Zomato Data Set EDA

DS Trainer:- Ankit Mishra

INTRODUCTION

Project Overview

The restaurant industry significantly shapes regional culinary cultures, influenced by urbanization and increasing dining-out trends. This study utilizes a Zomato dataset to analyze consumer preferences, spending habits, and service expectations in India. Starting with over 200,000 rows, data cleaning narrowed it down to 55,000 unique restaurants, ensuring accurate insights. Through exploratory data analysis (EDA), the research identifies patterns in restaurant distribution, types, cuisine preferences, and pricing, providing valuable information for restaurateurs and investors. The study aims to highlight growth opportunities and identify profitable restaurant types in India.

Here's a summary of the libraries used:

NumPy: Ideal for mathematical computations, it handles multidimensional data, making it suitable for various datasets.

Pandas: Built on NumPy, it provides high-level data structures like Series and DataFrames for effective data analysis and manipulation.

Matplotlib: A versatile library for creating static, animated, and interactive visualizations, offering extensive customization options for different types of plots.

Seaborn: Built on Matplotlib, it simplifies the creation of statistical graphics, integrating well with pandas DataFrames for enhanced data exploration and visualization.

In [73]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import os
print('All Modules Imported Successfully!!')
```

```
All Modules Imported Successfully!!
```

In [2]:
```
df = pd.read_csv(r"C:\Users\Lenovo\Downloads\zomato_restaurants_in_India.cs
print('Data Loaded Successfully!!')
```

```
Data Loaded Successfully!!
```

Data Cleaning

In [3]:
```
df.head(3)
```

Out[3]:

	res_id	name	establishment	url	address	city
0	3400299	Bikanervala	['Quick Bites']	https://www.zomato.com/agra/bikanervala-khanda...	Kalyani Point, Near Tulsi Cinema, Bypass Road,...	Agra
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
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

3 rows × 26 columns

In [4]:
```
print(f"Number of Columns: {df.shape[1]} \nNumber of Rows: {df.shape[0]}")
```

```
Number of Columns: 26
Number of Rows: 211944
```

In [5]:
```
# To check Redunant Data
df['res_id'].nunique()
```

Out[5]:
```
55568
```

In [6]:
```
df.drop_duplicates('res_id',keep='first',inplace=True)
```

In [7]:
```
print(f"Number of Columns: {df.shape[1]} \nNumber of Rows: {df.shape[0]}")
```

