```
import pandas as pd
import numpy as np
import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt
```

```
df=pd.read_csv("/content/Indian-Resturants (2).csv")
```

```
df
```

	res_id	name	establishment	url	address	city	city_id	locality	latitude
0	3400299	Bikanervala	['Quick Bites']	https://www.zomato.com/agra/bikanervala-khanda...	Kalyani Point, Near Tulsi Cinema, Bypass Road,...	Agra	34	Khandari	27.211450
1	3400005	Mama Chicken Mama Franky House	['Quick Bites']	https://www.zomato.com/agra/mama-chicken-mama-...	Main Market, Sadar Bazaar, Agra Cantt, Agra	Agra	34	Agra Cantt	27.160569
2	3401013	Bhagat Halwai	['Quick Bites']	https://www.zomato.com/agra/bhagat-halwai-2-sh...	62/1, Near Easy Day, West Shivaji Nagar, Goalp...	Agra	34	Shahganj	27.182938
3	3400290	Bhagat Halwai	['Quick Bites']	https://www.zomato.com/agra/bhagat-halwai-civi...	Near Anjana Cinema, Nehru Nagar, Civil Lines, ...	Agra	34	Civil Lines	27.205668
4	3401744	The Salt Cafe Kitchen & Bar	['Casual Dining']	https://www.zomato.com/agra/the-salt-cafe-kitc...	1C,3rd Floor, Fatehabad Road, Tajganj, Agra	Agra	34	Tajganj	27.157709
...	...	...	...	...	...	...	...	...	...
211939	3202251	Kali Mirch Cafe And Restaurant	['Casual Dining']	https://www.zomato.com/vadodara/kali-mirch-caf...	Manu Smriti Complex, Near Navrachna School, GI...	Vadodara	32	Fatehgunj	22.336931
211940	3200996	Raju Omlet	['Quick Bites']	https://www.zomato.com/vadodara/raju-omlet-kar...	Mahalaxmi Apartment, Opposite B O B, Karoli Ba...	Vadodara	32	Karelibaug	22.322455
211941	18984164	The Grand Thakar	['Casual Dining']	https://www.zomato.com/vadodara/the-grand-thak...	3rd Floor, Shreem Shalini Mall, Opposite Conqu...	Vadodara	32	Alkapuri	22.310563
211942	3201138	Subway	['Quick Bites']	https://www.zomato.com/vadodara/subway-1-akota...	G-2, Vedant Platina, Near Cosmos, Akota, Vadodara	Vadodara	32	Akota	22.270027
211943	18879846	Freshco's - The Health Cafe	['Café']	https://www.zomato.com/vadodara/freshcos-the-h...	Shop 7, Ground Floor, Opposite Natubhai Circle...	Vadodara	32	Vadiwadi	22.309935

211944 rows × 26 columns

1. Explore the basic characteristics of the dataset, including dimensions, data types, and missing values.

```
df.duplicated().sum()
```

```
np.int64(151527)
```

```
df.drop_duplicates().inplace=True
```

here i show the dimension of the dataframe of df

```
df.shape
```

```
(211944, 26)
```

here i show datatypes of each columns

```
df.dtypes
```

	0
res_id	int64
name	object
establishment	object
url	object
address	object
city	object
city_id	int64
locality	object
latitude	float64
longitude	float64
zipcode	object
country_id	int64
locality_verbose	object
cuisines	object
timings	object
average_cost_for_two	int64
price_range	int64
currency	object
highlights	object
aggregate_rating	float64
rating_text	object
votes	int64
photo_count	int64
opentable_support	float64
delivery	int64
takeaway	int64

dtype: object

```
df.describe()
```

	res_id	city_id	latitude	longitude	country_id	average_cost_for_two	price_range	aggregate_rating	2
count	2.119440e+05	211944.000000	211944.000000	211944.000000	211944.0	211944.000000	211944.000000	211944.000000	2
mean	1.349411e+07	4746.785434	21.499758	77.615276	1.0	595.812229	1.882535	3.395937	
std	7.883722e+06	5568.766386	22.781331	7.500104	0.0	606.239363	0.892989	1.283642	
min	5.000000e+01	1.000000	0.000000	0.000000	1.0	0.000000	1.000000	0.000000	
25%	3.301027e+06	11.000000	15.496071	74.877961	1.0	250.000000	1.000000	3.300000	
50%	1.869573e+07	34.000000	22.514494	77.425971	1.0	400.000000	2.000000	3.800000	
75%	1.881297e+07	11306.000000	26.841667	80.219323	1.0	700.000000	2.000000	4.100000	
max	1.915979e+07	11354.000000	10000.000000	91.832769	1.0	30000.000000	4.000000	4.900000	4

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 211944 entries, 0 to 211943
Data columns (total 26 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   res_id                211944 non-null  int64
 1   name                  211944 non-null  object
 2   establishment         211944 non-null  object
 3   url                   211944 non-null  object
 4   address               211810 non-null  object
 5   city                  211944 non-null  object
 6   city_id               211944 non-null  int64
 7   locality              211944 non-null  object
 8   latitude              211944 non-null  float64
 9   longitude             211944 non-null  float64
 10  zipcode               48757 non-null   object
 11  country_id            211944 non-null  int64
 12  locality_verbose      211944 non-null  object
 13  cuisines              210553 non-null  object
 14  timings               208070 non-null  object
 15  average_cost_for_two  211944 non-null  int64
 16  price_range           211944 non-null  int64
 17  currency              211944 non-null  object
 18  highlights            211944 non-null  object
 19  aggregate_rating      211944 non-null  float64
 20  rating_text           211944 non-null  object
 21  votes                 211944 non-null  int64
 22  photo_count           211944 non-null  int64
 23  opentable_support     211896 non-null  float64
 24  delivery              211944 non-null  int64
 25  takeaway              211944 non-null  int64
dtypes: float64(4), int64(9), object(13)
memory usage: 42.0+ MB
```

here i calacuated the total percantage of null values in each column contain

```
null=(df.isnull().sum()/df.shape[0])*100
a=null[null.round(2)>0]
a.round(1)
```

	0
address	0.1
zipcode	77.0
cuisines	0.7
timings	1.8
opentable_support	0.0

dtype: float64

Total number of null values in dataset according to the columns

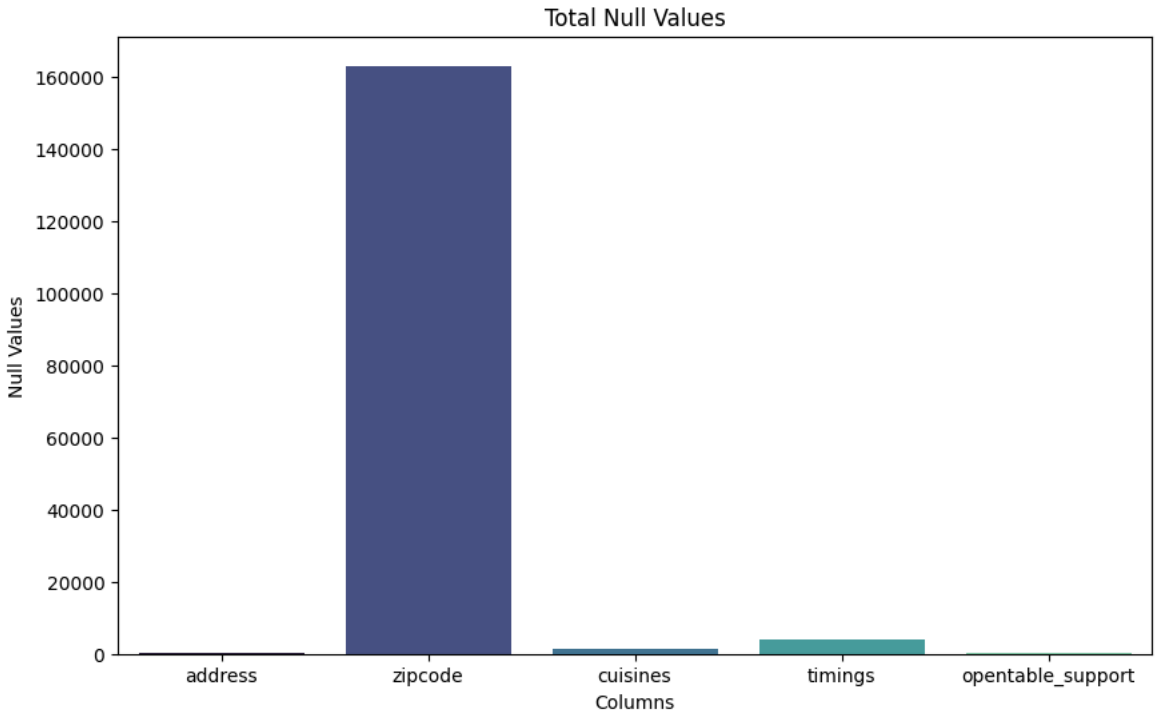
```
b=df.isnull().sum()
leat=b[b>0]
leat
```

	0
address	134
zipcode	163187
cuisines	1391
timings	3874
opentable_support	48

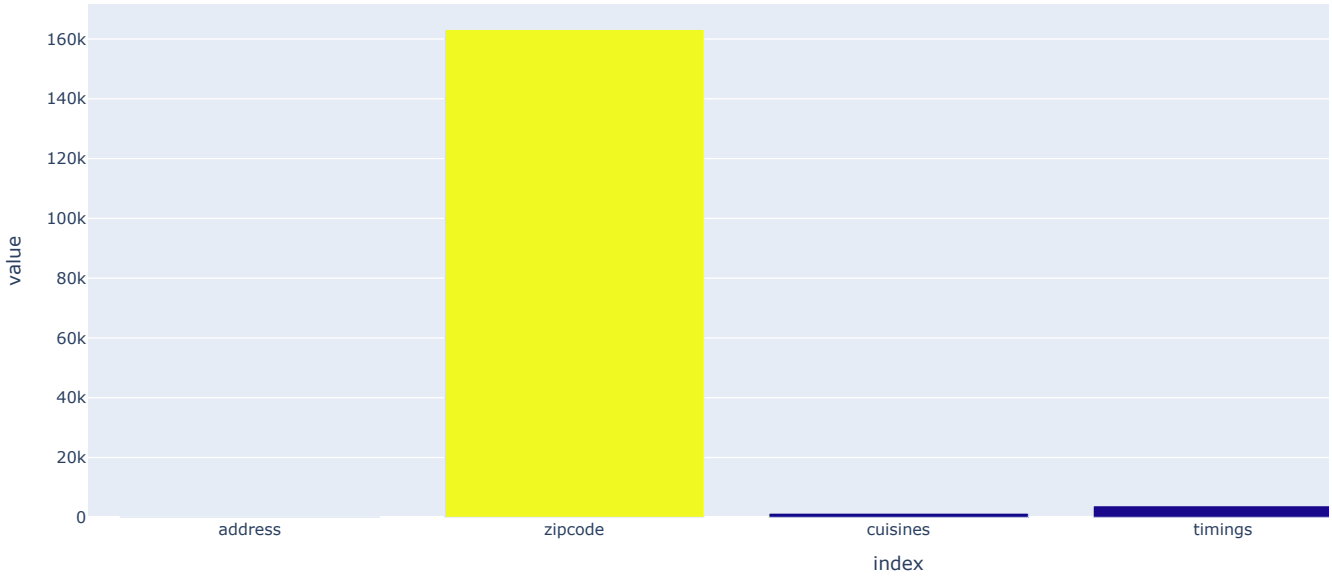
dtype: int64

