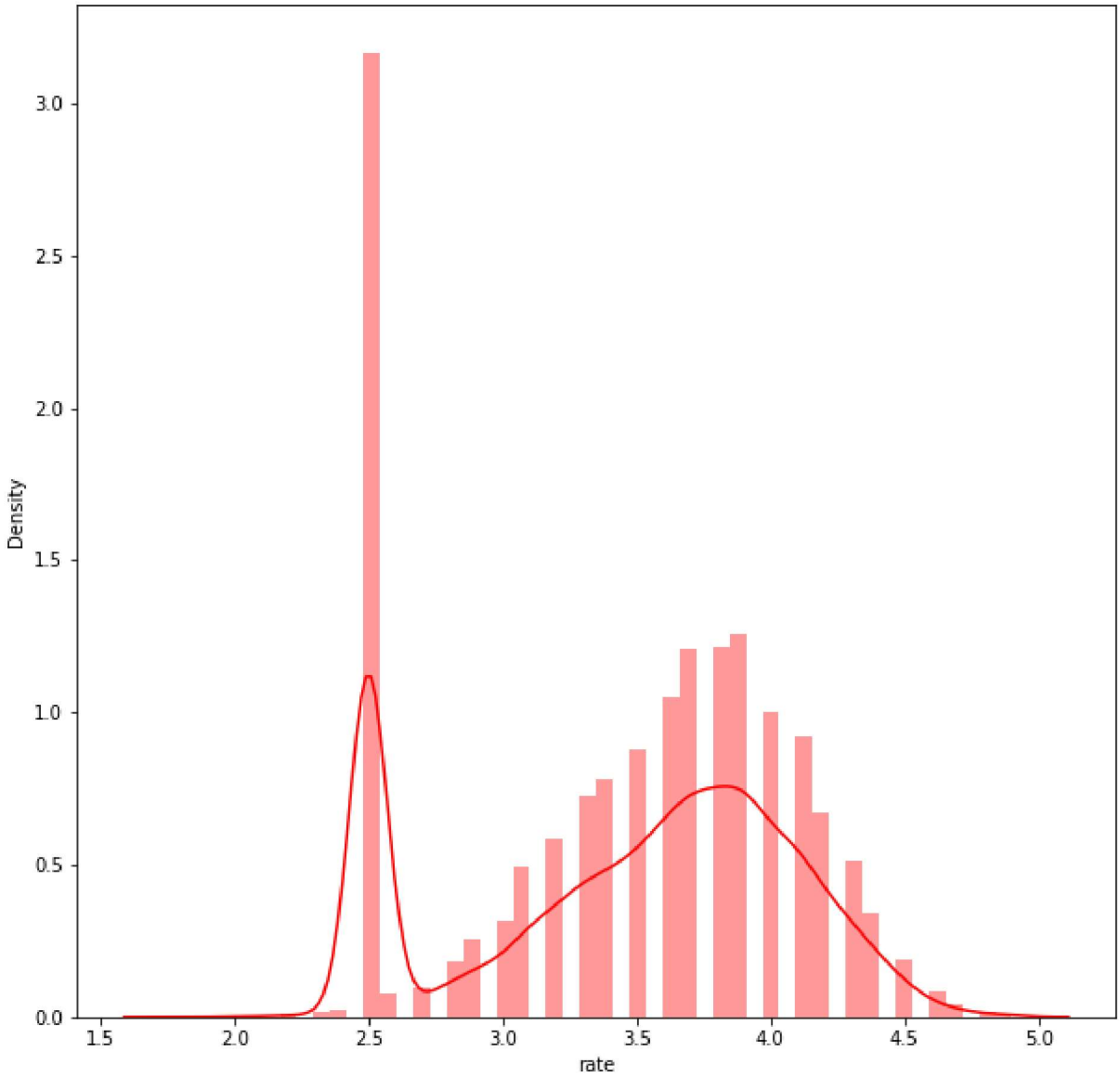
```
float64
```

Bivariate Analysis

In [44]:
```
1 plt.figure(figsize = (10, 10))
2 sns.distplot(df['rate'], color = 'red')
3
```

```
c:\users\sasi\appdata\local\programs\python\python38-32\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

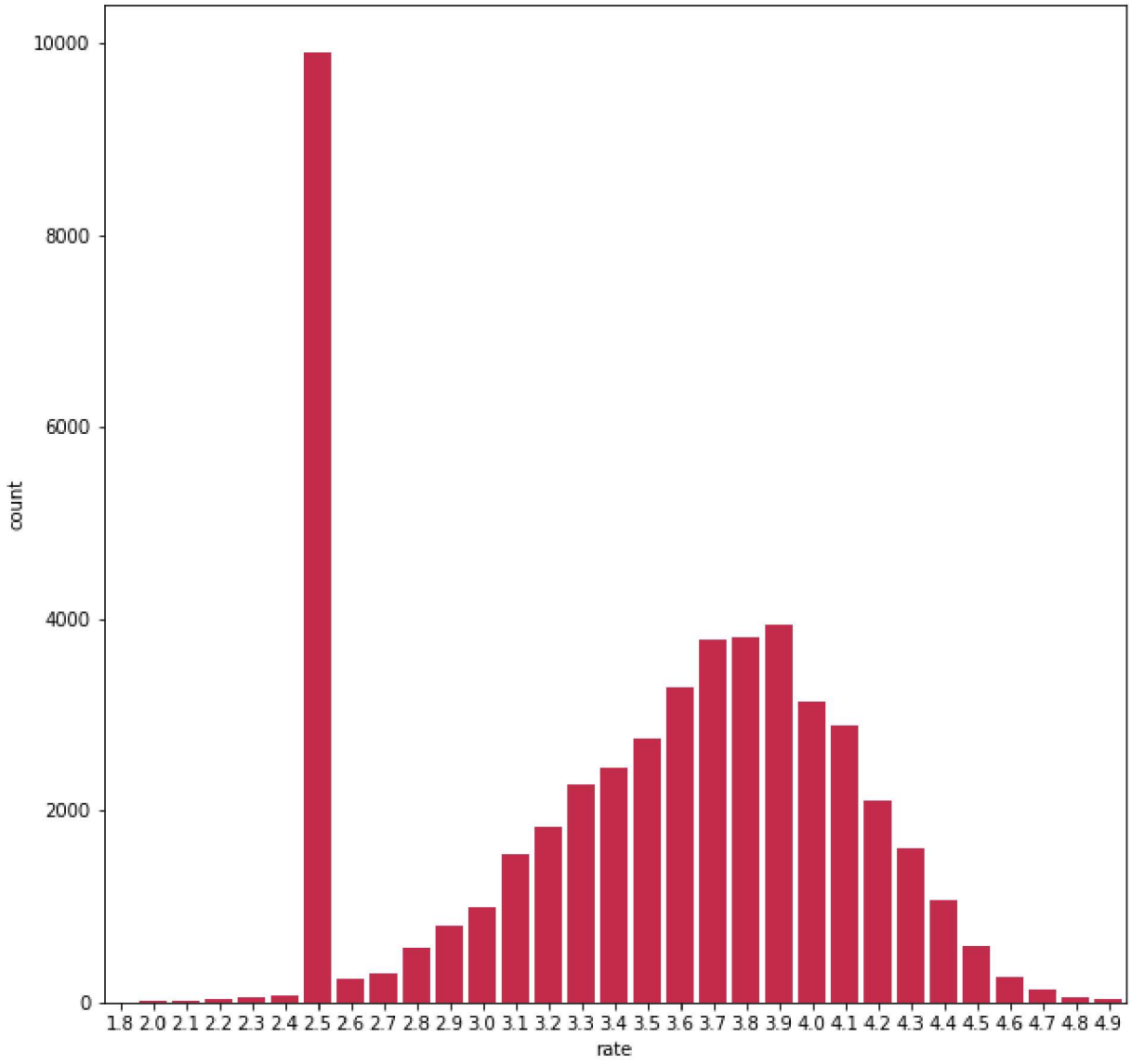
Out[44]: <AxesSubplot:xlabel='rate', ylabel='Density'>



In [45]:
```
1 plt.figure(figsize = (10, 10))
2 sns.countplot(df['rate'], color = 'crimson')
```

```
c:\users\sasi\appdata\local\programs\python\python38-32\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[45]: <AxesSubplot:xlabel='rate', ylabel='count'>