```
Number of Columns: 26
Number of Rows: 55568
```

- After removing duplicates res_id from dataset , no we have 55568 unique rows out of 211944 rows

In [8]:
```
df.sample()
```

Out[8]:

	res_id	name	establishment	url	address	
95122	18899909	Ashok Bakery	['Bakery']	https://www.zomato.com/jamnagar/ashok-bakery-b...	Siddhanth Complex Limda Lane	Jam

1 rows × 26 columns

In [9]:
```
#now set the res_id as index
# Why??
df.set_index('res_id',inplace=True)
```

In [10]:
```
df.sample()
```

Out[10]:

res_id	name	establishment	url	address	
3701084	Golden Restaurant	['Quick Bites']	https://www.zomato.com/puducherry/golden-resta...	Hotel South Avenue, Opposite to Railway statio...	Puduch

1 rows × 25 columns

In [11]:
```
df.shape
```

Out[11]:
```
(55568, 25)
```

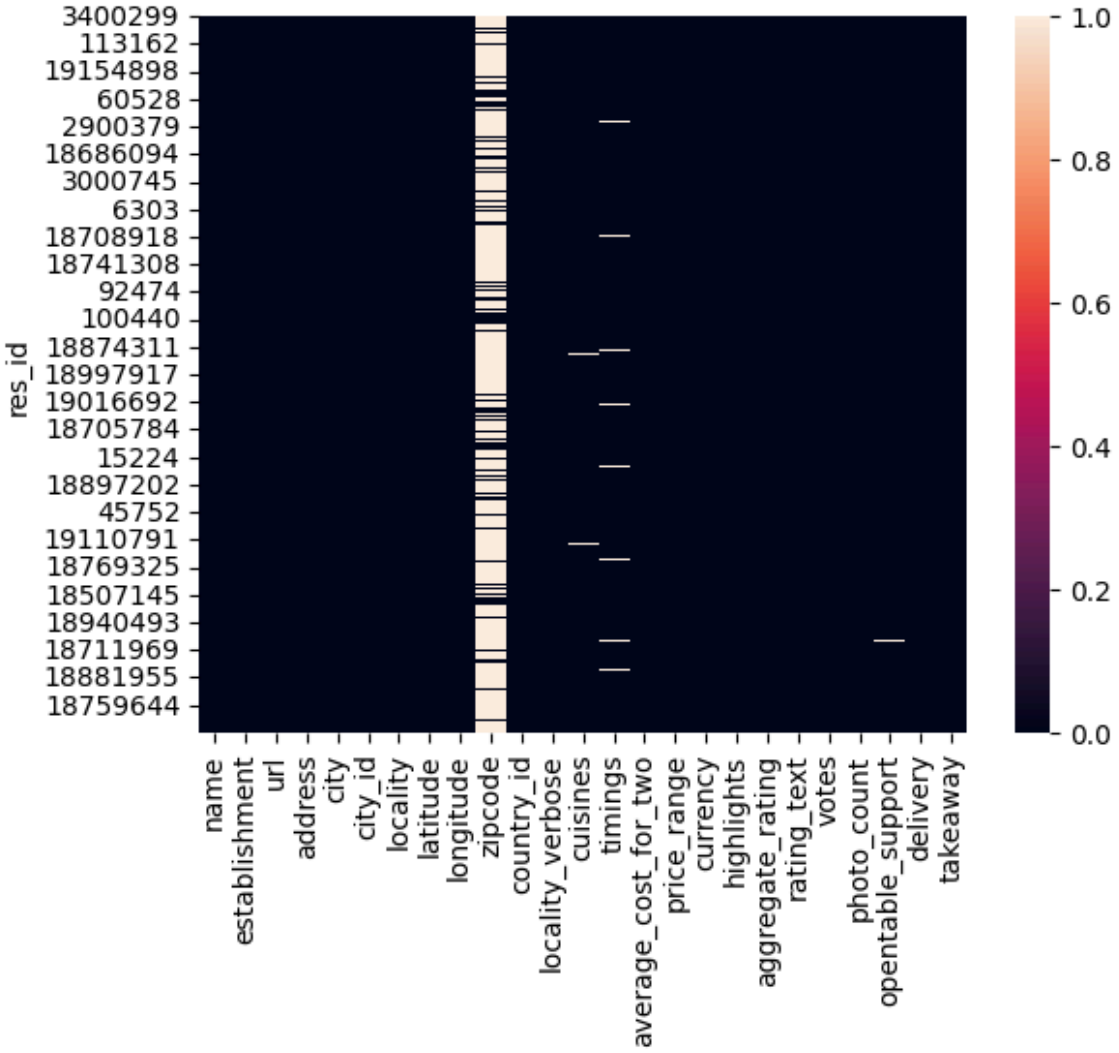
In [ ]:

In [12]:
```
#use to check datatype and nonnull values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 55568 entries, 3400299 to 3201138
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   name                  55568 non-null  object 
 1   establishment         55568 non-null  object 
 2   url                   55568 non-null  object 
 3   address               55550 non-null  object 
 4   city                  55568 non-null  object 
 5   city_id               55568 non-null  int64  
 6   locality              55568 non-null  object 
 7   latitude              55568 non-null  float64
 8   longitude             55568 non-null  float64
 9   zipcode               10945 non-null  object 
 10  country_id            55568 non-null  int64  
 11  locality_verbose      55568 non-null  object 
 12  cuisines              55098 non-null  object 
 13  timings               54565 non-null  object 
 14  average_cost_for_two  55568 non-null  int64  
 15  price_range           55568 non-null  int64  
 16  currency              55568 non-null  object 
 17  highlights            55568 non-null  object 
 18  aggregate_rating      55568 non-null  float64
 19  rating_text           55568 non-null  object 
 20  votes                 55568 non-null  int64  
 21  photo_count           55568 non-null  int64  
 22  opentable_support     55556 non-null  float64
 23  delivery              55568 non-null  int64  
 24  takeaway              55568 non-null  int64  
dtypes: float64(4), int64(8), object(13)
memory usage: 11.0+ MB
```

In [13]:
```
# timings,currency
```

In [14]:
```
# To check null values
sns.heatmap(df.isna())
plt.show()
```



In [15]:
```
df.shape
```

Out[15]: (55568, 25)

In [16]:
```
((df.isnull().sum()/len(df))*100).round(2)
```

Out[16]:
```
name                    0.00
establishment           0.00
url                     0.00
address                 0.03
city                    0.00
city_id                 0.00
locality                0.00
latitude                0.00
longitude               0.00
zipcode                80.30
country_id              0.00
locality_verbose        0.00
cuisines                0.85
timings                 1.80
average_cost_for_two    0.00
price_range             0.00
currency                0.00
highlights              0.00
aggregate_rating        0.00
rating_text             0.00
votes                   0.00
photo_count             0.00
opentable_support       0.02
delivery                0.00
takeaway                0.00
dtype: float64
```

In [17]:
```
# Zipcode:- To do
```

Statstical Analysis

In [18]:
```
df.describe().round(2)
```

Out[18]:

	city_id	latitude	longitude	country_id	average_cost_for_two	price_range	aggrega
count	55568.00	55568.00	55568.00	55568.0	55568.00	55568.00	
mean	3409.50	21.45	76.50	1.0	528.21	1.71	
std	5174.94	42.90	10.98	0.0	595.03	0.88	
min	1.00	0.00	0.00	1.0	0.00	1.00	
25%	8.00	16.52	74.65	1.0	200.00	1.00	
50%	26.00	22.47	77.11	1.0	350.00	1.00	
75%	11294.00	26.75	79.83	1.0	600.00	2.00	
max	11354.00	10000.00	91.83	1.0	30000.00	4.00	

In [19]:
```
df['establishment'].unique()
```

Out[19]:
```
array(["['Quick Bites']", "['Casual Dining']", "['Bakery']", "['Café']",
       "['Dhaba']", "['Bhojanalya']", "['Bar']", "['Sweet Shop']",
       "['Fine Dining']", "['Food Truck']", "['Dessert Parlour']",
       "['Lounge']", "['Pub']", "['Beverage Shop']", "['Kiosk']",
       "['Paan Shop']", "['Confectionery']", '[]', "['Shack']",
       "['Club']", "['Food Court']", "['Mess']", "['Butcher Shop']",
       "['Microbrewery']", "['Cocktail Bar']", "['Pop up']",
       "['Irani Cafe']"], dtype=object)
```

In [20]:
```
for i in df['establishment'].unique():
    print(i)
```

```
['Quick Bites']
['Casual Dining']
['Bakery']
['Café']
['Dhaba']
['Bhojanalya']
['Bar']
['Sweet Shop']
['Fine Dining']
['Food Truck']
['Dessert Parlour']
['Lounge']
['Pub']
['Beverage Shop']
['Kiosk']
['Paan Shop']
['Confectionery']
[]
['Shack']
['Club']
['Food Court']
['Mess']
['Butcher Shop']
['Microbrewery']
['Cocktail Bar']
['Pop up']
['Irani Cafe']
```

In [21]:
```
df['establishment'].nunique()
```

Out[21]:
```
27
```

In [22]:
```
# ['Irani Cafe']
''
```

Out[22]:
```
''
```

In [23]:
```
#Remove [''] Frrom establishment columns
# Replace  [] to N/A