```
plt.figure(figsize=(10,6))
sns.barplot(data=leat,palette="mako")
plt.title("Total Null Values")
plt.xlabel("Columns")
plt.ylabel("Null Values")
plt.grid(False)
plt.show()
```

```
/tmp/ipython-input-1219372354.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(data=leat,palette="mako")
```

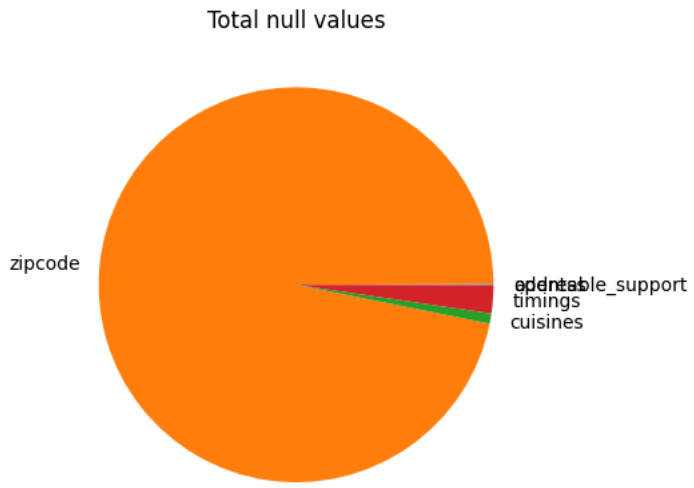



```
fig=px.bar(leat,color=leat)
fig.update_traces(textposition='outside')
fig.show()
```




```
valu=["address","zipcode","cuisines","timings","opentable_support"]
leat.plot(kind="pie",)
plt.title("Total null values")
```

```
Text(0.5, 1.0, 'Total null values')
```



here i deleted the uncessory columns one column had more then 75% of null value and one columns is not that important to the data

```
df.drop(columns=["zipcode","address"],axis=1,inplace=True)
```

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

	0
res_id	0
name	0
establishment	0
url	0
city	0
city_id	0
locality	0
latitude	0
longitude	0
country_id	0
locality_verbose	0
cuisines	1391
timings	3874
average_cost_for_two	0
price_range	0
currency	0
highlights	0
aggregate_rating	0
rating_text	0
votes	0
photo_count	0
opentable_support	48
delivery	0
takeaway	0

dtype: int64

here i filter out the data by filling them

```
cuisines_mode = df["cuisines"].mode()[0]
```

```
df["cuisines"]=df["cuisines"].fillna(cuisines_mode)
```

```
df["opentable_support"] = df["opentable_support"].fillna(1)
```

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

	0
res_id	0
name	0
establishment	0
url	0
city	0
city_id	0
locality	0
latitude	0
longitude	0
country_id	0
locality_verbose	0
cuisines	0
timings	3874
average_cost_for_two	0
price_range	0
currency	0
highlights	0
aggregate_rating	0
rating_text	0
votes	0
photo_count	0
opentable_support	0
delivery	0
takeaway	0

dtype: int64

2. Calculate and visualize the average rating of restaurants.Analyze the distribution of restaurant ratings to understand the overall rating landscape.

Average ratings with value 0

```
avg=df["aggregate_rating"].mean()
print("total avregae rating of resturants is :", avg.round(2))
```

```
total avregae rating of resturants is : 3.4
```

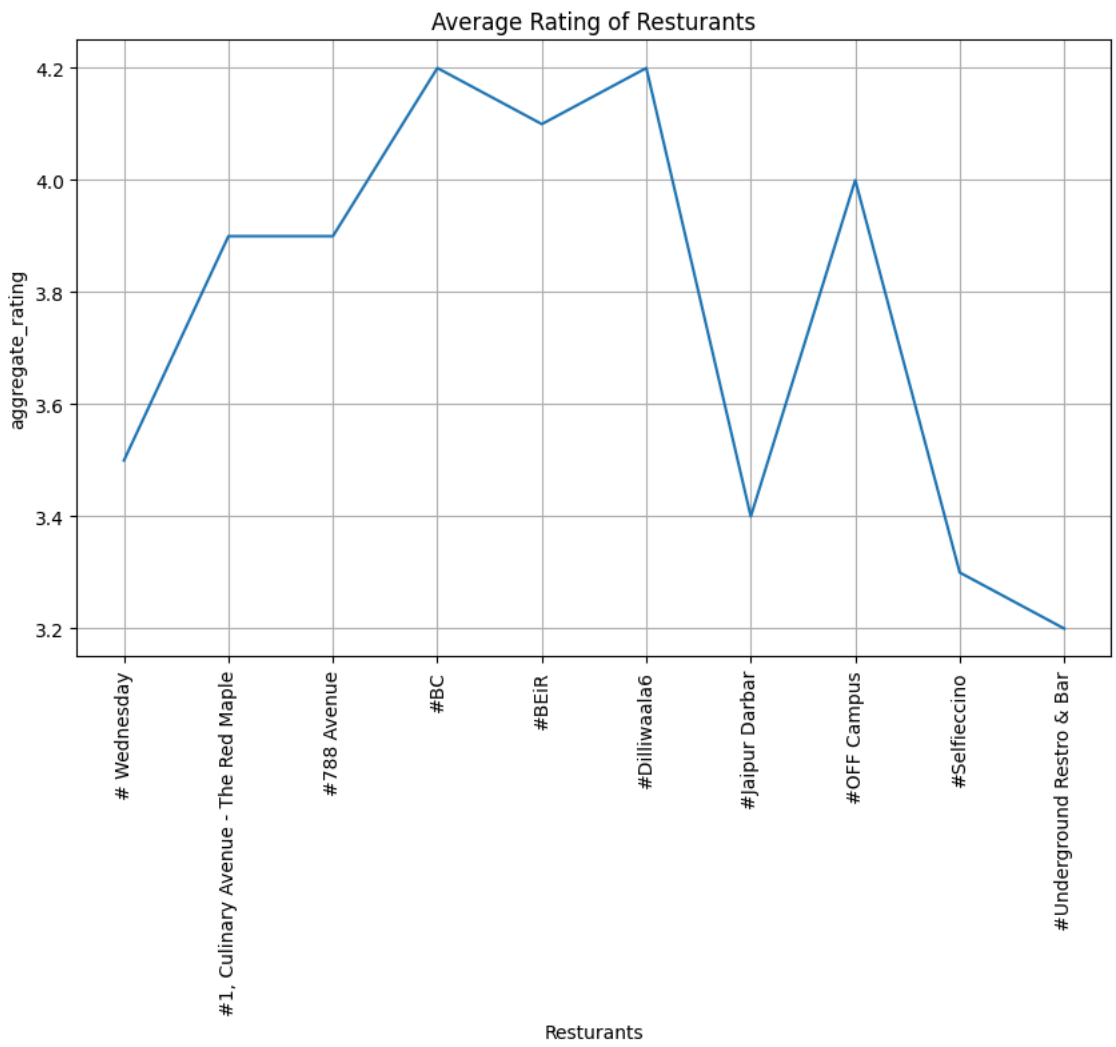
```
a=df.groupby("name")["aggregate_rating"].mean()
avg_rating=a[a>0]
avg_rating
```

name	aggregate_rating
# Wednesday	3.5
#1, Culinary Avenue - The Red Maple	3.9
#788 Avenue	3.9
#BC	4.2
#BEiR	4.1
...	...
Food Street - Veg	2.9
ट 4 Tasty	3.7
द Vege टेबल	4.2
स्पेस Bar	4.3
ह-tea The Tea Hut	4.2

33116 rows × 1 columns

dtype: float64