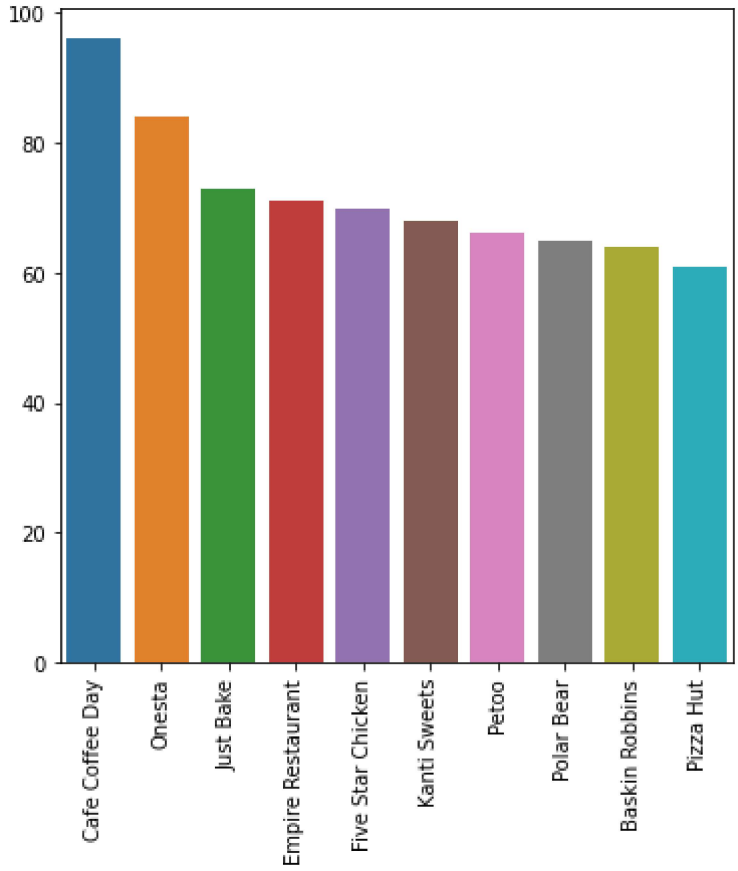


In [46]:
```
1 print("Restaurants having the highest ratings are: \n", df.loc[(df['rate'] =
```

```
Restaurants having the highest ratings are:
 ["Brahmin's Coffee Bar" 'The Black Pearl' 'Byg Brewski Brewing Company'
 'Belgian Waffle Factory' "AB's - Absolute Barbecues" 'Flechazo'
 'Punjab Grill' 'Asia Kitchen By Mainland China' 'House Of Commons'
 'The Pizza Bakery' 'Milano Ice Cream' 'Barbeque Nation' 'Biergarten'
 "Chili's American Grill & Bar" 'O.G. Variar & Sons' 'The Globe Grub'
 'The Boozy Griffin'
 'SantÃ\x83Â\x83Ã\x82Â\x83Ã\x83Â\x82Ã\x82Â\x83Ã\x83Â\x83Ã\x82Â\x82Ã\x83Â\x82Ã\x
82Â\x83Ã\x83Â\x83Ã\x82Â\x83Ã\x83Â\x82Ã\x82Â\x82Ã\x83Â\x83Ã\x82Â\x82Ã\x83Â\x82Ã
\x82Â© Spa Cuisine'
 'CTR' 'Barbecue by Punjab Grill']
```

In [47]:
```
plt.figure(figsize = (6, 6))
df['name'].value_counts().head(10)
index = df['name'].value_counts().head(10).index
values = df['name'].value_counts().head(10).values
g = sns.barplot(x = index, y = values, data = df)
g.set_xticklabels(labels = index, rotation = 90)
g
```

Out[47]: <AxesSubplot:>



In [48]:
```
1 df.groupby('name')['rate'].mean().sort_values(ascending = False).head(15)
```

Out[48]:
```
name
SantÃ‚Ã‚Ã‚Ã
```

In [49]:
```
1 df.groupby('name')['votes', 'rate'].max().sort_values(ascending = False, by
```

```
<ipython-input-49-45c3b3c43393>:1: FutureWarning: Indexing with multiple keys
(implicitly converted to a tuple of keys) will be deprecated, use a list instea
d.
  df.groupby('name')['votes', 'rate'].max().sort_values(ascending = False, by =
'votes').head(15)
```

Out[49]:

name	votes	rate
Byg Brewski Brewing Company	16832	4.9
Toit	14956	4.7
Truffles	14723	4.7
AB's - Absolute Barbecues	12121	4.9
The Black Pearl	10547	4.8
Onesta	9085	4.6
Big Pitcher	9041	4.7
Arbor Brewing Company	8414	4.5
Empire Restaurant	8304	4.4
Prost Brew Pub	7870	4.5
Church Street Social	7584	4.3
Hoot	7330	4.2
Barbeque Nation	7270	4.8
Meghana Foods	7238	4.5
Flechazo	7154	4.9

Knowing the best restaurant in a location

In [50]:
```
names = df.groupby('name')['rate'].mean().sort_values(ascending = False).hea
locations = []
for i in names:
    loc = []
    locations.append(df.loc[df['name'] == i]['location'].unique().tolist())
name_location = dict(zip(names, locations))
name_location
```

Out[50]:
```
{'SantÃ\x83Â\x83Ã\x82Â\x83Ã\x83Â\x82Ã\x82Â\x83Ã\x83Â\x83Ã\x82Â\x82Ã\x83Â\x82Ã\x
82Â\x83Ã\x83Â\x83Ã\x82Â\x83Ã\x83Â\x82Ã\x82Â\x82Ã\x83Â\x83Ã\x82Â\x82Ã\x83Â\x82Ã
\x82Â© Spa Cuisine': ['Indiranagar'],
 'Byg Brewski Brewing Company': ['Sarjapur Road'],
 'Asia Kitchen By Mainland China': ['Koramangala 5th Block'],
 'Punjab Grill': ['Whitefield', 'Malleshwaram'],
 'Flechazo': ['Whitefield', 'Marathahalli'],
 'Belgian Waffle Factory': ['Brigade Road',
  'Koramangala 5th Block',
  'Vasanth Nagar',
  'Kalyan Nagar'],
 'The Pizza Bakery': ['Indiranagar'],
 'O.G. Variar & Sons': ['Rajajinagar'],
 "AB's - Absolute Barbecues": ['Sarjapur Road',
  'Whitefield',
  'BTM',
  'Kalyan Nagar',
  'Marathahalli'],
 'Barbecue by Punjab Grill': ['Malleshwaram'],
 'CTR': ['Malleshwaram'],
 'House Of Commons': ['Koramangala 5th Block', 'HSR'],
 'The Black Pearl': ['Marathahalli', 'Koramangala 5th Block'],
 'Burma Burma': ['Indiranagar'],
 'TBC Sky Lounge': ['HSR']}
```

In [51]:
```
df['online_order'].unique()
```

Out[51]:
```
array(['Yes', 'No'], dtype=object)
```

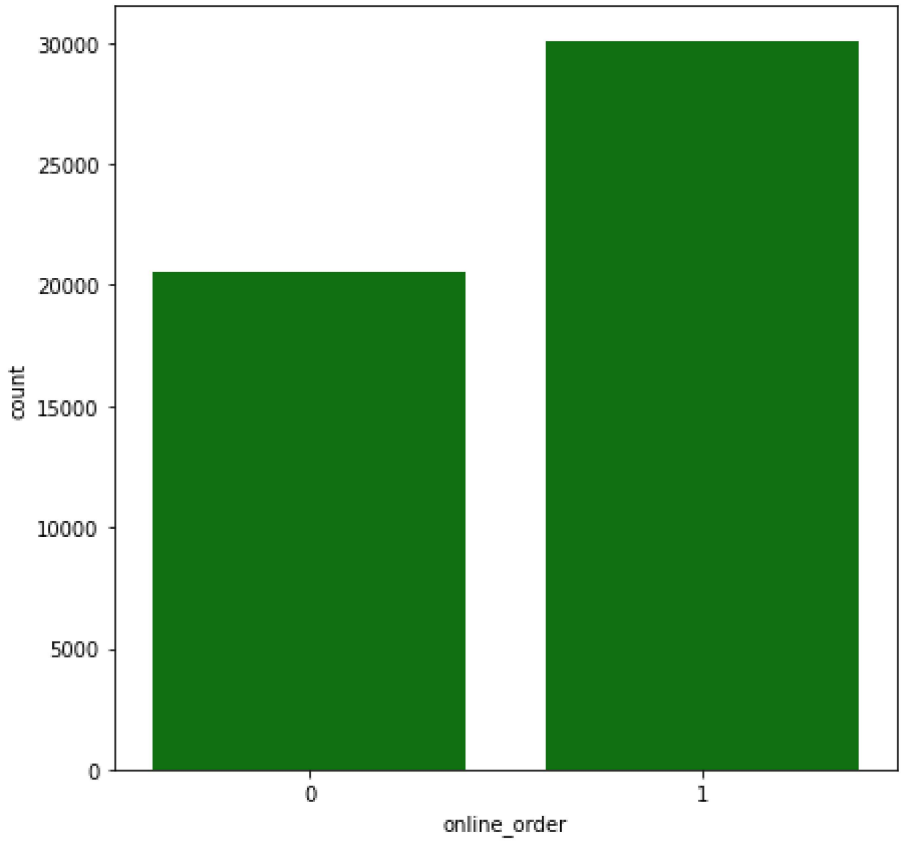
In [52]:
```
df['online_order'].replace({'Yes' : 1, 'No' : 0}, inplace = True)
```

Plots

In [53]:
```
plt.figure(figsize = (7, 7))
sns.countplot(df['online_order'], color = 'green')
```

```
c:\users\sasi\appdata\local\programs\python\python38-32\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[53]: <AxesSubplot:xlabel='online_order', ylabel='count'>



In [54]:
```
correlation = df.corr()['rate'].sort_values(ascending = False).to_frame().he
```

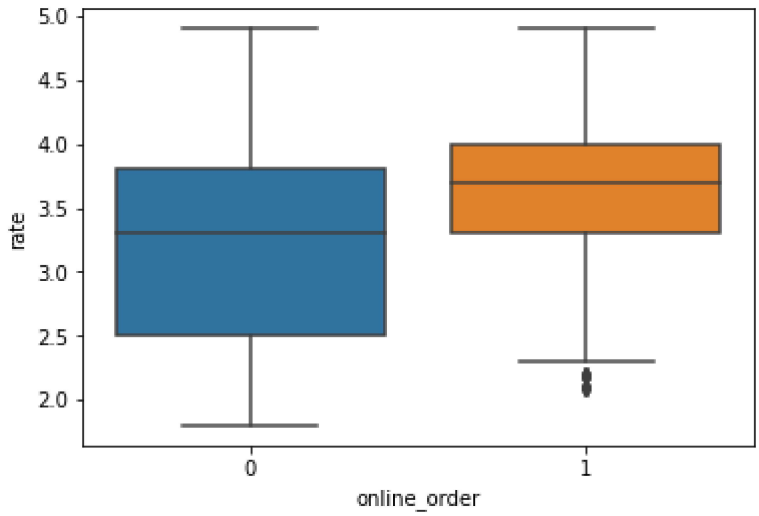
In [55]:
```
cmap = sns.light_palette("cyan", as_cmap = True)

s = correlation.style.background_gradient(cmap = cmap)