df['establishment'] = df['establishment'].apply(lambda x : x[2:-2])
```

In [24]:
```
df['establishment'] = df['establishment'].apply(lambda x : np.where(x=="",
```

In [ ]:
```

```

In [ ]:
```

```

In [25]:
```
df['establishment'].unique()
```

Out[25]:
```
array(['Quick Bites', 'Casual Dining', 'Bakery', 'Café', 'Dhaba',
       'Bhojanalya', 'Bar', 'Sweet Shop', 'Fine Dining', 'Food Truck',
       'Dessert Parlour', 'Lounge', 'Pub', 'Beverage Shop', 'Kiosk',
       'Paan Shop', 'Confectionery', 'NA', 'Shack', 'Club', 'Food Court',
       'Mess', 'Butcher Shop', 'Microbrewery', 'Cocktail Bar', 'Pop up',
       'Irani Cafe'], dtype=object)
```

In [26]:
```
df['establishment'].nunique()
```

Out[26]:
```
27
```

In [27]:
```
df['establishment'].nunique()
```

Out[27]:
```
27
```

In [28]:
```
df.columns
```

Out[28]:
```
Index(['name', 'establishment', 'url', 'address', 'city', 'city_id',
       'locality', 'latitude', 'longitude', 'zipcode', 'country_id',
       'locality_verbose', 'cuisines', 'timings', 'average_cost_for_two',
       'price_range', 'currency', 'highlights', 'aggregate_rating',
       'rating_text', 'votes', 'photo_count', 'opentable_support', 'delive
ry',
       'takeaway'],
      dtype='object')
```

Todo

- 1>> Zip Code
- 2>> Timing
- 3>> Currency
- 4>> NA

In [29]:
```
df['zipcode'].fillna('Others',inplace = True)
```

In [30]:
```
df['zipcode'].isna().sum()
```

Out[30]:
```
0
```

In [31]:
```
df['timings'].nunique()
```

Out[31]:
```
7740
```

In [32]:
```
df['timings']
```

Out[32]:
```
res_id
3400299                                8:30am – 10:30pm (Mon-Sun)
3400005     12:30PM to 12Midnight (Mon, Wed, Thu, Fri, Sat...
3401013                                        9:30 AM to 11 PM
3400290                                    8am – 11pm (Mon-Sun)
3401744                                    11:30 AM to 11:30 PM
                                    ...
19142822                                   11 AM to 12 Midnight
18984164                11:30 AM to 3:30 PM, 7:30 PM to 11 PM
18019952                   11 AM to 3:30 PM, 7 PM to 10:30 PM
3200996     4pm – 11pm (Mon, Tue, Wed, Fri, Sat, Sun), 4pm...
3201138                                            8 AM to 1 AM
Name: timings, Length: 55568, dtype: object
```

In [33]:
```
df['currency'].unique()
```

Out[33]:
```
array(['Rs.'], dtype=object)
```

In [34]:
```
df[df['establishment'] == 'NA'].shape
```

Out[34]:
```
(1830, 25)
```

In [35]:
```
df['establishment'].unique()
```

Out[35]:
```
array(['Quick Bites', 'Casual Dining', 'Bakery', 'Café', 'Dhaba',
       'Bhojanalya', 'Bar', 'Sweet Shop', 'Fine Dining', 'Food Truck',
       'Dessert Parlour', 'Lounge', 'Pub', 'Beverage Shop', 'Kiosk',
       'Paan Shop', 'Confectionery', 'NA', 'Shack', 'Club', 'Food Court',
       'Mess', 'Butcher Shop', 'Microbrewery', 'Cocktail Bar', 'Pop up',
       'Irani Cafe'], dtype=object)
```

In [36]:
```
df['establishment'].replace('NA','Other establishments',inplace=True)
```

In [37]:
```
df['establishment'].unique()
```

Out[37]:
```
array(['Quick Bites', 'Casual Dining', 'Bakery', 'Café', 'Dhaba',
       'Bhojanalya', 'Bar', 'Sweet Shop', 'Fine Dining', 'Food Truck',
       'Dessert Parlour', 'Lounge', 'Pub', 'Beverage Shop', 'Kiosk',
       'Paan Shop', 'Confectionery', 'Other establishments', 'Shack',
       'Club', 'Food Court', 'Mess', 'Butcher Shop', 'Microbrewery',
       'Cocktail Bar', 'Pop up', 'Irani Cafe'], dtype=object)
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Top Data Analysis

In [ ]:

In [ ]:

In [ ]:

In [38]:
```
top_20_cities = df['city'].value_counts().head(20).reset_index()
top_20_cities.head(5)
```

Out[38]:

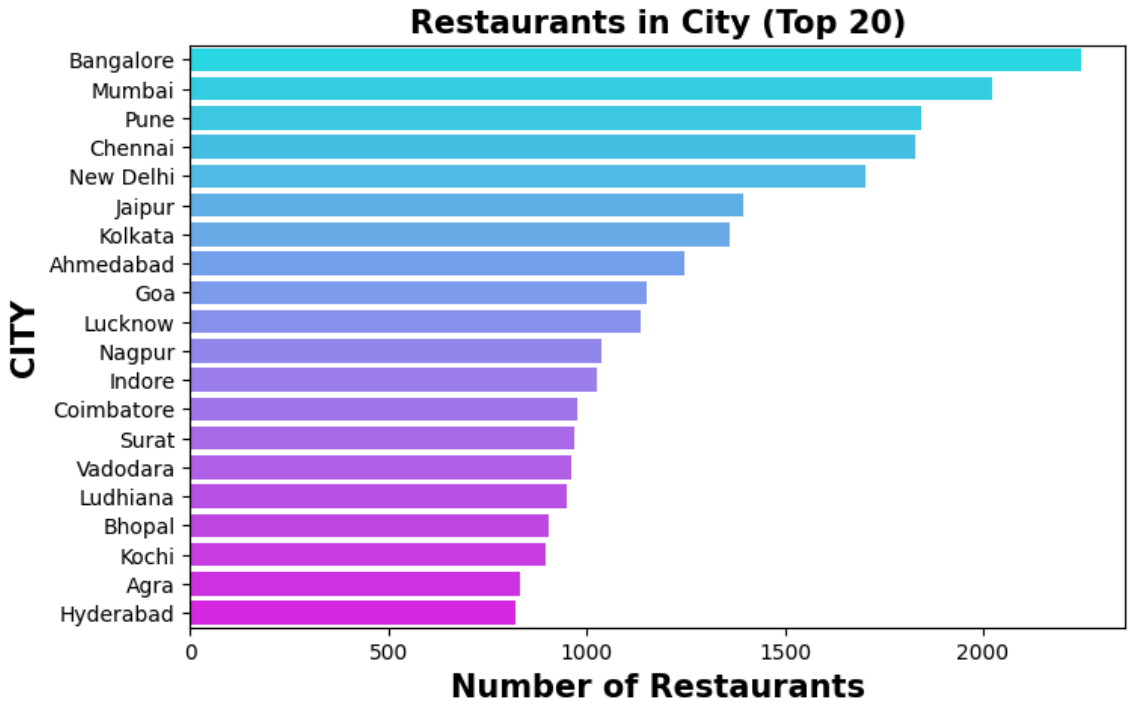
	city	count
0	Bangalore	2247
1	Mumbai	2022
2	Pune	1843
3	Chennai	1827
4	New Delhi	1704