```
plt.figure(figsize=(10,6))
sns.lineplot(data=avg_rating.head(10))
plt.grid(True)
plt.xticks(rotation=90)
plt.title("Average Rating of Resturants")
plt.xlabel("Resturants")
plt.show()
```

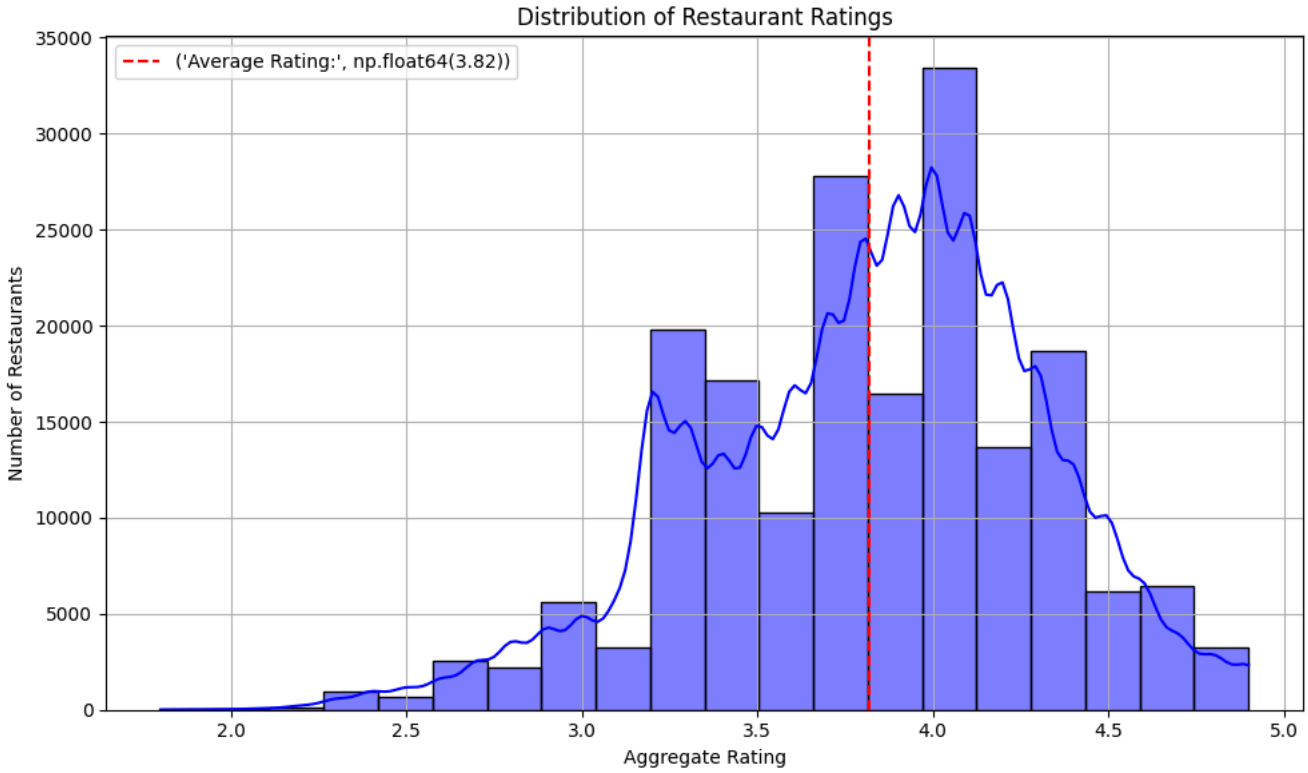


Average Value without 0

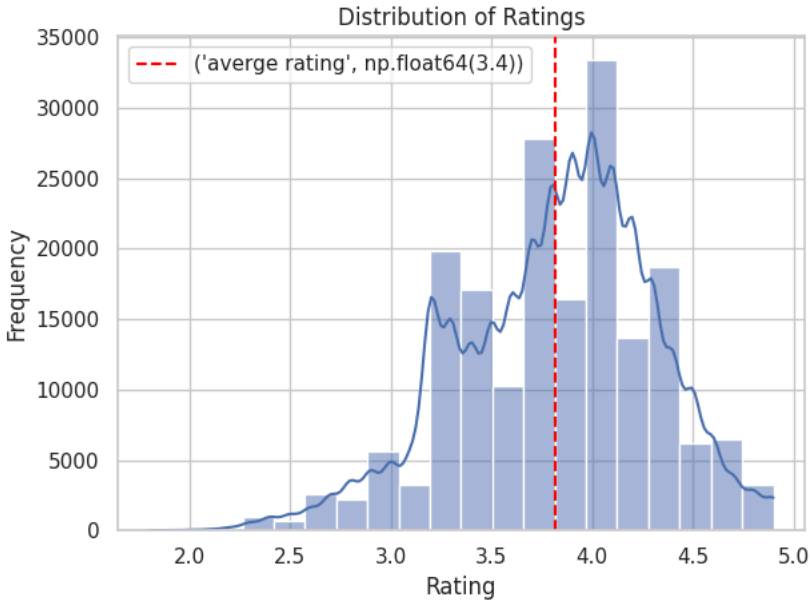
```
valid_ratings = df[(df['aggregate_rating'] > 0)]
average_rating = valid_ratings['aggregate_rating'].mean()
print("Average Rating:",average_rating.round(2))
```

```
Average Rating: 3.82
```

```
plt.figure(figsize=(10,6))
sns.histplot(valid_ratings['aggregate_rating'], bins=20, kde=True, color='blue')
plt.axvline(average_rating, color='red', linestyle='--', label=('Average Rating:',average_rating.round(2) ))
plt.title('Distribution of Restaurant Ratings')
plt.xlabel('Aggregate Rating')
plt.ylabel('Number of Restaurants')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```




```
filtered_ratings = df[df['aggregate_rating'] != 0]['aggregate_rating'].dropna()
sns.histplot(filtered_ratings,bins=20, kde=True)
plt.axvline(average_rating, color='red', linestyle='--', label=("averge rating",avg.round(2)))
plt.title("Distribution of Ratings")
plt.xlabel("Rating")
plt.ylabel("Frequency")
plt.legend()
plt.tight_layout()
plt.show()
```



Double-click (or enter) to edit

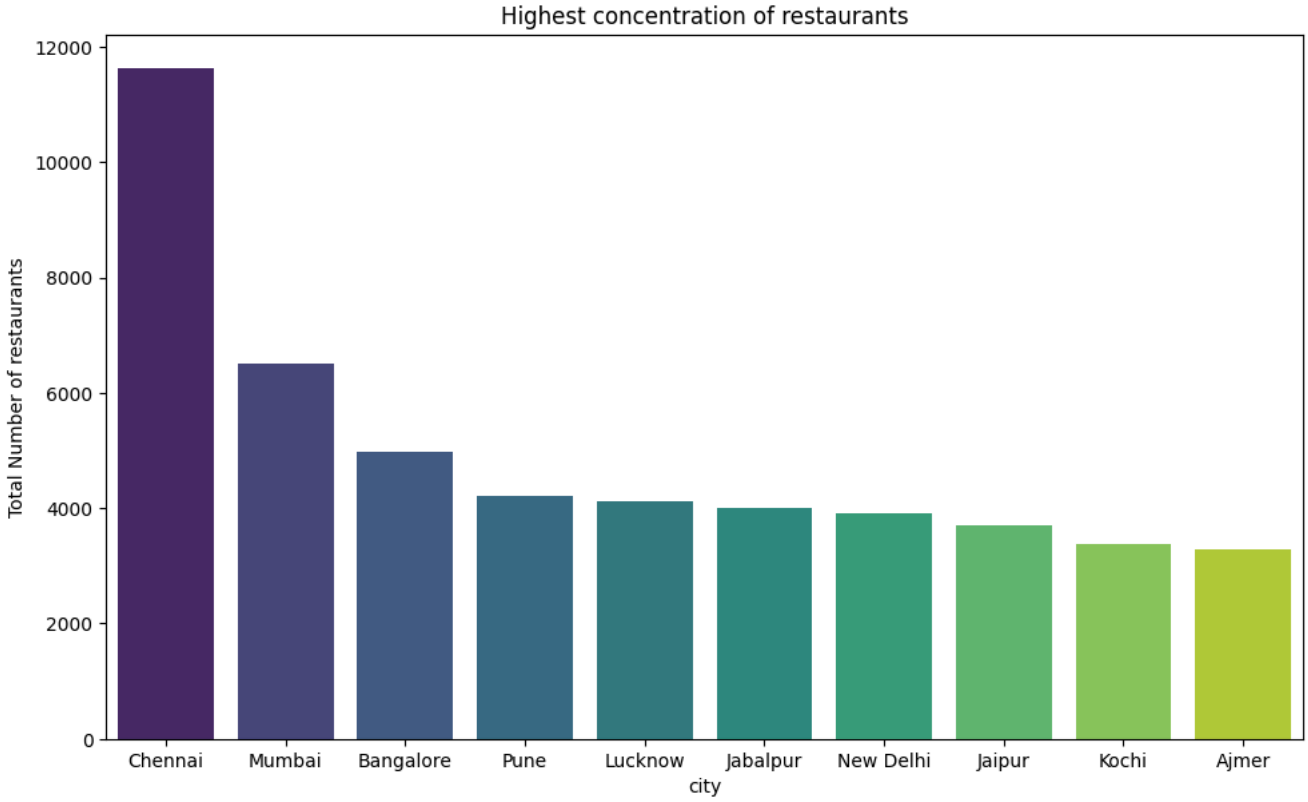
3. Identify the city with the highest concentration of restaurants. Visualize the distribution of restaurant ratings across different cities.

```
cities=df.groupby("city")["name"].agg("count").sort_values(ascending=False)
print("Highest concentration of restaurants :",cities.head(1))
```

```
Highest concentration of restaurants : city
Chennai    11630
Name: name, dtype: int64
```

```
plt.figure(figsize=(10,6))
sns.barplot(data=cities.head(10),palette="viridis")
plt.grid(False)
plt.tight_layout()
plt.title("Highest concentration of restaurants")
plt.ylabel("Total Number of restaurants")
```

```
/tmp/ipython-input-3808714420.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

Text(65.84722222222221, 0.5, 'Total Number of restaurants')
```