s
```

Out[55]:

	rate
rate	1.000000
votes	0.410401
online_order	0.246909

In [56]:
```
sns.boxplot(x = 'online_order', y = 'rate', data = df)


```

Out[56]: <AxesSubplot:xlabel='online_order', ylabel='rate'>



In [57]:
```
df['cuisines'].unique()
```

Out[57]:
```
array(['North Indian, Mughlai, Chinese', 'Chinese, North Indian, Thai',
       'Cafe, Mexican, Italian', ...,
       'North Indian, Street Food, Biryani', 'Chinese, Mughlai',
       'North Indian, Chinese, Arabian, Momos'], dtype=object)
```

In [58]:
```
len(df['cuisines'][4].split(','))
```

Out[58]: 2

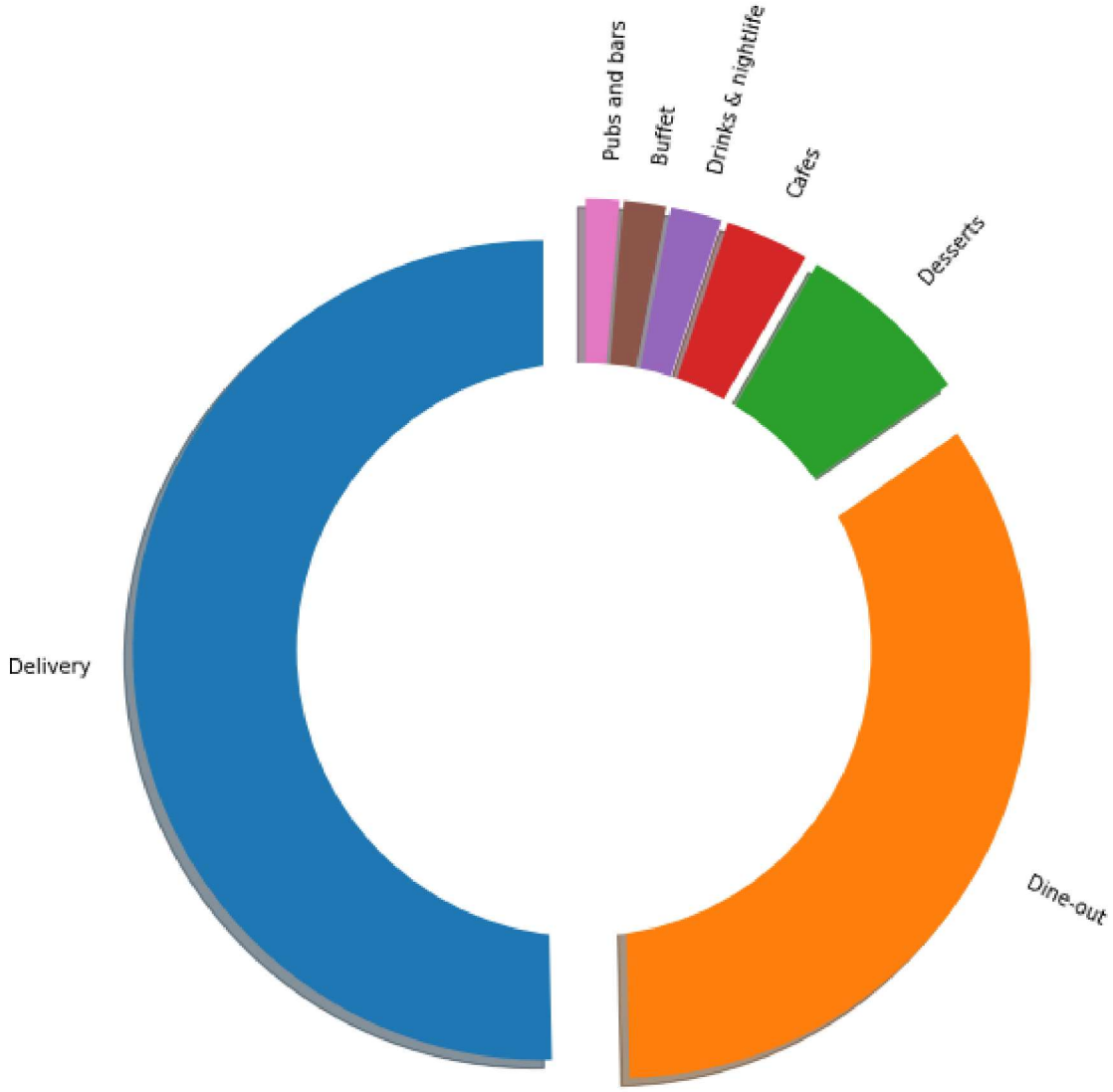
In [59]:
```
1 df['Number_of_cuisines_offered'] = df['cuisines'].apply(lambda x : len(x.spl
2
3 df.corr()
```

Out[59]:

	online_order	rate	votes	Number_of_cuisines_offered
online_order	1.000000	0.246909	0.047480	0.107209
rate	0.246909	1.000000	0.410401	0.240100
votes	0.047480	0.410401	1.000000	0.233108
Number_of_cuisines_offered	0.107209	0.240100	0.233108	1.000000

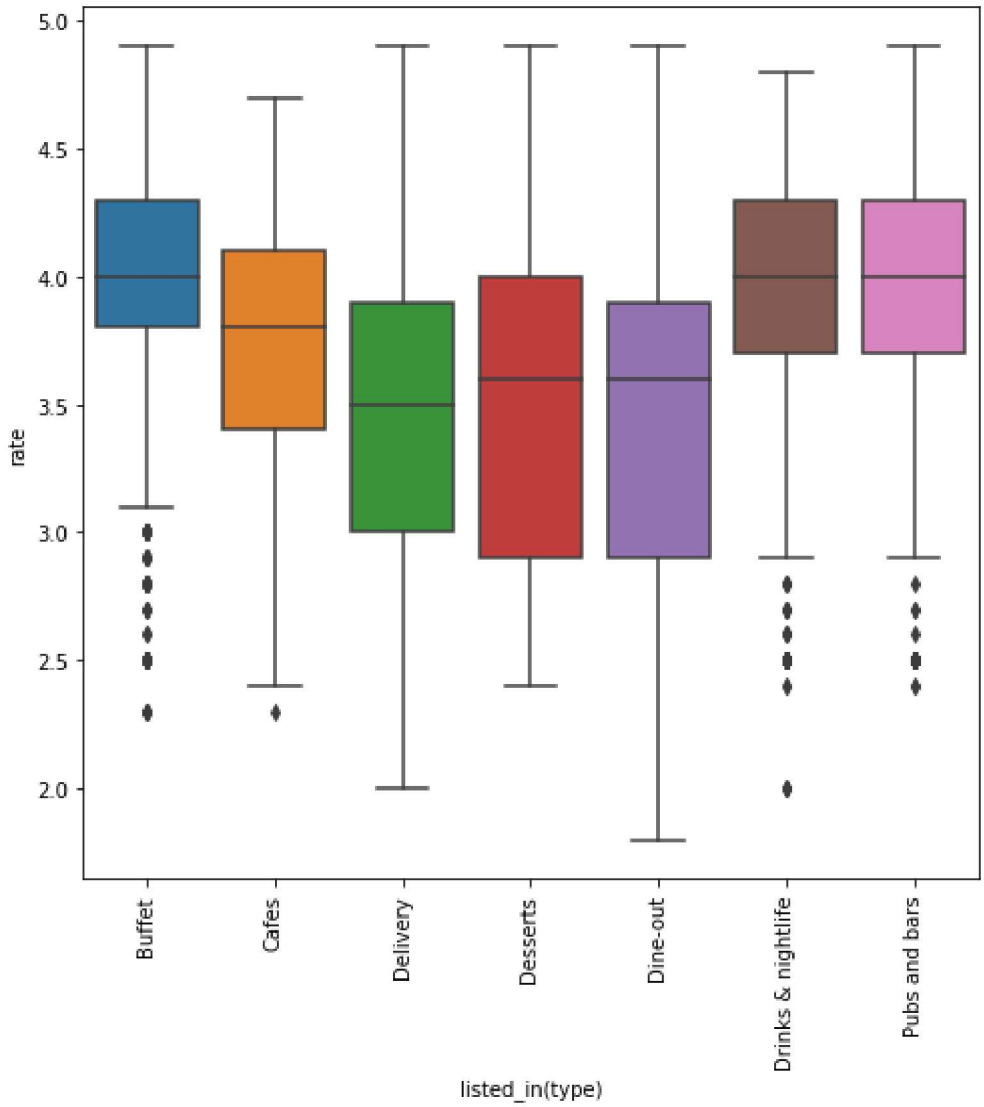
Pie chart

In [60]:
```
# Pie chart
labels = df['listed_in(type)'].value_counts().index
sizes = df['listed_in(type)'].value_counts().values
# only "explode" the 2nd slice (i.e. 'Hogs')
explode = (0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1)
fig1, ax1 = plt.subplots(figsize = (8, 8))

ax1.pie(sizes, labels = labels,
        shadow = True, startangle = 90, explode = explode, rotatelabels = Tr
centre_circle = plt.Circle((0, 0), 0.70,fc = 'white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

# Equal aspect ratio ensures that pie is drawn as a circle
ax1.axis('equal')
plt.tight_layout()
plt.show()
```



Box plot

In [61]:
```
plt.figure(figsize = (8, 8))
g = sns.boxplot(x = 'listed_in(type)', y = 'rate', data = df)
plt.xticks(rotation = 90)
plt.show()
```



In [3]:
```
import numpy as np # linear algebra
import os # accessing directory structure
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

```

In [4]:
```
df = pd.read_csv('zomato.csv', error_bad_lines = False, engine='python')
df.dataframeName = 'zomato.csv'
nRow, nCol = df.shape
print(f'There are {nRow} rows and {nCol} columns')

```