In [ ]:

In [ ]:

In [ ]:

In [39]:
```
plt.figure(figsize=(8,5))
sns.barplot(y='city',x='count',data=top_20_cities,palette='cool')

plt.title('Restaurants in City (Top 20)',fontsize=15,fontweight='bold')
plt.ylabel('CITY',fontsize=15,fontweight='bold')
plt.xlabel('Number of Restaurants',fontsize=15,fontweight='bold')

plt.show()
```



The bar graph showing the top 20 cities ranked by number of Restaurants ,here are some insights:

- Bangalore and Mumbai have the highest number of restaurants among the cities listed.
- There is a noticeable drop after Bangalore and Mumbai .in the number of restaurants for the next group of cities like Pune, Chennai, and New Delhi,Jabalpur
- The cities towards the bottom of the graph like Kochi, Agra, and Hyderabad have relatively fewer restaurants compared to the top cities.
- As you can see that metro cities have more number of restaurants than others with South India dominating the Top 4 (Bangalore, Pune , Mumbai, Chennai)

In [ ]:

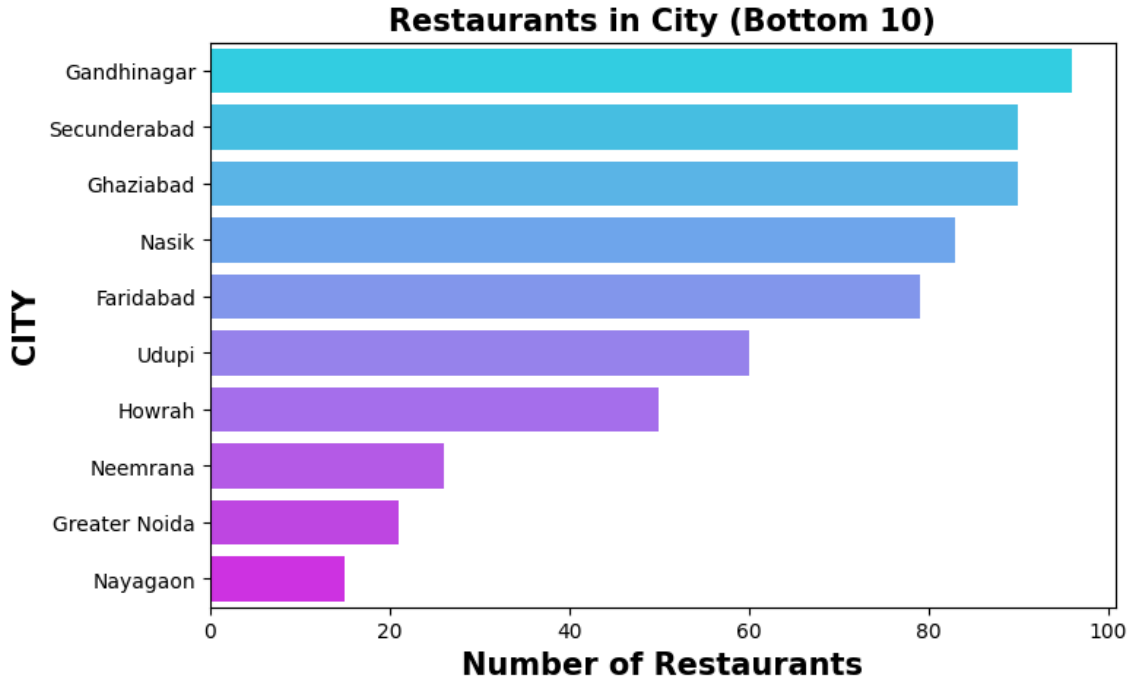
In [40]:
```
bottom_10_cities = df['city'].value_counts().tail(10).reset_index()
bottom_10_cities.head()
```

Out[40]:

	city	count
0	Gandhinagar	96
1	Secunderabad	90
2	Ghaziabad	90
3	Nasik	83
4	Faridabad	79

In [ ]:
```
```

In [41]:
```
plt.figure(figsize=(8,5))
sns.barplot(y='city',x='count',data=bottom_10_cities,palette='cool')

plt.title('Restaurants in City (Bottom 10)',fontsize=15,fontweight='bold')
plt.ylabel('CITY',fontsize=15,fontweight='bold')
plt.xlabel('Number of Restaurants',fontsize=15,fontweight='bold')

plt.show()
```



In [ ]:
```
```

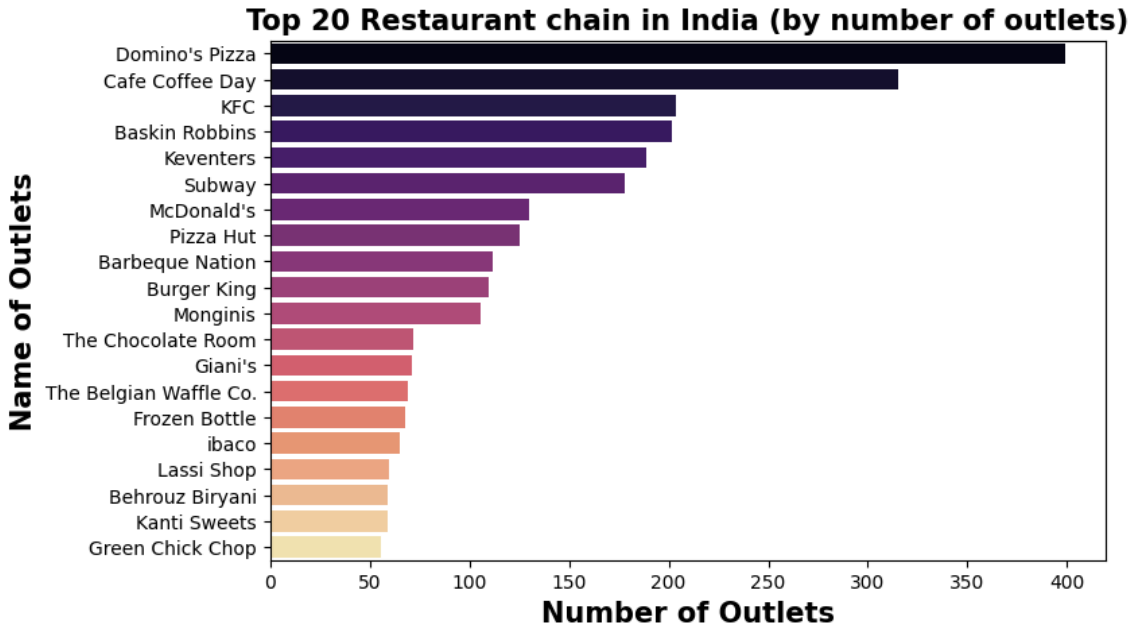
In [42]:
```
top_20_name = df['name'].value_counts().head(20).reset_index()
```