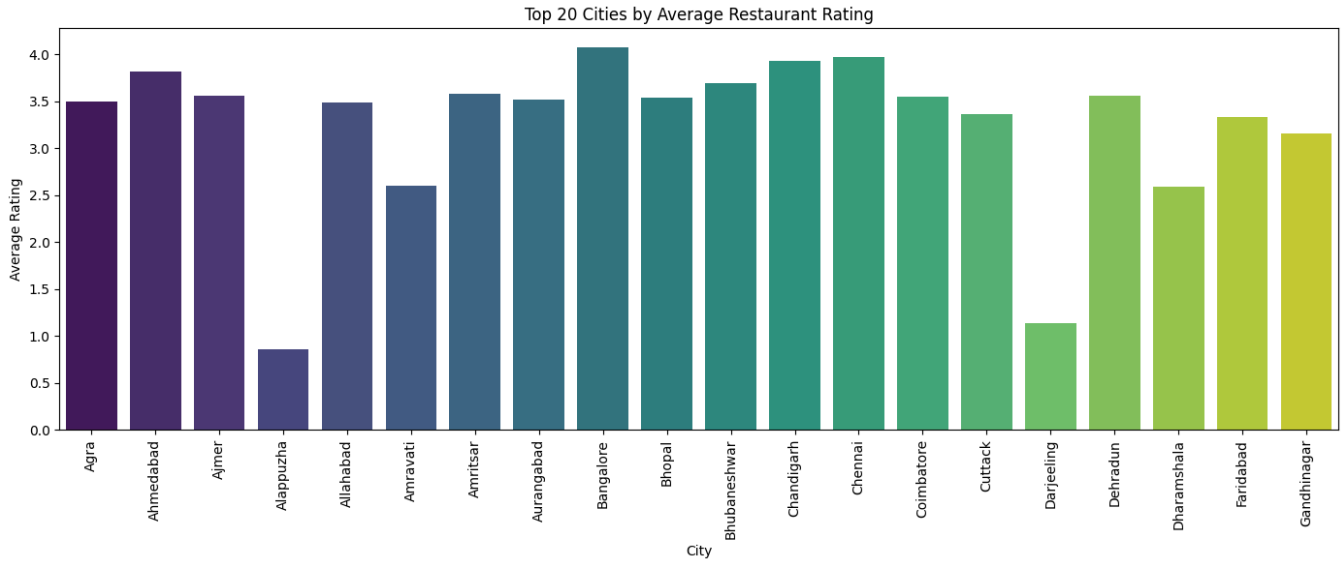

```
city_rating = df.groupby("city")["aggregate_rating"].mean()
city_rating.head(10).round(1)
```

city	aggregate_rating
Agra	3.5
Ahmedabad	3.8
Ajmer	3.6
Alappuzha	0.9
Allahabad	3.5
Amravati	2.6
Amritsar	3.6
Aurangabad	3.5
Bangalore	4.1
Bhopal	3.5

dtype: float64

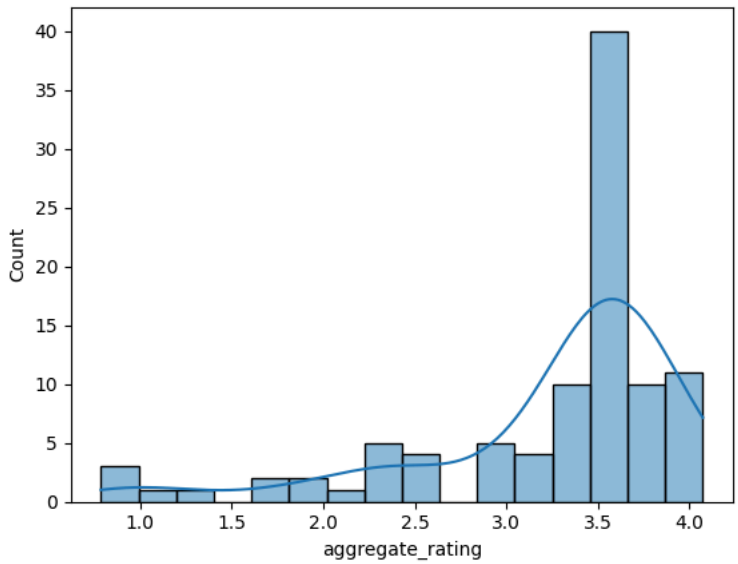
```
city_rating = df.groupby("city")["aggregate_rating"].mean()

plt.figure(figsize=(14,6))
sns.barplot(x=city_rating.head(20).index,y=city_rating.head(20).values, palette="viridis")
plt.title("Top 20 Cities by Average Restaurant Rating")
plt.xlabel("City")
plt.ylabel("Average Rating")
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2660253439.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le
```




```
sns.histplot(data=city_rating,kde=True)
```

```
<Axes: xlabel='aggregate_rating', ylabel='Count'>
```



4. Determine the most popular cuisines among the listed restaurants. Investigate if there's a correlation between the variety of cuisines offered and restaurant ratings.

Here i sepreated the values in different columns

```
df["cuisines"].str.split(",",expand=True)
```

	0	1	2	3	4	5	6	7
0	North Indian	South Indian	Mithai	Street Food	Desserts	None	None	None
1	North Indian	Mughlai	Rolls	Chinese	Fast Food	Street Food	None	None
2	Fast Food	Mithai	None	None	None	None	None	None
3	Desserts	Bakery	Fast Food	South Indian	None	None	None	None
4	North Indian	Continental	Italian	None	None	None	None	None
...	...	...	...	...	...	...	...	...
211939	North Indian	None	None	None	None	None	None	None
211940	Fast Food	None	None	None	None	None	None	None
211941	Gujarati	North Indian	Chinese	None	None	None	None	None
211942	Fast Food	Sandwich	Salad	None	None	None	None	None
211943	Cafe	Healthy Food	Coffee	None	None	None	None	None

211944 rows × 8 columns

Now i sepreated the first column and named it cuisine and septreated the other columns as temp and then i will delete the temp column which is no need in the column

```
df[["cuisine","temp"]]=df["cuisines"].str.split(",",n=1,expand=True)
```

```
df[["cuisine","temp"]]
```

	cuisine	temp
0	North Indian	South Indian, Mithai, Street Food, Desserts
1	North Indian	Mughlai, Rolls, Chinese, Fast Food, Street Food
2	Fast Food	Mithai
3	Desserts	Bakery, Fast Food, South Indian
4	North Indian	Continental, Italian
...	...	...
211939	North Indian	None
211940	Fast Food	None
211941	Gujarati	North Indian, Chinese
211942	Fast Food	Sandwich, Salad
211943	Cafe	Healthy Food, Coffee

211944 rows × 2 columns

Now i deleted the column temp which is unescesary for the data

```
df.drop(columns="temp",axis=1,inplace=True)
```

```
top_cuisine=df.groupby("cuisines")["name"].count().sort_values(ascending=False)
top_cuisine.head(10)
```

cuisines	name
North Indian	17387
Fast Food	6721
Cafe	6190
North Indian, Chinese	5820
South Indian	5217
Pizza, Fast Food	4075
Bakery	3238
Street Food	2837
Biryani	2118
Chinese	2116

dtype: int64

```
df["cuisine"]
```

	cuisine
0	North Indian
1	North Indian
2	Fast Food
3	Desserts
4	North Indian
...	...
211939	North Indian
211940	Fast Food
211941	Gujarati
211942	Fast Food
211943	Cafe

211944 rows × 1 columns

dtype: object

```
popular_cuisines = df.groupby("cuisine").size().reset_index(name="Count")
pop_cuisine=popular_cuisines.sort_values(by="Count",ascending=False)
pop_cuisine
```

	cuisine	Count
84	North Indian	54637
25	Cafe	17680
37	Fast Food	15177
102	South Indian	14117
29	Chinese	11204
...	...	...
45	Grill	4
89	Peruvian	3
12	Bangladeshi	2
19	Bohri	2
53	Indonesian	1