```
Skipping line 1130: field larger than field limit (131072)
Skipping line 2046: field larger than field limit (131072)
Skipping line 2588: field larger than field limit (131072)
Skipping line 4133: field larger than field limit (131072)
Skipping line 4244: field larger than field limit (131072)
Skipping line 4804: field larger than field limit (131072)
Skipping line 4806: field larger than field limit (131072)
Skipping line 4810: field larger than field limit (131072)
Skipping line 4812: field larger than field limit (131072)
Skipping line 4946: field larger than field limit (131072)
Skipping line 4947: field larger than field limit (131072)
Skipping line 5033: field larger than field limit (131072)
Skipping line 5176: field larger than field limit (131072)
Skipping line 5300: field larger than field limit (131072)
Skipping line 5328: field larger than field limit (131072)
Skipping line 5497: field larger than field limit (131072)
Skipping line 5711: field larger than field limit (131072)
Skipping line 5932: field larger than field limit (131072)
Skipping line 6048: field larger than field limit (131072)
```

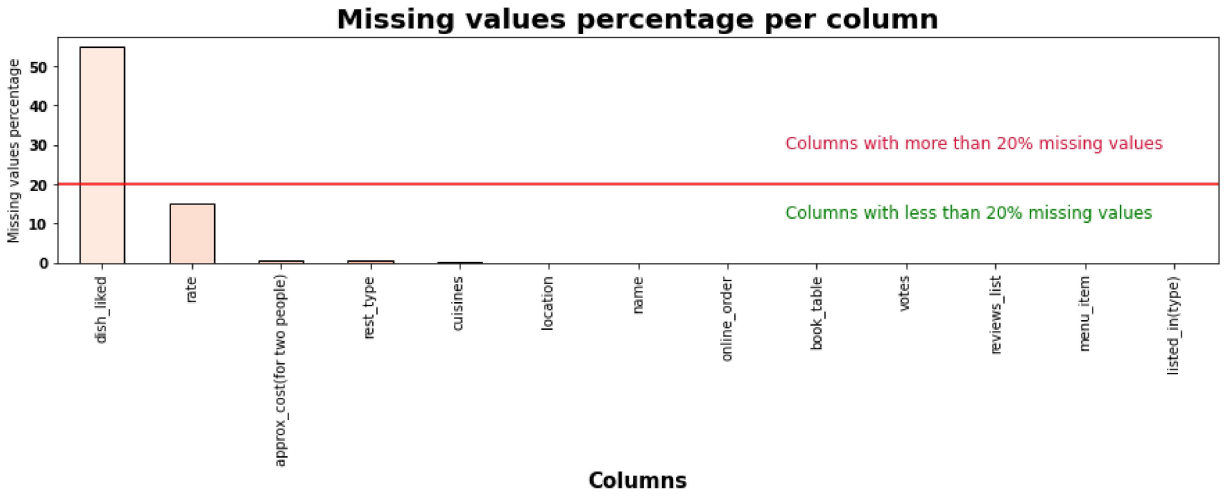
In [5]:
```
df.to_csv(r'C:\Users\sasi\AppData\Local\Programs\Python\Python38-32\zom.csv'
```

In [6]:
```
df.drop(['url', 'address', 'phone', 'listed_in(city)'], axis = 1, inplace =

df.isnull().sum()
```

Out[6]:
```
name                                0
online_order                        0
book_table                          0
rate                             7775
votes                               0
location                           21
rest_type                         224
dish_liked                      28074
cuisines                           45
approx_cost(for two people)       344
reviews_list                        0
menu_item                           0
listed_in(type)                     0
dtype: int64
```

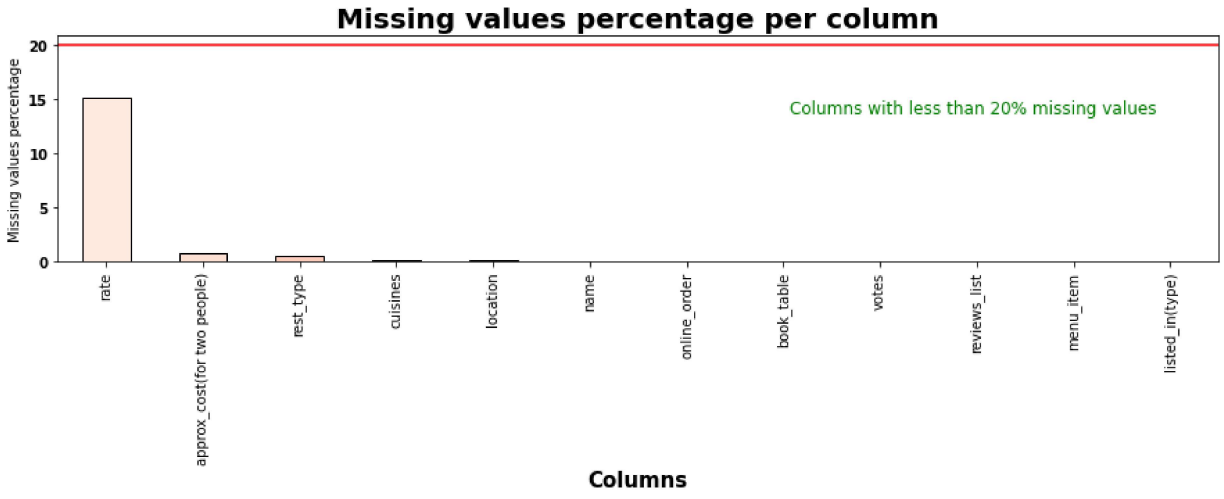
In [7]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

def missing_values(data, thresh = 20, color = 'black', edgecolor = 'black',

    plt.figure(figsize = (width, height))
    percentage = (data.isnull().mean()) * 100
    percentage.sort_values(ascending = False).plot.bar(color = color, edgeco
    plt.axhline(y = thresh, color = 'r', linestyle = '-')

    plt.title('Missing values percentage per column', fontsize = 20, weight

    plt.text(len(data.isnull().sum()/len(data))/1.7, thresh + 12.5, f'Column
         ha = 'left' ,va = 'top')
    plt.text(len(data.isnull().sum()/len(data))/1.7, thresh - 5, f'Columns w
         ha = 'left' ,va = 'top')
    plt.xlabel('Columns', size = 15, weight = 'bold')
    plt.ylabel('Missing values percentage')
    plt.yticks(weight = 'bold')

    return plt.show()
```

In [8]:
```
missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))

```

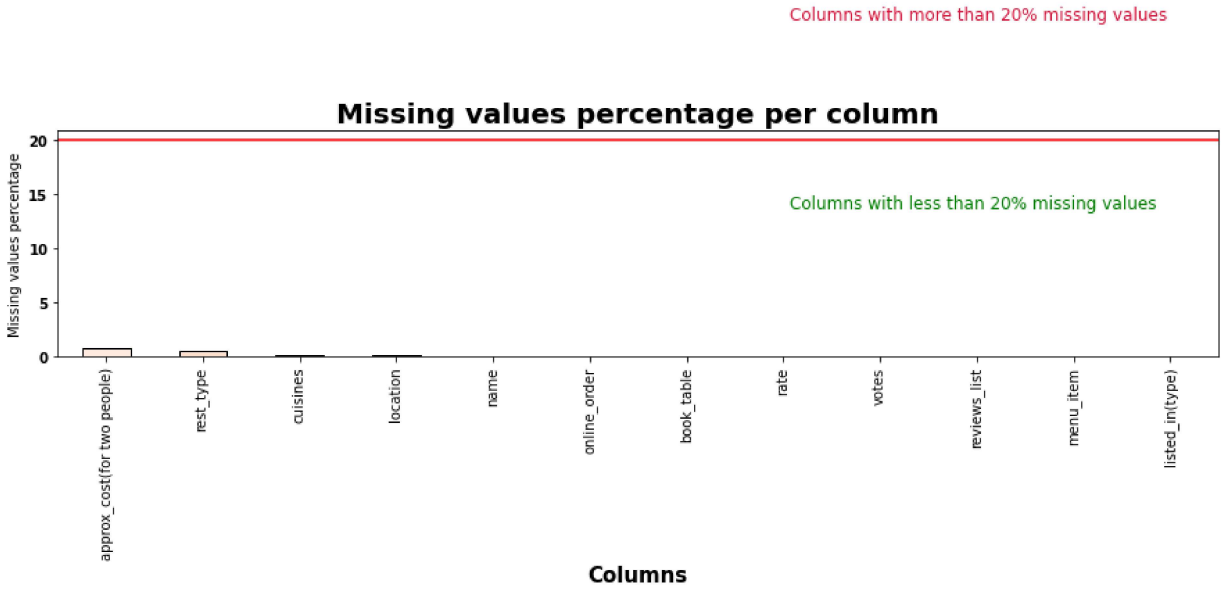


In [9]:
```
df.drop(['dish_liked'], axis = 1, inplace = True)

missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))

```

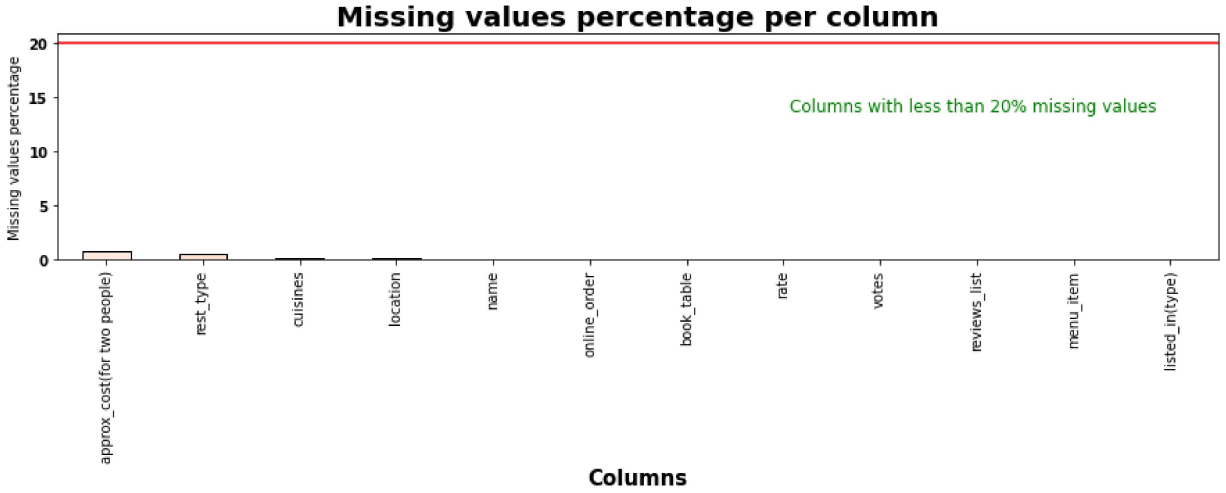
Columns with more than 20% missing values