In [43]:
```
top_20_name.head()
```

Out[43]:

	name	count
0	Domino's Pizza	399
1	Cafe Coffee Day	315
2	KFC	204
3	Baskin Robbins	202
4	Keventers	189

In [44]:
```
plt.figure(figsize=(8,5))
sns.barplot(y='name',x='count',data=top_20_name,palette='magma')
plt.title('Top 20 Restaurant chain in India (by number of outlets)',fontsiz

plt.xlabel('Number of Outlets',fontsize=15,fontweight='bold')
plt.ylabel('Name of Outlets',fontsize=15,fontweight='bold')

plt.show()
```



The bar graph represents the top 20 restaurants chain in India. Based on number of outlets they operated. You can see in the figure Domino's Pizza have highest number of outlets 390+ & Café Coffee Day (CCD) have 300+ outlets all over the India and KFC & Baskin Robbins have same size Bar.

- Through this graphical visualization we can say that big giants like Domino's, CCD, KFC are dominating the restaurant industry.
- consumers have a strong preference for certain types of cuisine, including

1. Pizza : Chart is lead by the Domino's. Number of outlets indicating the significance demand for pizza restaurants across India.

2.Burgers: Several burger chains are also in the top 20 McDonald's, Burger King, and Keventers suggesting that burger restaurants are popular among consumers.

1 Chicken-based restaurants: KFC (Kentucky Fried Chicken) ranks third in terms of the

In [ ]:

To Do Bottom:- Name

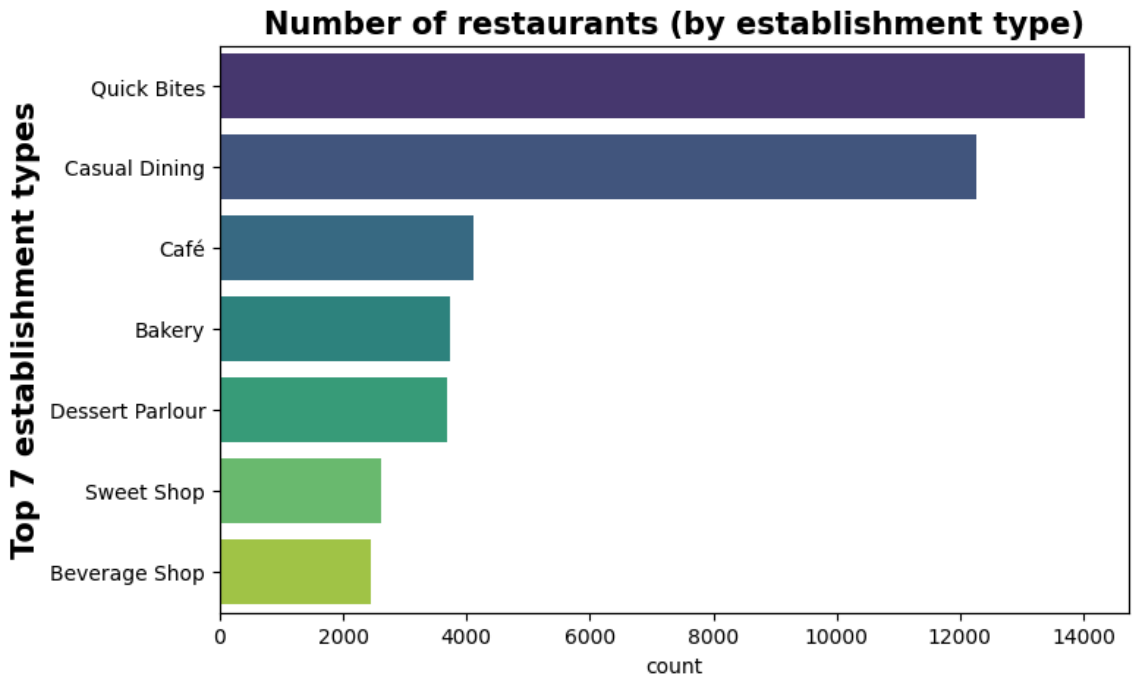
In [ ]:

In [45]:
```
by_establishment = df['establishment'].value_counts().head(7).reset_index()
by_establishment
```

Out[45]:

	establishment	count
0	Quick Bites	14032
1	Casual Dining	12270
2	Café	4123
3	Bakery	3741
4	Dessert Parlour	3675
5	Sweet Shop	2615
6	Beverage Shop	2440

In [46]:
```
plt.figure(figsize=(8,5))

sns.barplot(y='establishment',x='count',data=by_establishment,palette='viri
plt.title('Number of restaurants (by establishment type)',fontsize=15,fontw
plt.ylabel("Top 7 establishment types",fontsize=15,fontweight='bold', color
plt.show()
```



- The graph represents the Restaurants on basis of Establishment type. It shows the count (Number of Restaurants) falling under each category, in this figure we take top 7 establishment types Quick Bites, Casual Dining, Café, Bakery, Dessert Parlour, Sweet shops, Beverage Shop in decreasing order of count.