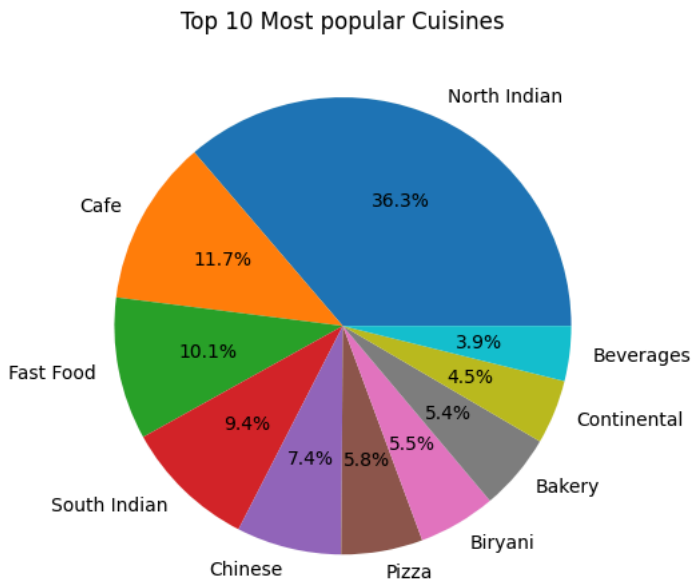
115 rows × 2 columns

Next steps: View recommended plots New interactive sheet

```
plt.pie(pop_cuisine['Count'].head(10),labels=pop_cuisine['cuisine'].head(10),autopct='%1.1f%%')
plt.title('Top 10 Most popular Cuisines')
```

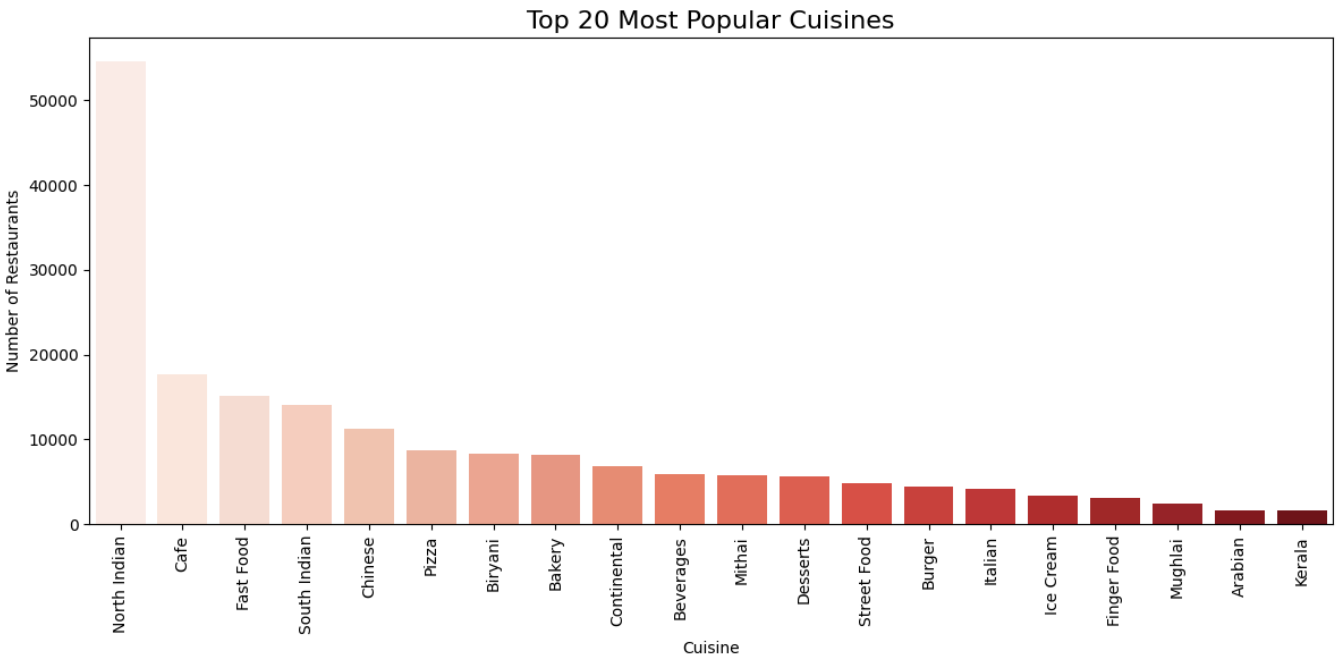
```
plt.tight_layout()
plt.shadow()
plt.show()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
/tmp/ipython-input-1580177389.py in <cell line: 0>()
      2 plt.title('Top 10 Most popular Cuisines')
      3 plt.tight_layout()
----> 4 plt.shadow()
      5 plt.show()

AttributeError: module 'matplotlib.pyplot' has no attribute 'shadow'
```

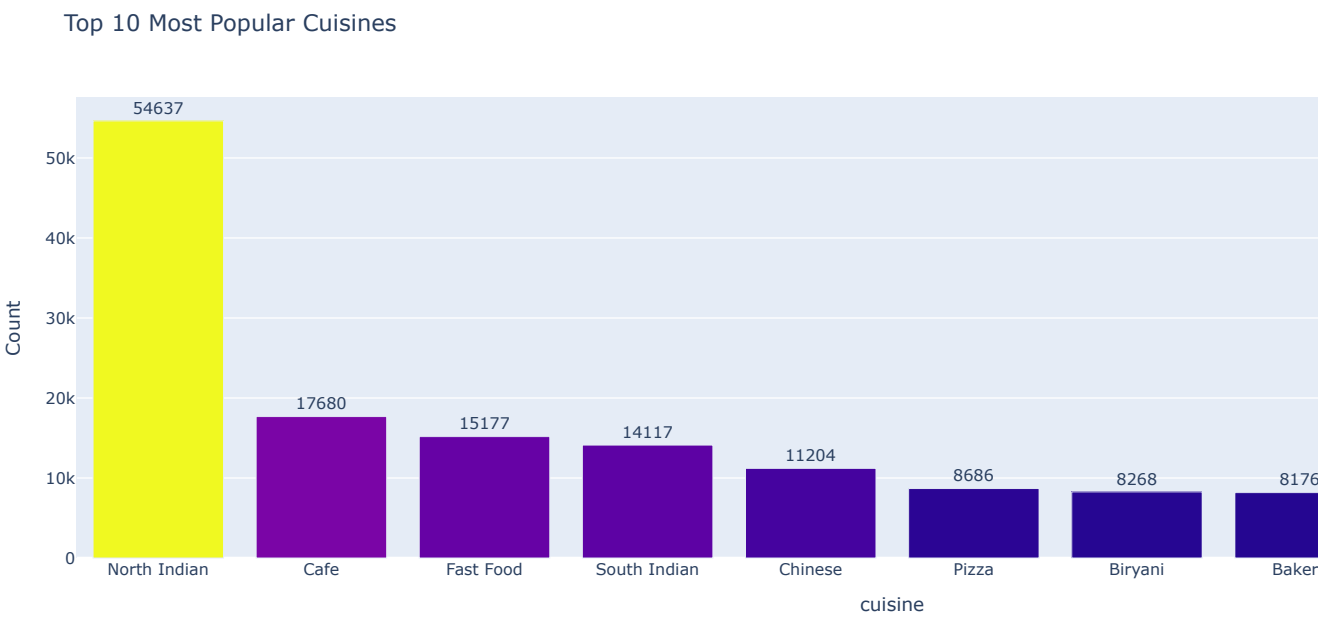



```
plt.figure(figsize=(12,6))
sns.barplot(data=pop_cuisine.head(20), x="cuisine", y="Count", palette="Reds")
plt.title("Top 20 Most Popular Cuisines", fontsize=16)
plt.xlabel("Cuisine")
plt.ylabel("Number of Restaurants")
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-3607238092.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le
```




```
fig = px.bar(pop_cuisine.head(10), x="cuisine", y="Count", text="Count",title="Top 10 Most Popular Cuisines", color="Count")
fig.update_traces(textposition='outside')
fig.show()
```

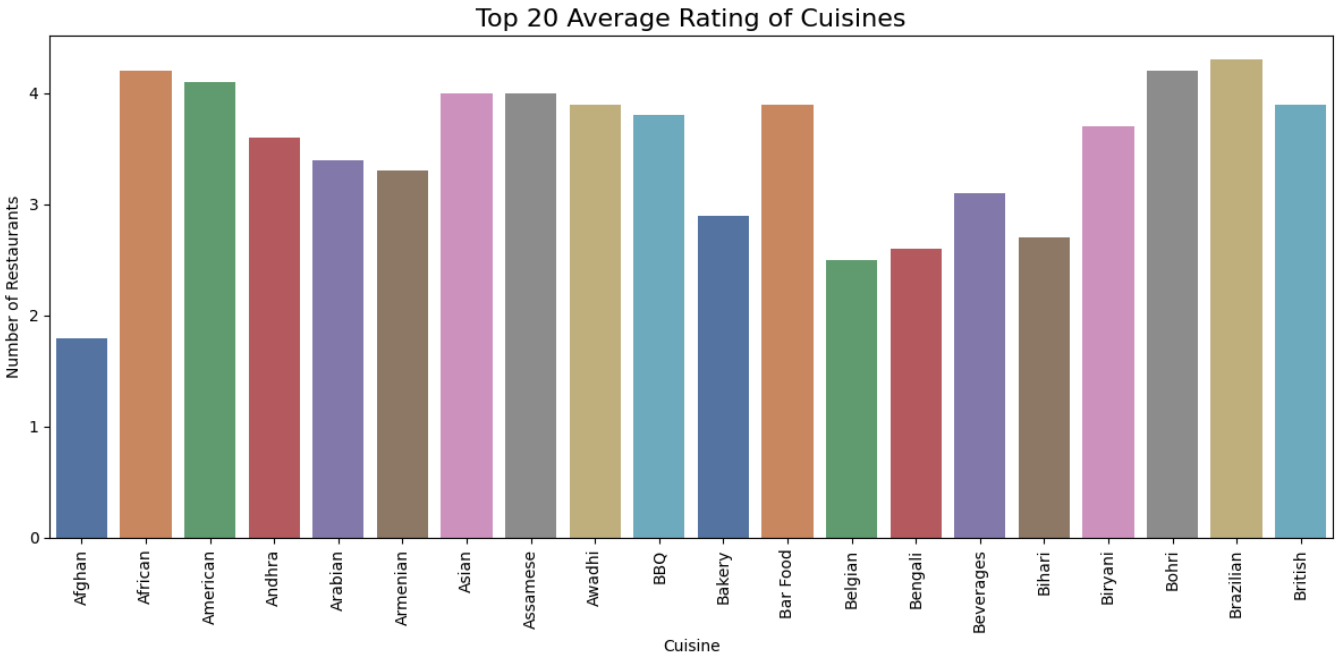



```
cuisne_rating=df.groupby("cuisine")["aggregate_rating"].mean()
cuisne_rate=cuisne_rating[cuisne_rating.round(1)>0]
```

```
plt.figure(figsize=(12,6))
sns.barplot(data=cuisne_rate.head(20).round(1),palette="deep")
plt.title("Top 20 Average Rating of Cuisines", fontsize=16)
plt.xlabel("Cuisine")
plt.ylabel("Number of Restaurants")
plt.xticks(rotation=90)
```

```
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2128797378.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le
```