In [10]:
```
df['rate'] = df['rate'].fillna(df['rate'].mode()[0])

missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))
```



In [11]:
```
missing_values(df, thresh = 20, color = sns.color_palette('Reds',15))


df.isnull().sum().sum()

```



Out[11]: 634

In [12]:
```
1 df.head(5)
```

Out[12]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	approx_cos two pe
0	Jalsa	Yes	Yes	4.1/5	775	Banashankari	Casual Dining	North Indian, Mughlai, Chinese	
1	Spice Elephant	Yes	No	4.1/5	787	Banashankari	Casual Dining	Chinese, North Indian, Thai	
2	San Churro Cafe	Yes	No	3.8/5	918	Banashankari	Cafe, Casual Dining	Cafe, Mexican, Italian	
3	Addhuri Udupi Bhojana	No	No	3.7/5	88	Banashankari	Quick Bites	South Indian, North Indian	
4	Grand Village	No	No	3.8/5	166	Basavanagudi	Casual Dining	North Indian, Rajasthani	

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

In [14]:
```
1 df.head(10)
```

Out[14]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	approx_ two
0	Jalsa	Yes	Yes	4.1/5	775	Banashankari	Casual Dining	North Indian, Mughlai, Chinese	
1	Spice Elephant	Yes	No	4.1/5	787	Banashankari	Casual Dining	Chinese, North Indian, Thai	
2	San Churro Cafe	Yes	No	3.8/5	918	Banashankari	Cafe, Casual Dining	Cafe, Mexican, Italian	
3	Addhuri Udupi Bhojana	No	No	3.7/5	88	Banashankari	Quick Bites	South Indian, North Indian	
4	Grand Village	No	No	3.8/5	166	Basavanagudi	Casual Dining	North Indian, Rajasthani	
5	Timepass Dinner	Yes	No	3.8/5	286	Basavanagudi	Casual Dining	North Indian	
6	Rosewood International Hotel - Bar & Restaurant	No	No	3.6/5	8	Mysore Road	Casual Dining	North Indian, South Indian, Andhra, Chinese	
7	Onesta	Yes	Yes	4.6/5	2556	Banashankari	Casual Dining, Cafe	Pizza, Cafe, Italian	
8	Penthouse Cafe	Yes	No	4.0/5	324	Banashankari	Cafe	Cafe, Italian, Continental	
9	Smacznego	Yes	No	4.2/5	504	Banashankari	Cafe	Cafe, Mexican, Italian, Momos, Beverages	

In [15]:
```
df['rate'].replace({"NEW" : "2.5 /5"}, inplace = True)

df['rate'].unique()

df.loc[df['rate'] == '-'].head(5)
```

Out[15]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	approx_cost(for two people)
3065	House of Tasty Food	No	No	-	0	Wilson Garden	Quick Bites	North Indian	
3066	Super Chef's (New Royal treat)	No	No	-	0	Bannerghatta Road	Quick Bites	North Indian, Chinese, South Indian	
3370	Right Pizza	Yes	No	-	0	Basavanagudi	Quick Bites	Pizza	
3375	Mezban Family Restaurant	Yes	No	-	0	Basavanagudi	Quick Bites	Chinese, North Indian	
3384	Mota Bawarchi	No	No	-	0	Basavanagudi	Quick Bites	North Indian, Biryani, Fast Food	

In [16]:
```
a = df.loc[df['rate'] == '-'].index
df.drop(a, axis = 0, inplace = True)

df['rate'].unique()
```

Out[16]:
```
array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
       '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
       '4.3/5', '2.5 /5', '2.9/5', '3.5/5', '2.6/5', '3.8 /5', '3.4/5',
       '4.5/5', '2.5/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5',
       '3.4 /5', '3.6 /5', '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5',
       '4.1 /5', '3.7 /5', '3.1 /5', '2.9 /5', '3.3 /5', '2.8 /5',
       '3.5 /5', '2.7 /5', '3.2 /5', '2.6 /5', '4.5 /5', '4.3 /5',
       '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '4.6 /5', '4.9 /5',
       '3.0 /5', '4.8 /5', '2.3 /5', '2.4 /5', '4.7 /5', '2.2 /5',
       '2.0 /5', '2.1 /5', '1.8 /5'], dtype=object)
```

In [17]:
```
df['rate'] = df['rate'].apply(lambda x : x.split('/')[0])

df['rate'].unique()
```

Out[17]:
```
array(['4.1', '3.8', '3.7', '3.6', '4.6', '4.0', '4.2', '3.9', '3.1',
       '3.0', '3.2', '3.3', '2.8', '4.4', '4.3', '2.5 ', '2.9', '3.5',
       '2.6', '3.8 ', '3.4', '4.5', '2.5', '2.7', '4.7', '2.4', '2.2',
       '2.3', '3.4 ', '3.6 ', '4.8', '3.9 ', '4.2 ', '4.0 ', '4.1 ',
       '3.7 ', '3.1 ', '2.9 ', '3.3 ', '2.8 ', '3.5 ', '2.7 ', '3.2 ',
       '2.6 ', '4.5 ', '4.3 ', '4.4 ', '4.9', '2.1', '2.0', '1.8', '4.6 ',
       '4.9 ', '3.0 ', '4.8 ', '2.3 ', '2.4 ', '4.7 ', '2.2 ', '2.0 ',
       '2.1 ', '1.8 '], dtype=object)
```

In [18]:
```
df['rate'] = df['rate'].apply(lambda x : x.split(' ')[0])

df['rate'].unique()
```

Out[18]:
```
array(['4.1', '3.8', '3.7', '3.6', '4.6', '4.0', '4.2', '3.9', '3.1',
       '3.0', '3.2', '3.3', '2.8', '4.4', '4.3', '2.5', '2.9', '3.5',
       '2.6', '3.4', '4.5', '2.7', '4.7', '2.4', '2.2', '2.3', '4.8',
       '4.9', '2.1', '2.0', '1.8'], dtype=object)
```

In [19]:
```
print(df['rate'].dtype)
```

```
object
```

In [20]:
```

df = df.astype({'rate' : float})
print(df['rate'].dtype)

```

```
float64
```

In [21]:
```
df.head(10)
```

Out[21]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	approx_two
0	Jalsa	Yes	Yes	4.1	775	Banashankari	Casual Dining	North Indian, Mughlai, Chinese	
1	Spice Elephant	Yes	No	4.1	787	Banashankari	Casual Dining	Chinese, North Indian, Thai	
2	San Churro Cafe	Yes	No	3.8	918	Banashankari	Cafe, Casual Dining	Cafe, Mexican, Italian	
3	Addhuri Udupi Bhojana	No	No	3.7	88	Banashankari	Quick Bites	South Indian, North Indian	
4	Grand Village	No	No	3.8	166	Basavanagudi	Casual Dining	North Indian, Rajasthani	
5	Timepass Dinner	Yes	No	3.8	286	Basavanagudi	Casual Dining	North Indian	
6	Rosewood International Hotel - Bar & Restaurant	No	No	3.6	8	Mysore Road	Casual Dining	North Indian, South Indian, Andhra, Chinese	
7	Onesta	Yes	Yes	4.6	2556	Banashankari	Casual Dining, Cafe	Pizza, Cafe, Italian	
8	Penthouse Cafe	Yes	No	4.0	324	Banashankari	Cafe	Cafe, Italian, Continental	
9	Smacznego	Yes	No	4.2	504	Banashankari	Cafe	Cafe, Mexican, Italian, Momos, Beverages	

In [22]:
```
df['online_order'].replace({'Yes' : 1, 'No' : 0}, inplace = True)
```

In [23]:
```
print(df['online_order'].dtype)
```

```
int64
```

In [24]:
```
1 df.head(10)
```

Out[24]:

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	approx_ two
0	Jalsa	1	Yes	4.1	775	Banashankari	Casual Dining	North Indian, Mughlai, Chinese	
1	Spice Elephant	1	No	4.1	787	Banashankari	Casual Dining	Chinese, North Indian, Thai	
2	San Churro Cafe	1	No	3.8	918	Banashankari	Cafe, Casual Dining	Cafe, Mexican, Italian	
3	Addhuri Udupi Bhojana	0	No	3.7	88	Banashankari	Quick Bites	South Indian, North Indian	
4	Grand Village	0	No	3.8	166	Basavanagudi	Casual Dining	North Indian, Rajasthani	
5	Timepass Dinner	1	No	3.8	286	Basavanagudi	Casual Dining	North Indian	
6	Rosewood International Hotel - Bar & Restaurant	0	No	3.6	8	Mysore Road	Casual Dining	North Indian, South Indian, Andhra, Chinese	
7	Onesta	1	Yes	4.6	2556	Banashankari	Casual Dining, Cafe	Pizza, Cafe, Italian	
8	Penthouse Cafe	1	No	4.0	324	Banashankari	Cafe	Cafe, Italian, Continental	
9	Smacznego	1	No	4.2	504	Banashankari	Cafe	Cafe, Mexican, Italian, Momos, Beverages	

In [25]:
```
1 df.to_csv(r'C:\Users\sasi\AppData\Local\Programs\Python\Python38-32\zom.csv'
```