- You can see that quick Bites having around 14000+ and in Casual Dining having 12000+ are dominating. there is significant drop after casual dining and quick bites more than 66%.
- The graph provides a comparison of different types of restaurant establishments, highlighting the dominance of quick bites and casual dining options.

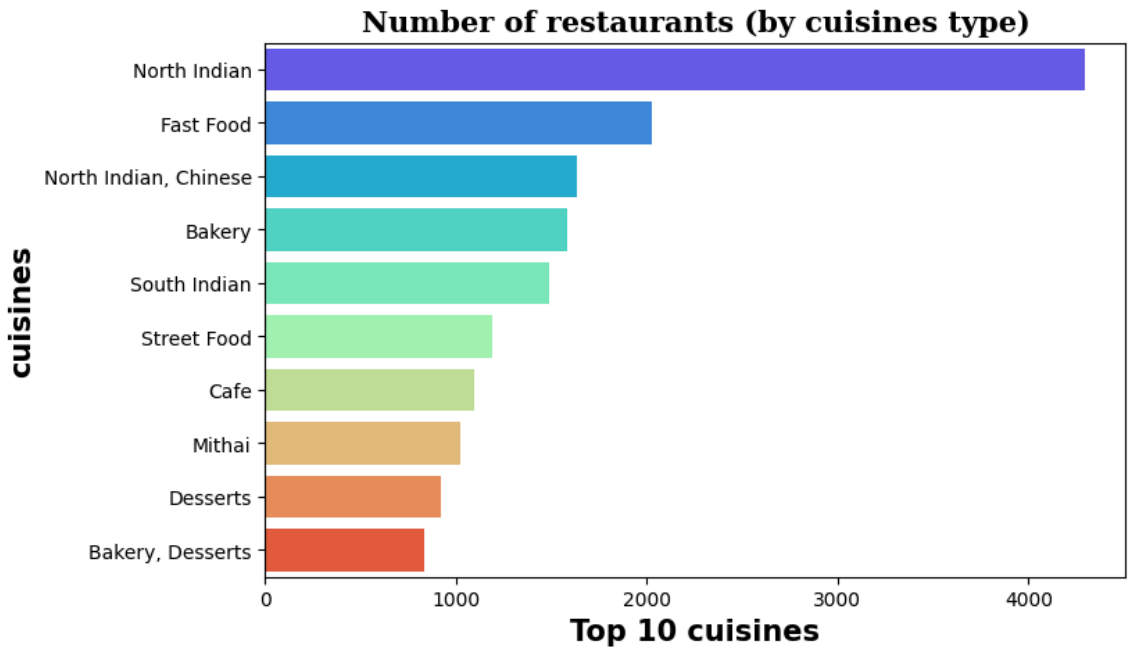
In [ ]:

In [47]:
```
by_cusines = df['cuisines'].value_counts().head(10).reset_index()
by_cusines.head()
```

Out[47]:

	cuisines	count
0	North Indian	4295
1	Fast Food	2025
2	North Indian, Chinese	1636
3	Bakery	1585
4	South Indian	1489

In [48]:
```
plt.figure(figsize=(8,5))
sns.barplot(y='cuisines',x='count',data=by_cusines,palette='rainbow')

plt.title('Number of restaurants (by cuisines type)',fontsize=15,fontweight
plt.xlabel("Top 10 cuisines",fontsize=15,fontweight='bold')
plt.ylabel("cuisines" ,fontsize=15,fontweight='bold')
plt.show()
```



- The bar graph presents number of restaurants categorized by cuisines type they serve.
- The chart shows that North Indian cuisine has the highest number of restaurants around 4500+ which is followed by Fast Food 2000+ and a combination of North Indian 1800+, Chinese cuisines.
- Chinese food comes second in the list of cuisines that Indians prefer,more than fast food, desserts and South Indian food.
- Other notable categories include Bakery, South Indian, Street Food.

In a Zomato dataset, the price range typically indicated by numbers 1, 2, 3, and 4 corresponds to the cost of dining at a restaurant, with each number representing a different price bracket. Here's a general breakdown:

1. **Price Range 1**: Inexpensive (e.g., street food, casual dining)
2. **Price Range 2**: Moderate (e.g., mid-range casual dining)
3. **Price Range 3**: Expensive (e.g., fine dining, premium restaurants)
4. **Price Range 4**: Very expensive (e.g., luxury dining experiences)

In [49]:
```
df['price_range'].unique()
```

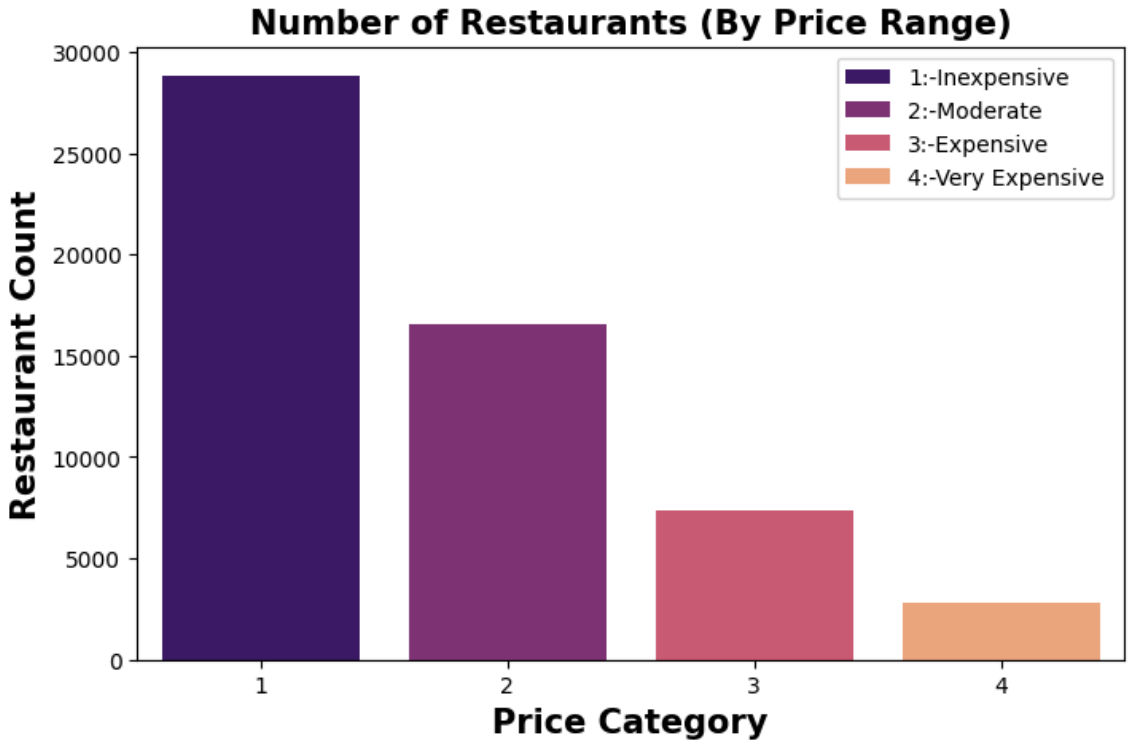
Out[49]:
```
array([2, 1, 3, 4], dtype=int64)
```