5. Analyze the relationship between price range and restaurant ratings. Visualize the average cost for two people in different price categories.

```
correlation = df["price_range"].corr(df["aggregate_rating"])
print("the correlation Between Price range and Resturant Ranting is: ",correlation.round(1))
```

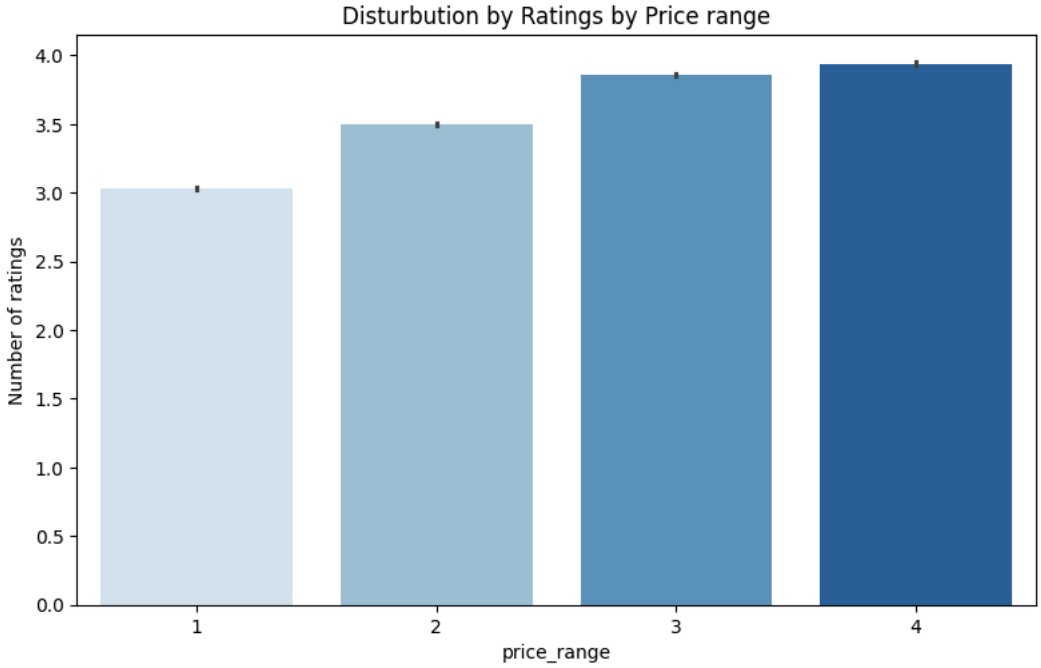
```
the correlation Between Price range and Resturant Ranting is:  0.3
```

```
tt=df["price_range"].unique()
print("number of price range are :",tt)
```

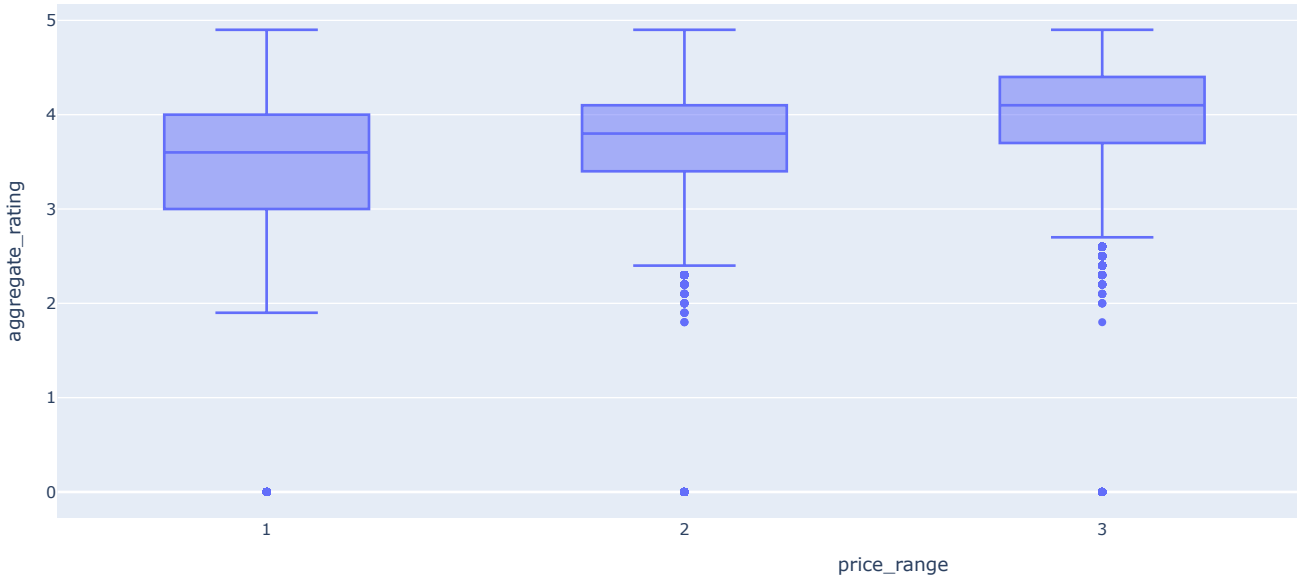
```
number of price range are : [2 1 3 4]
```

```
plt.figure(figsize=(8,5))
sns.barplot(data=df,x="price_range",y="aggregate_rating",palette="Blues")
plt.tight_layout()
plt.title("Disturbution by Ratings by Price range")
plt.ylabel("Number of ratings")
```

```
/tmp/ipython-input-3691854436.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

Text(62.722222222222214, 0.5, 'Number of ratings')
```

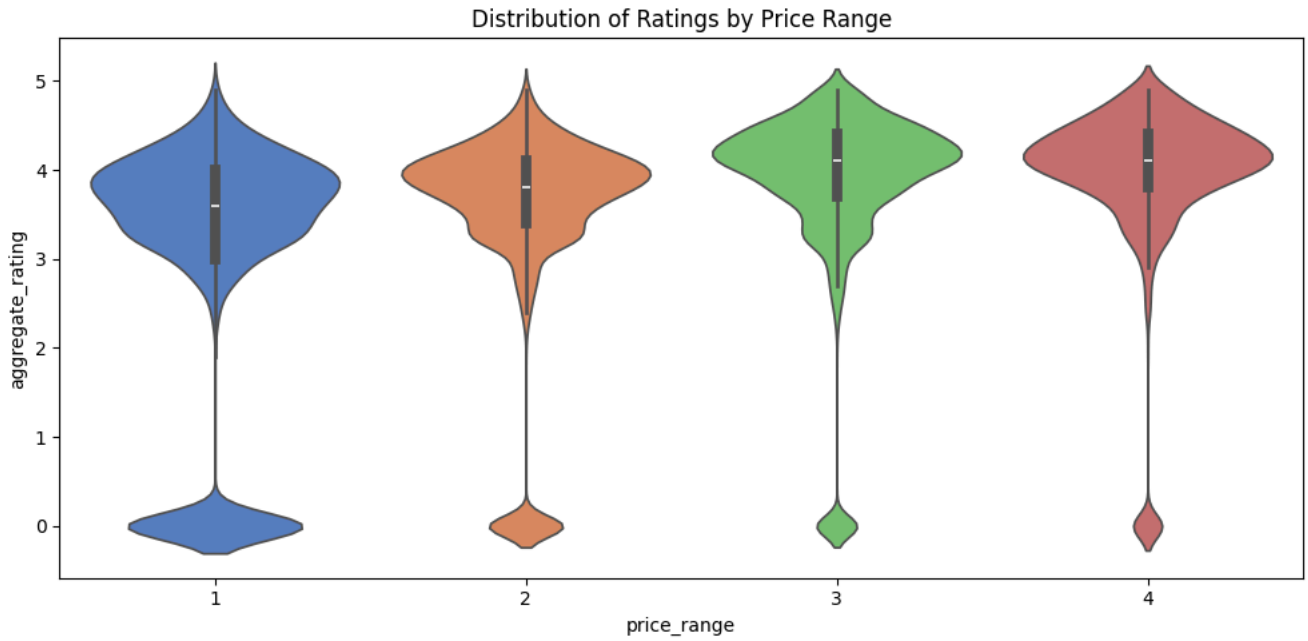



```
q=px.box(df,x="price_range", y="aggregate_rating")
q.show()
```




```
plt.figure(figsize=(10,5))
sns.violinplot(data=df, x="price_range", y="aggregate_rating", palette="muted")
plt.title("Distribution of Ratings by Price Range")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-1583041278.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le
```