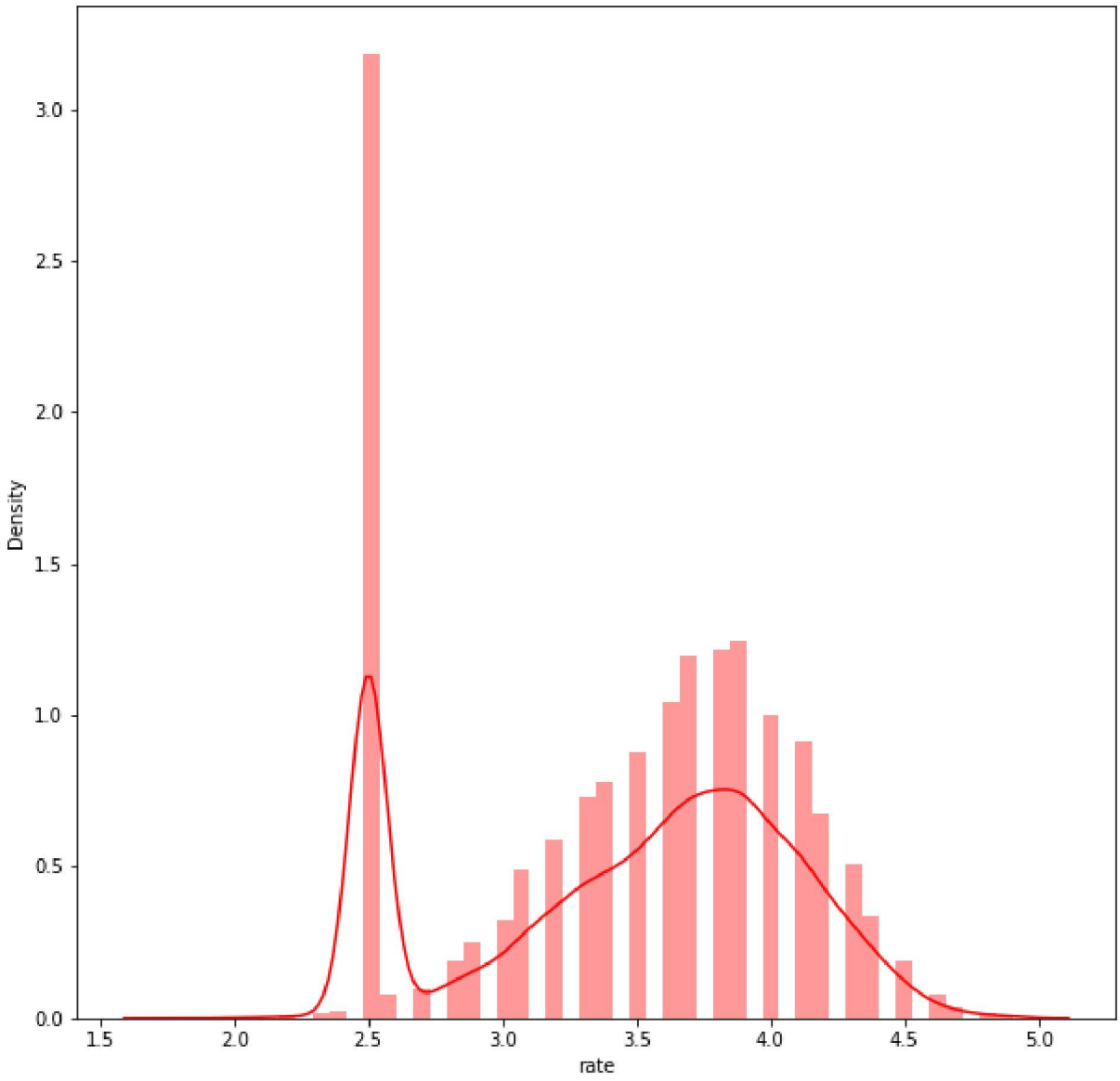
In [26]:
```
1 df = pd.read_csv('zom.csv', error_bad_lines = False, engine='python')
2 df.dataframeName = 'zom.csv'
3 nRow, nCol = df.shape
4 print(f'There are {nRow} rows and {nCol} columns')
```

```
There are 51086 rows and 11 columns
```

In [27]:
```
plt.figure(figsize = (10, 10))
sns.distplot(df['rate'], color = 'red')

```

```
c:\users\sasi\appdata\local\programs\python\python38-32\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

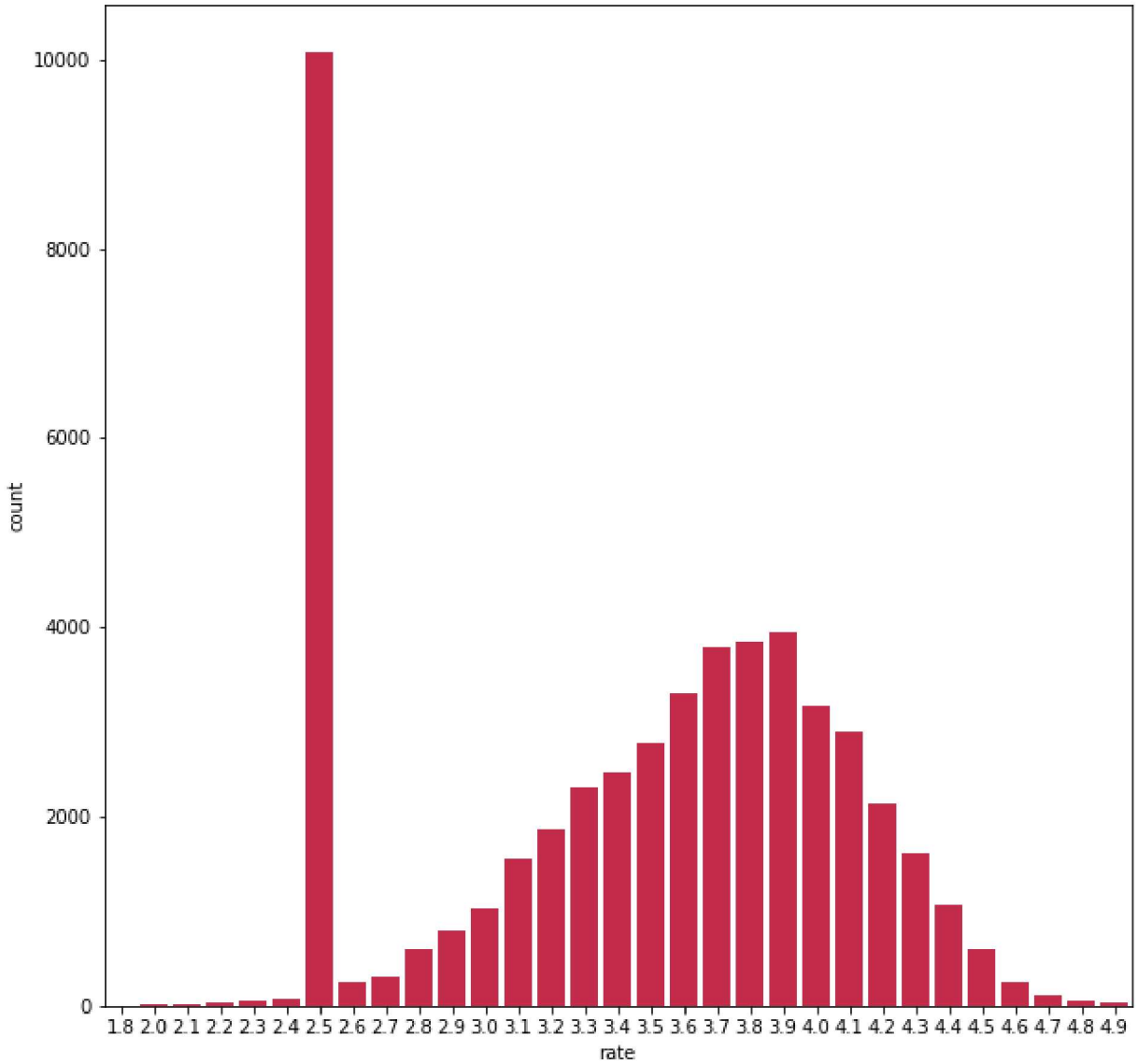
Out[27]: <AxesSubplot:xlabel='rate', ylabel='Density'>



In [28]:
```
plt.figure(figsize = (10, 10))
sns.countplot(df['rate'], color = 'crimson')

```

```
c:\users\sasi\appdata\local\programs\python\python38-32\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[28]: <AxesSubplot:xlabel='rate', ylabel='count'>

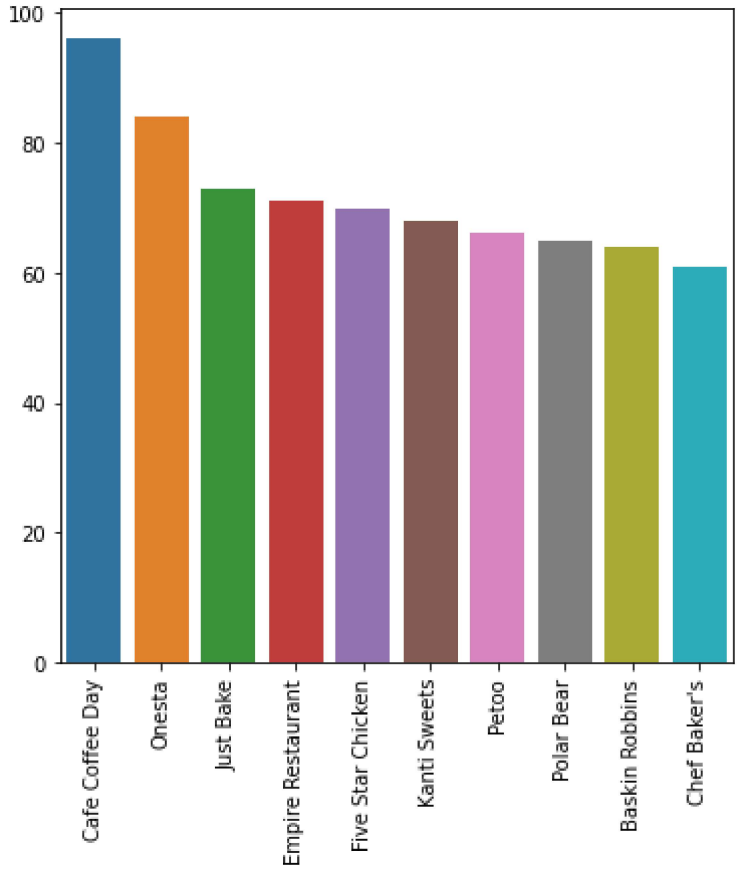


In [29]:
```
print("Total number of restaurants in Bengaluru are: ", len(df['name'].uniqu
```

```
Total number of restaurants in Bengaluru are:  8780
```