In [50]:
```
# df['price_range'].value_counts().reset_index()
```

In [51]:
```
'1:-Inexpensive',
```

Out[51]:
```
('1:-Inexpensive',)
```

In [52]:
```
plt.figure(figsize=(8,5))

sns.countplot(x='price_range',data=df,palette='magma')

plt.title('Number of Restaurants (By Price Range)',fontsize=15,fontweight='
plt.xlabel('Price Category',fontsize=15,fontweight='bold')
plt.ylabel('Restaurant Count',fontsize=15,fontweight='bold')

plt.legend(['1:-Inexpensive','2:-Moderate','3:-Expensive','4:-Very Expensiv
plt.show()
```



In [53]:
```
df.columns
```

Out[53]:
```
Index(['name', 'establishment', 'url', 'address', 'city', 'city_id',
       'locality', 'latitude', 'longitude', 'zipcode', 'country_id',
       'locality_verbose', 'cuisines', 'timings', 'average_cost_for_two',
       'price_range', 'currency', 'highlights', 'aggregate_rating',
       'rating_text', 'votes', 'photo_count', 'opentable_support', 'delive
ry',
       'takeaway'],
      dtype='object')
```

In [54]:
```
df[df['price_range'] == 1]['average_cost_for_two'].max()
```

Out[54]:
```
450
```

In [55]:
```
#Lets Find Price Range values
p1max = df[df['price_range'] == 1]['average_cost_for_two'].max()
p2max = df[df['price_range'] == 2]['average_cost_for_two'].max()
p3max = df[df['price_range'] == 3]['average_cost_for_two'].max()
p4max = df[df['price_range'] == 4]['average_cost_for_two'].max()



p1min = df[df['price_range'] == 1]['average_cost_for_two'].min()
p2min = df[df['price_range'] == 2]['average_cost_for_two'].min()
p3min = df[df['price_range'] == 3]['average_cost_for_two'].min()
p4min = df[df['price_range'] == 4]['average_cost_for_two'].min()

print(p1max,p2max,p3max,p4max)
print(p1min,p2min,p3min,p4min)
```

```
450 999 1900 30000
0 250 500 1000
```

In [56]:
```
#minimum Price for category assume Rs.50
prices=pd.DataFrame({ 'Price category':[1,2,3,4],
                'Price Range':['50-450','250-999','500-1900','1000-30000']},i
prices
```

Out[56]:

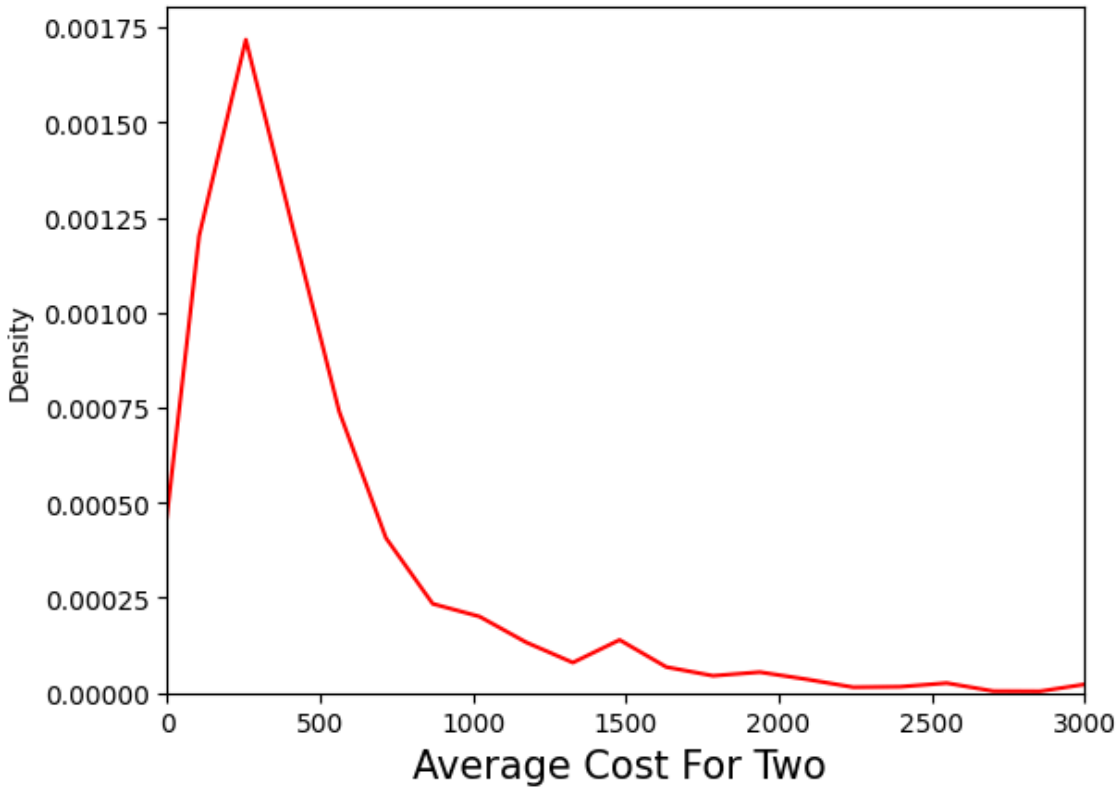
	Price category	Price Range
1	1	50-450
2	2	250-999
3	3	500-1900
4	4	1000-30000

- The graph displays a number of Restaurants categorized by Price Range Price category 1 (50-450) has the highest number of restaurants among the 4 categories around 28000+ restaurants are in category one price range .it shows the cosumer spending in restaurants all over the India.
- As you can see the sharp drop between category 1 and 2,3,4
- As the price range increases, the number of restaurants decreases, its indicating that fewer consumers are willing or able to spend higher amounts on dining out.
- The relatively small number of restaurants in the highest price range of 1,000-30,000 (category 4) implies that only a niche segment of consumers is willing to spend

exorbitant amounts on dining experiences.