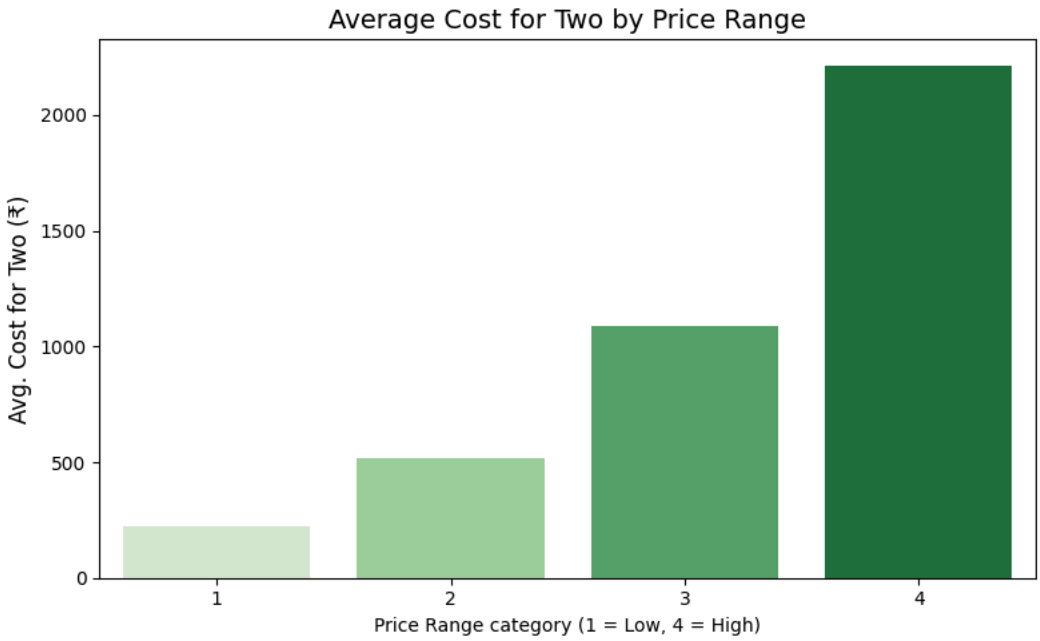

```
avg_price=df.groupby("price_range")["average_cost_for_two"].mean()
avg_price.round(1)
```

price_range	average_cost_for_two
1	225.3
2	516.3
3	1088.0
4	2215.7

dtype: float64

```
plt.figure(figsize=(8,5))
sns.barplot(data=avg_price.round(1), palette="Greens")

plt.title("Average Cost for Two by Price Range", fontsize=14)
plt.xlabel("Price Range category (1 = Low, 4 = High)")
plt.ylabel("Avg. Cost for Two (₹)", fontsize=12)
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-1768541821.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le
```



6. Investigate the impact of online order availability on restaurant ratings. Analyze the distribution of restaurants that offer table booking.

```
df.columns
```

```
Index(['res_id', 'name', 'establishment', 'url', 'city', 'city_id', 'locality',
       'latitude', 'longitude', 'country_id', 'locality_verbose', 'cuisines',
       'timings', 'average_cost_for_two', 'price_range', 'currency',
       'highlights', 'aggregate_rating', 'rating_text', 'votes', 'photo_count',
       'opentable_support', 'delivery', 'takeaway', 'cuisine'],
      dtype='object')
```

In this delivery column i consider 1 as yes and 0 as no and -1 for both

```
df["delivery"].unique()
```

```
array([-1,  1,  0])
```

i created a new column name online_order_availability in which i put yes, no and both values for online orders

```
df["online_order_availability"] = df["delivery"].apply(lambda x: "Yes" if x == 1 else ("No" if x == 0 else "Both"))
```

```
avg_online=df.groupby("online_order_availability")["aggregate_rating"].mean()
avg_online.round(1)
```

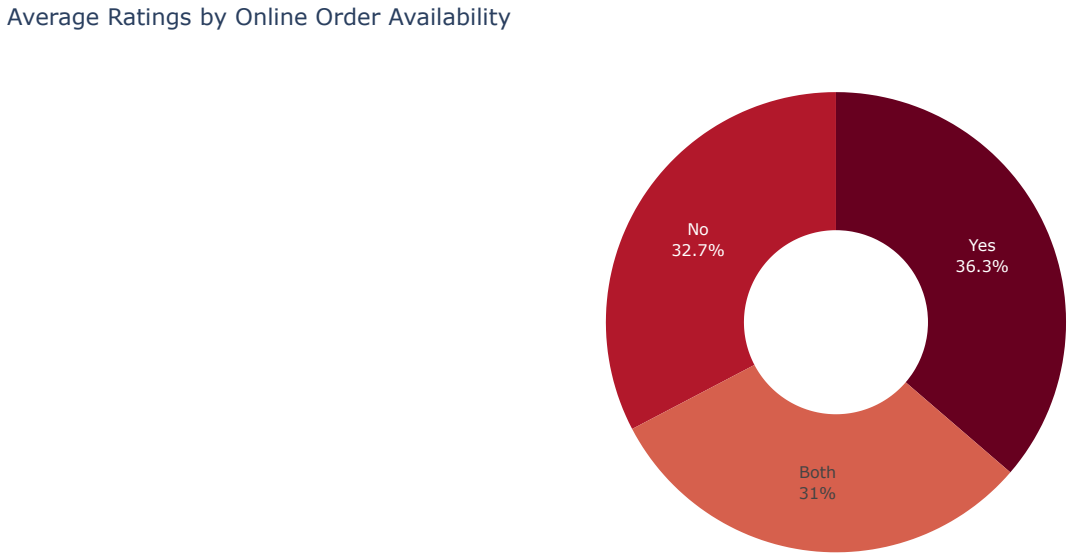
online_order_availability	aggregate_rating
Both	3.2
No	3.4
Yes	3.7

dtype: float64

```
sns.barplot(data=avg_online,palette='Reds')
plt.title("Average Ratings by Online Order Availability")
```

```
/tmp/ipython-input-3920566209.py:1: FutureWarning:


Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le


Text(0.5, 1.0, 'Average Ratings by Online Order Availability')
```

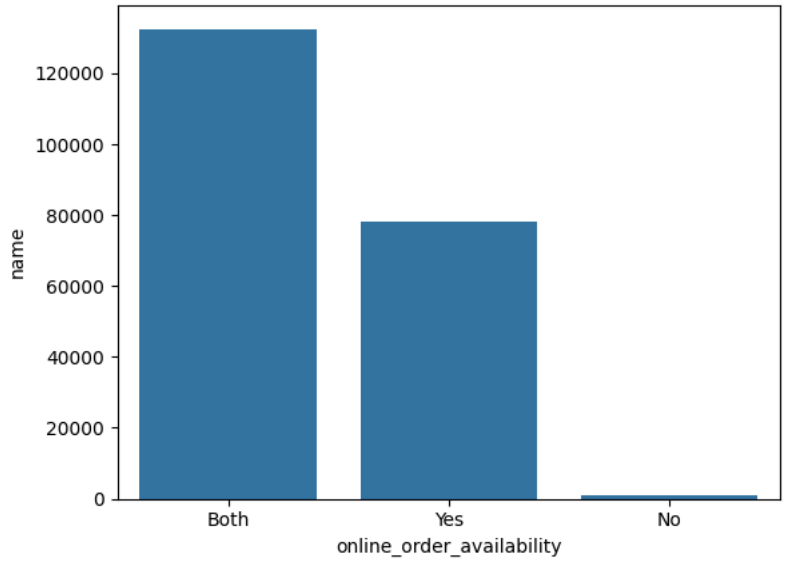

```
gig=px.pie(avg_online,values="aggregate_rating",names=["Both","No","Yes"],title="Average Ratings by Online Order Availability",color_di
gig.update_traces(textinfo="percent+label",hovertemplate="Availability: %{label}<br>Rating: %{value}<extra></extra>")
gig.show()
```




```
k=df.groupby("online_order_availability")["name"].count().sort_values(ascending=False)
```

```
sns.barplot(data=k)
```

```
<Axes: xlabel='online_order_availability', ylabel='name'>
```




```
df["opentable_support"].unique()
```

```
array([0., 1.])
```

here i change the datatype of opentable_support from float to int

```
df["opentable_support"] = df["opentable_support"].astype(int)
```

```
df["opentable_support"].unique()
```

```
array([0, 1])
```

here i create a new column name open table in which i added if the column had 1 then yes it support_open_table if no it doesnt support

```
df["open_table"]=np.where(df["opentable_support"]==1,"support_open_table","does_not_support_open_table")
```

```
df.groupby(["opentable_support","open_table"])["name"].count()
```