In [30]:
```
plt.figure(figsize = (6, 6))
df['name'].value_counts().head(10)
index = df['name'].value_counts().head(10).index
values = df['name'].value_counts().head(10).values
g = sns.barplot(x = index, y = values, data = df)
g.set_xticklabels(labels = index, rotation = 90)
g
```

Out[30]: <AxesSubplot:>



In [31]:
```
1 df.groupby('name')['rate'].mean().sort_values(ascending = False).head(15)
```

Out[31]:
```
name
SantÃ¤Â¤Ã¤Â¤Ã¤Â¤Ã
```

In [32]:
```
1 df.groupby('name')['votes', 'rate'].max().sort_values(ascending = False, by
2
```

```
<ipython-input-32-45c3b3c43393>:1: FutureWarning: Indexing with multiple keys
(implicitly converted to a tuple of keys) will be deprecated, use a list instea
d.
  df.groupby('name')['votes', 'rate'].max().sort_values(ascending = False, by =
'votes').head(15)
```

Out[32]:

name	votes	rate
Byg Brewski Brewing Company	16832	4.9
Toit	14956	4.7
Truffles	14723	4.7
AB's - Absolute Barbecues	12121	4.9
The Black Pearl	10547	4.8
Onesta	9085	4.6
Big Pitcher	9041	4.7
Arbor Brewing Company	8414	4.5
Empire Restaurant	8304	4.4
Prost Brew Pub	7870	4.5
Church Street Social	7584	4.3
Hoot	7330	4.2
Barbeque Nation	7270	4.8
Meghana Foods	7238	4.5
Flechazo	7154	4.9

In [33]:
```
names = df.groupby('name')['rate'].mean().sort_values(ascending = False).hea
locations = []
for i in names:
    loc = []
    locations.append(df.loc[df['name'] == i]['location'].unique().tolist())
name_location = dict(zip(names, locations))
name_location
```

Out[33]:
```
{'SantÃ\x83Â\x83Ã\x82Â\x83Ã\x83Â\x82Ã\x82Â\x83Ã\x83Â\x83Ã\x82Â\x82Ã\x83Â\x82Ã\x
82Â\x83Ã\x83Â\x83Ã\x82Â\x83Ã\x83Â\x82Ã\x82Â\x82Ã\x83Â\x83Ã\x82Â\x82Ã\x83Â\x82Ã
\x82Â© Spa Cuisine': ['Indiranagar'],
 'Byg Brewski Brewing Company': ['Sarjapur Road'],
 'Asia Kitchen By Mainland China': ['Koramangala 5th Block'],
 'Punjab Grill': ['Whitefield', 'Malleshwaram'],
 'Flechazo': ['Whitefield', 'Marathahalli'],
 'Belgian Waffle Factory': ['Brigade Road',
  'Koramangala 5th Block',
  'Vasanth Nagar',
  'Kalyan Nagar'],
 'The Pizza Bakery': ['Indiranagar'],
 'O.G. Variar & Sons': ['Rajajinagar'],
 'CTR': ['Malleshwaram'],
 'Barbecue by Punjab Grill': ['Malleshwaram'],
 'House Of Commons': ['Koramangala 5th Block', 'HSR'],
 'The Black Pearl': ['Marathahalli', 'Koramangala 5th Block'],
 'Maziga': ['Whitefield'],
 'ECHOES Koramangala': ['Koramangala 5th Block'],
 'Toit': ['Indiranagar']}
```

In [34]:
```
1 df.groupby('location')['votes', 'rate'].max().sort_values(ascending = False,
```

```
<ipython-input-34-1c99830e0817>:1: FutureWarning: Indexing with multiple keys
(implicitly converted to a tuple of keys) will be deprecated, use a list instea
d.
  df.groupby('location')['votes', 'rate'].max().sort_values(ascending = False,
by = 'votes').head(15)
```

Out[34]:

location	votes	rate
Sarjapur Road	16832	4.9
Indiranagar	14956	4.9
Koramangala 5th Block	14723	4.9
Marathahalli	12121	4.8
Koramangala 4th Block	9085	4.6
Old Airport Road	9041	4.7
Brigade Road	8414	4.9
Church Street	7584	4.6
Whitefield	6876	4.9
HSR	6745	4.7
BTM	6490	4.9
Malleshwaram	6470	4.9
Koramangala 7th Block	6385	4.7
St. Marks Road	5294	4.6
Jayanagar	5060	4.6

In [36]:
```
1 df.groupby('location')['votes'].max().sort_values(ascending = False).head(15
```

Out[36]:
```
location
Sarjapur Road            16832
Indiranagar              14956
Koramangala 5th Block    14723
Marathahalli             12121
Koramangala 4th Block     9085
Old Airport Road          9041
Brigade Road              8414
Church Street             7584
Whitefield                6876
HSR                       6745
BTM                       6490
Malleshwaram              6470
Koramangala 7th Block     6385
St. Marks Road            5294
Jayanagar                 5060
Name: votes, dtype: int64
```

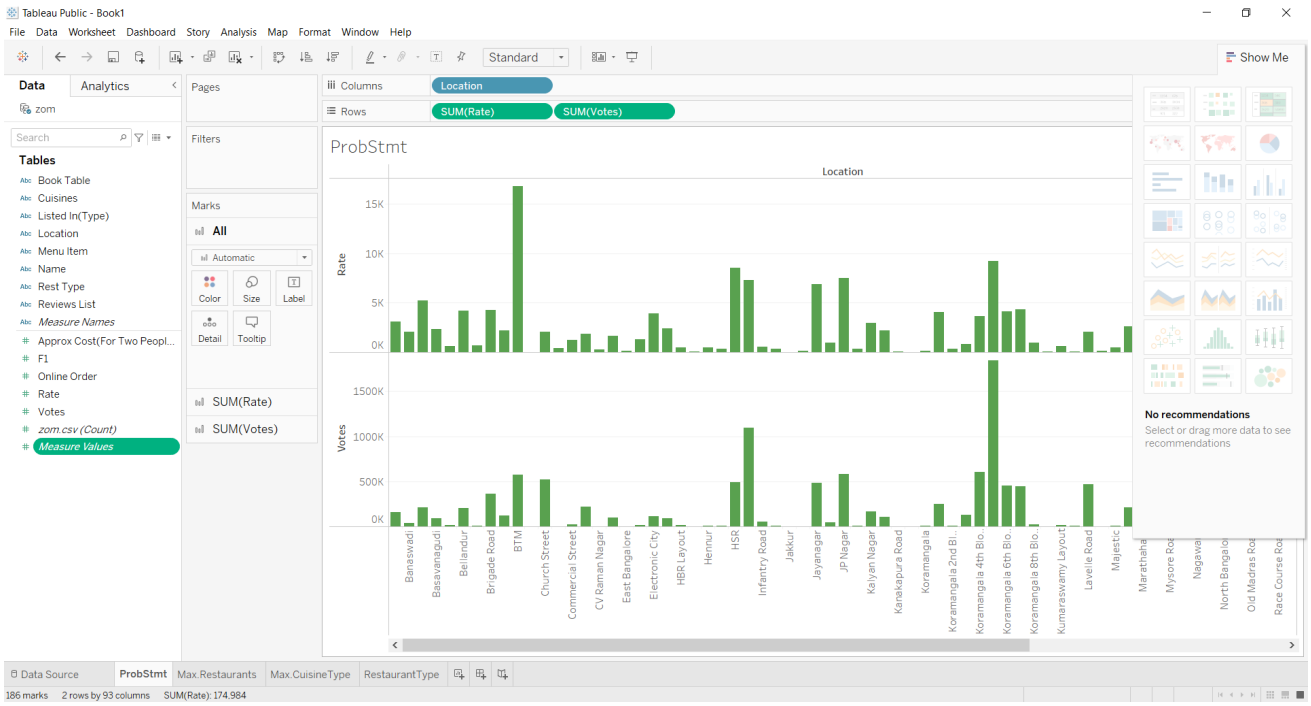
In [37]:
```
1 df.groupby('location')['rate'].max().sort_values(ascending = False).head(15)
```

Out[37]:
```
location
BTM                      4.9
Sarjapur Road            4.9
Whitefield               4.9
Koramangala 5th Block    4.9
Indiranagar              4.9
Malleshwaram             4.9
Brigade Road             4.9
JP Nagar                 4.8
Marathahalli             4.8
Rajajinagar              4.8
Kalyan Nagar             4.8
Basavanagudi             4.8
Vasanth Nagar            4.8
HSR                      4.7
Koramangala 7th Block    4.7
Name: rate, dtype: float64
```

In [ ]:
```
1
```

TABLEAU

Graph for location



Box plot:

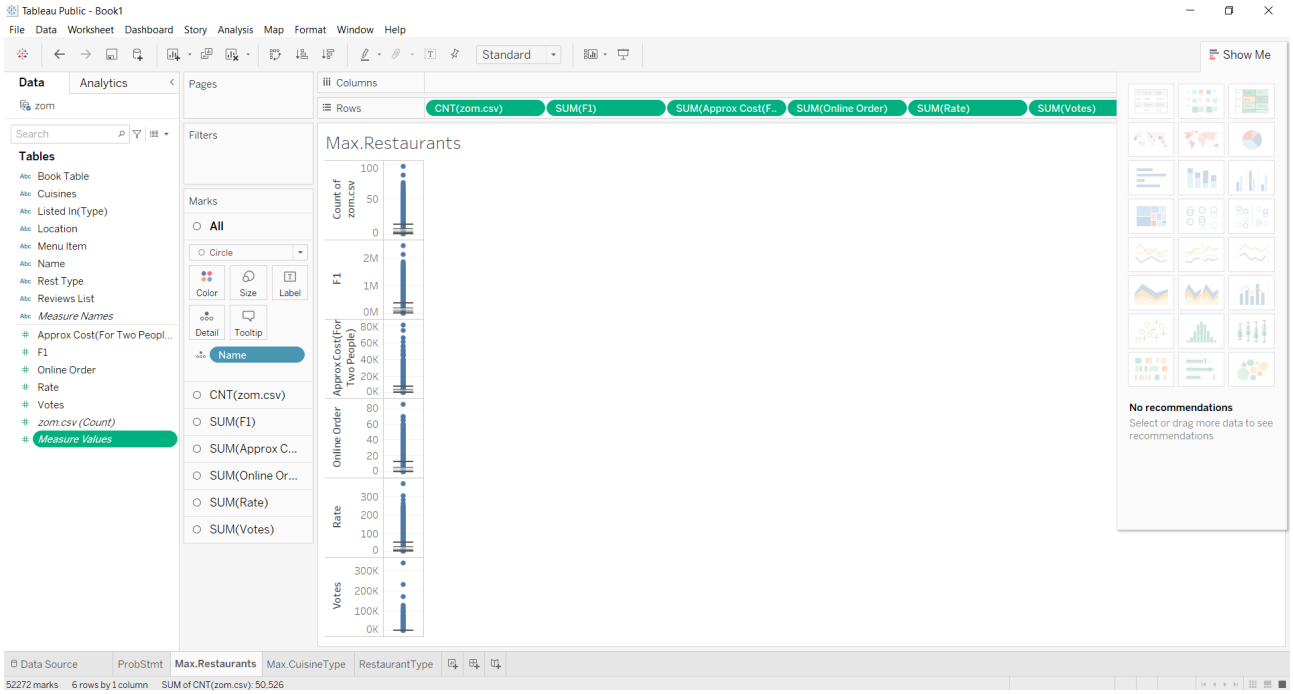
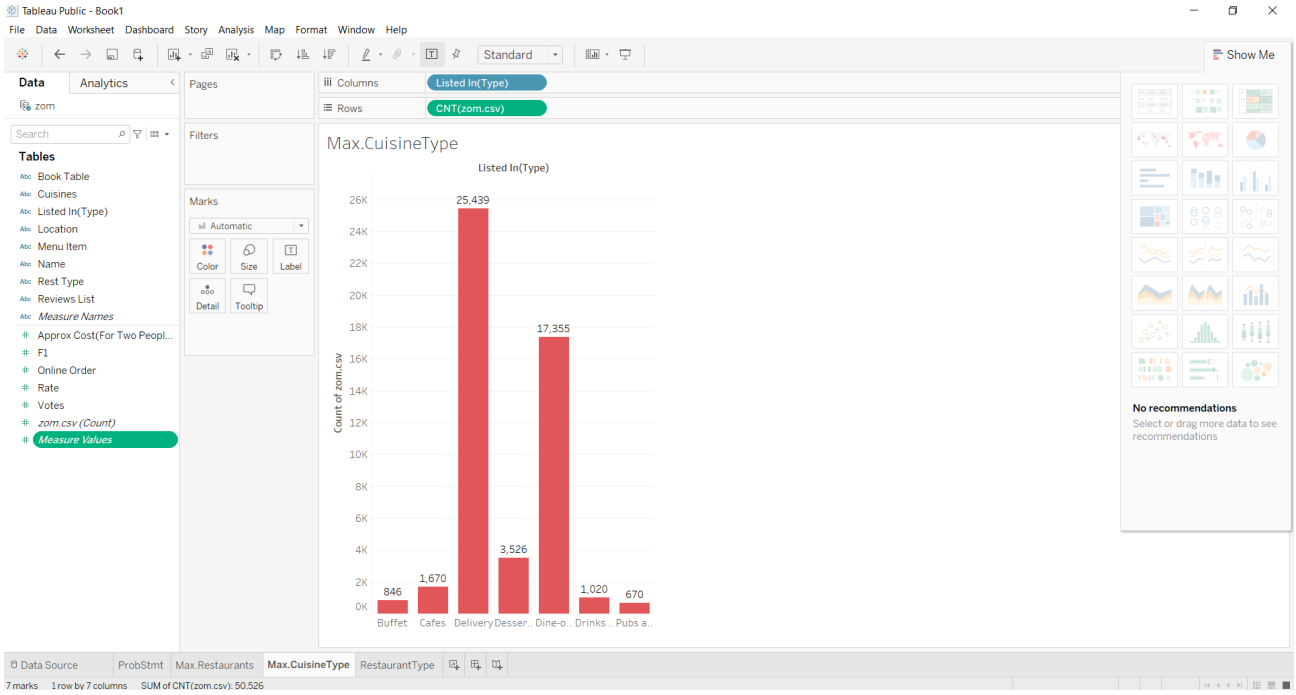


Tableau Public - Book1
File Data Worksheet Dashboard Story Analysis Map Format Window Help
Standard
Show Me
Data
Analytics
zom
Search
Tables
Book Table
Cuisines
Listed In(Type)
Location
Menu Item
Name
Rest Type
Reviews List
Measure Names
Approx Cost(For Two Peopl...
F1
Online Order
Rate
Votes
zom.csv (Count)
Measure Values
Pages
Filters
Marks
All
Circle
Color
Size
Label
Detail
Tooltip
Name
CNT(zom.csv)
SUM(F1)
SUM(Approx C...
SUM(Online Or...
SUM(Rate)
SUM(Votes)
Columns
Rows
CNT(zom.csv)
SUM(F1)
SUM(Approx Cost(F..
SUM(Online Order)
SUM(Rate)
SUM(Votes)
Max.Restaurants
Count of zom.csv
F1
Approx Cost(For Two People)
Online Order
Rate
Votes
No recommendations
Select or drag more data to see recommendations
Data Source
ProbStmt
Max.Restaurants
Max.CuisineType
RestaurantType
52272 marks
6 rows by 1 column
SUM of CNT(zom.csv): 50,526

LINEAR GRAPH:



TREE MAPS:

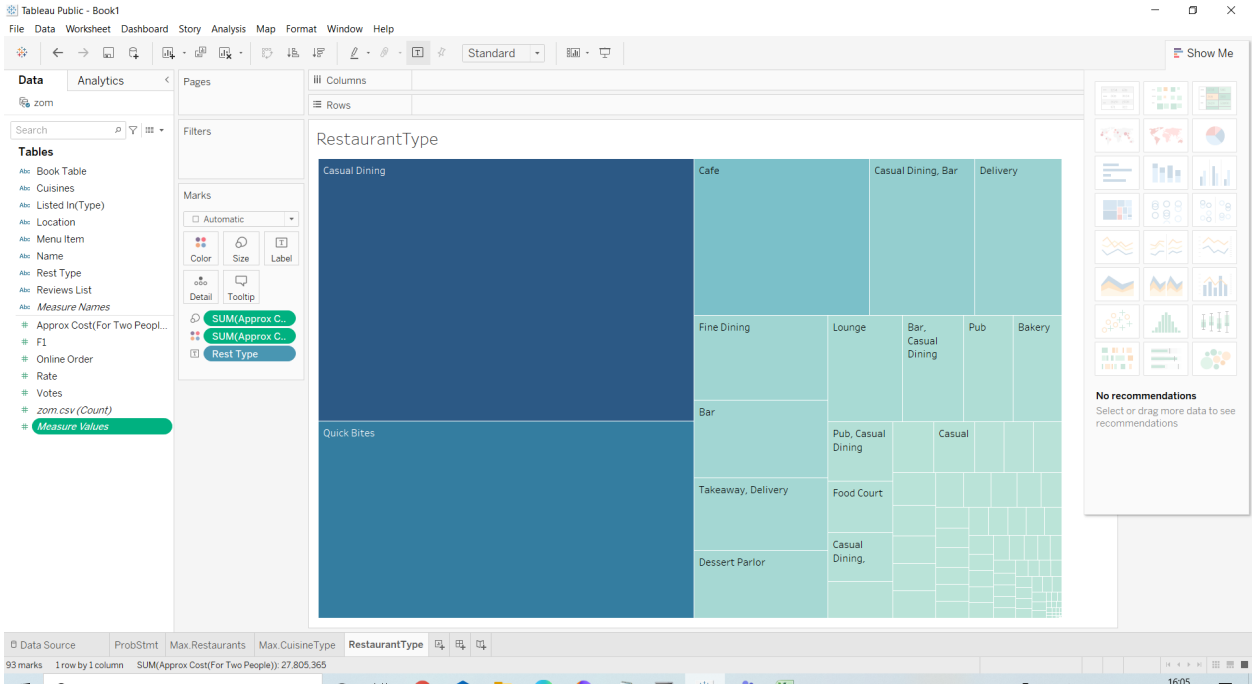


Tableau Public - Book1
File Data Worksheet Dashboard Story Analysis Map Format Window Help
Standard
Show Me
Data
Analytics
zom
Search
Tables
Book Table
Cuisines
Listed In(Type)
Location
Menu Item
Name
Rest Type
Reviews List
Measure Names
Approx Cost(For Two Peopl...
F1
Online Order
Rate
Votes
zom.csv (Count)
Measure Values
Pages
Filters
Marks
Automatic
Color
Size
Label
Detail
Tooltip
SUM(Approx C..
SUM(Approx C..
Rest Type
Columns
Rows
RestaurantType
Casual Dining
Cafe
Casual Dining, Bar
Delivery
Fine Dining
Lounge
Bar, Casual Dining
Pub
Bakery
Bar
Quick Bites
Pub, Casual Dining
Casual
Takeaway, Delivery
Food Court
Casual Dining,
Dessert Parlor
No recommendations
Select or drag more data to see recommendations
Data Source
ProbStmt
Max.Restaurants
Max.CuisineType
RestaurantType
93 marks
1 row by 1 column
SUM(Approx Cost(For Two People)): 27,805,365