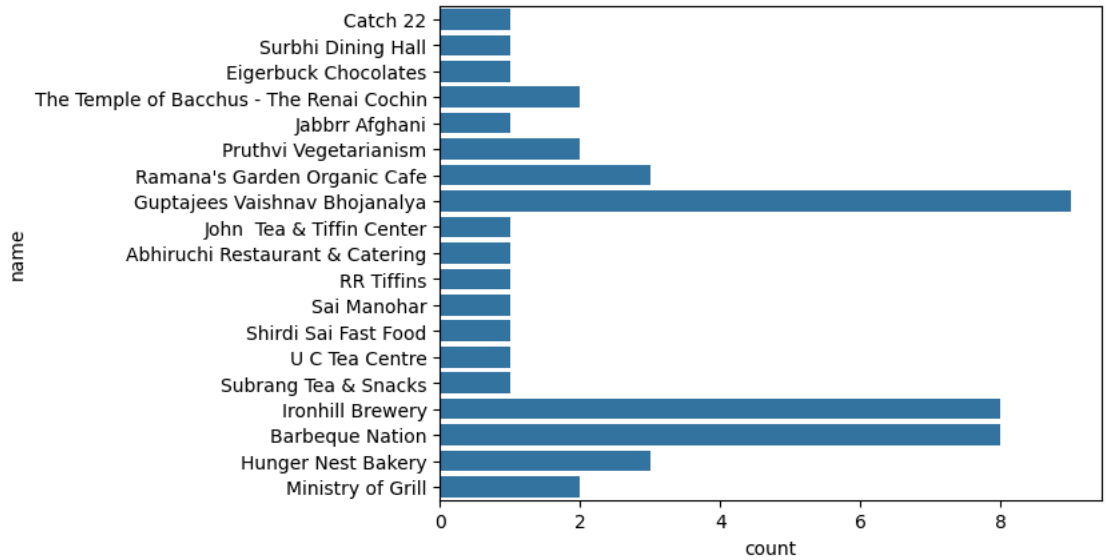
opentable_support	open_table	name
0	does_not_support_open_table	211896
1	support_open_table	48

dtype: int64

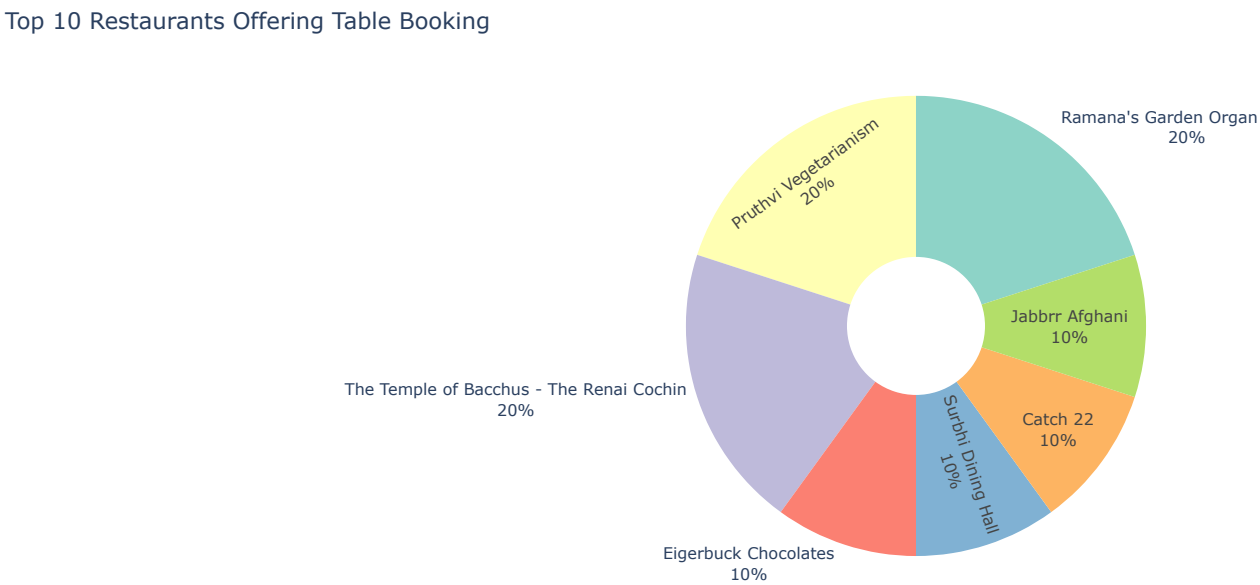
```
offer_booking=df[df["opentable_support"]==1]
```

```
sns.countplot(data=offer_booking,y="name")
```

```
<Axes: xlabel='count', ylabel='name'>
```




```
table_counts = offer_booking.head(10)['name'].value_counts().reset_index()
table_counts.columns = ['Table Booking', 'Count']

fig = px.pie(
    table_counts,
    names='Table Booking',
    values='Count',
    title='Top 10 Restaurants Offering Table Booking',
    color_discrete_sequence=px.colors.qualitative.Set3)

fig.update_traces(
    textinfo='percent+label',
    hole=0.3
)

fig.show()
```




```
table_counts = offer_booking['name'].value_counts().reset_index()
table_counts.columns = ['Table Booking', 'Count']

fig = px.bar(table_counts.head(10),x='Table Booking',  y='Count', color='Table Booking', text='Count',title='Number of Restaurants that
             color_discrete_sequence=px.colors.qualitative.Set2)
fig.show()
```

7. Identify and visualize the top restaurant chains based on the number of outlets. Explore the ratings of these top chains.

```
df.columns
```

```
Index(['res_id', 'name', 'establishment', 'url', 'city', 'city_id', 'locality',
       'latitude', 'longitude', 'country_id', 'locality_verbose', 'cuisines',
       'timings', 'average_cost_for_two', 'price_range', 'currency',
       'highlights', 'aggregate_rating', 'rating_text', 'votes', 'photo_count',
       'opentable_support', 'delivery', 'takeaway', 'cuisine',
       'online_order_availability', 'open_table'],
      dtype='object')
```

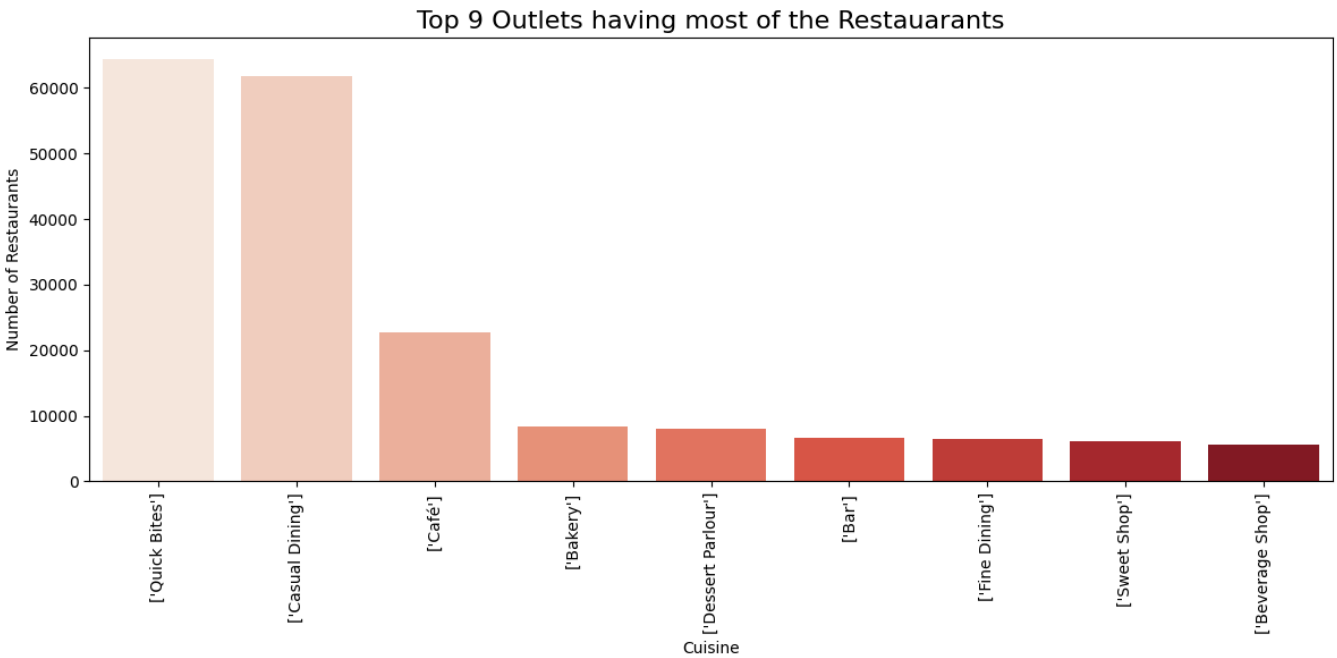
```
an=df.groupby("establishment")["name"].count().reset_index()
outlet=an.sort_values(by="name",ascending=False)
outlet
```

	establishment	name
23	['Quick Bites']	64390
6	['Casual Dining']	61808
5	['Café']	22760
0	['Bakery']	8282
10	['Dessert Parlour']	7961
1	['Bar']	6553
12	['Fine Dining']	6401
25	['Sweet Shop']	6103
2	['Beverage Shop']	5571
22	['Pub']	1396
14	['Food Truck']	1289
19	['Microbrewery']	866
18	['Mess']	611

Next steps: View recommended plots New interactive sheet

```
plt.figure(figsize=(12,6))
sns.barplot(data=outlet.head(9), x="establishment", y="name", palette="Reds")
plt.title("Top 9 Outlets having most of the Restauarants", fontsize=16)
plt.xlabel("Cuisine")
plt.ylabel("Number of Restaurants")
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2252442510.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le
```




```
outlet_rating=df.groupby("establishment")["aggregate_rating"].mean()
outlet_rating.round(1)
```