- Over all this graph shows the consumer spending habits in the restaurants, 50% of the restaurants are in category 1. majority of consumers gravitating towards more affordable dining options

In [57]:
```
sns.kdeplot(df["average_cost_for_two"],color = 'red')
plt.xlim(0,3000)
plt.xlabel('Average Cost For Two ',fontsize=15)
plt.show()
```



This KDE plot shows the distribution of average restaurant costs for two people. The single peak around 500- 600 indicates that most restaurants fall within that mid-range pricing for a meal for two. It means that most of the consumer prefer restaurant with mid range price.

In [ ]:

In [ ]:

In [58]:
```
luxury=df.groupby(['name','establishment','city'])['average_cost_for_two'].
luxury
```

Out[58]:

	name	establishment	city	average_cost_for_two
0	Ocean - The Private Dining Room - Sahara Star	Fine Dining	Mumbai	30000
1	Gol Bungalow - Taj Falaknuma Palace	Fine Dining	Hyderabad	15000
2	Bhairo	Fine Dining	Udaipur	15000
3	Fly Dining	Fine Dining	Bangalore	14000
4	Trophy Bar- Umaid Bhawan Palace	Bar	Jodhpur	12000
5	Pillars - Umaid Bhawan Palace	Fine Dining	Jodhpur	12000
6	Risala- Umaid Bhawan Palace	Fine Dining	Jodhpur	12000
7	Wasabi By Morimoto - The Taj Mahal Palace	Fine Dining	Mumbai	10000
8	Whiskys - Deltin Royale	Bar	Goa	8000
9	Adaa - Taj Falaknuma Palace	Fine Dining	Hyderabad	8000

- This Table shows the Top 10 most Expensive Restaurants with average cost for two
- The difference between First and second most expensive restaurant is twice **2x**
- As we can see that the most expensive restaurant is ocean-The Private Dining Room sahara located in Mumbai.
- 4 restaurants out of 10 is located in Rajasthan one in Udaipur and three in **Jodhpur**
- On the basis of establishment types 8 are the fine dining and rest 2 is bar type.

In [59]:
```
df[df['name']=='Bhairo']
```

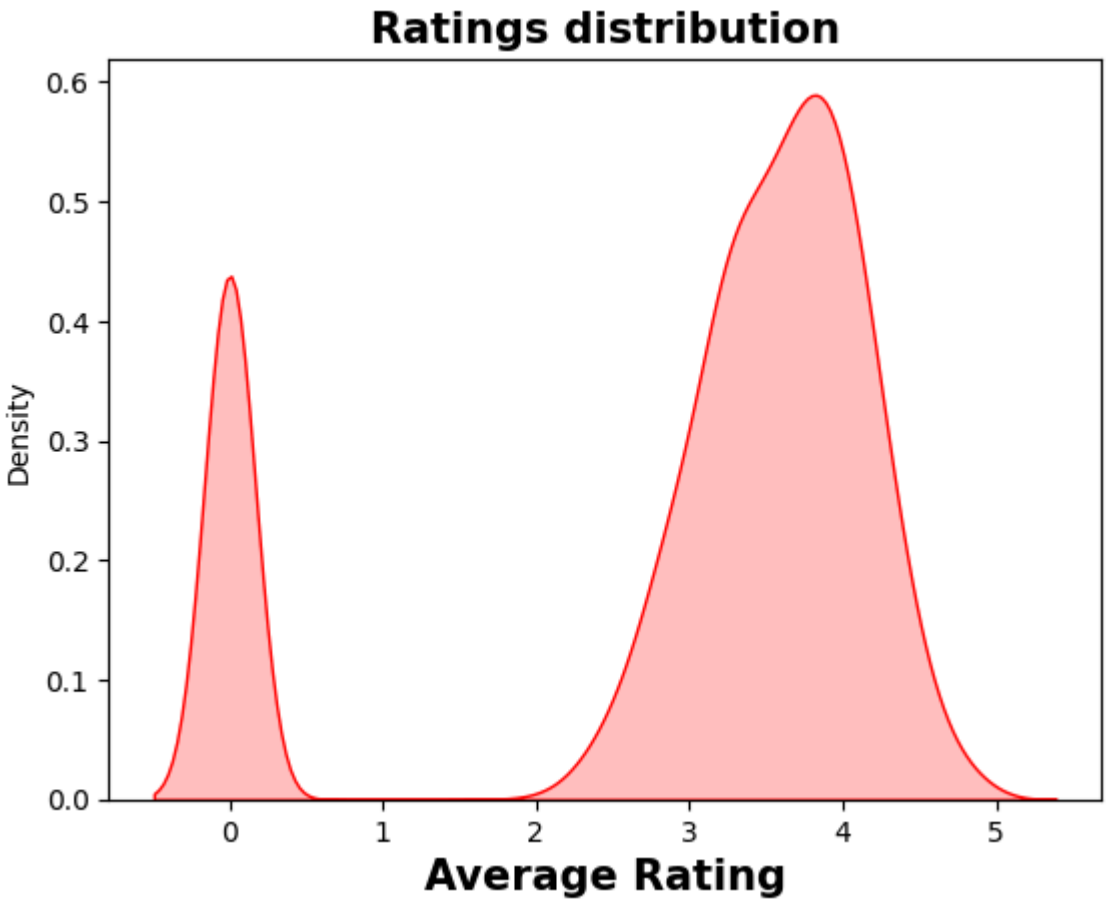
Out[59]:

	name	establishment	url	address	city	city_i
res_id						
18565535	Bhairo	Fine Dining	https://www.zomato.com/udaipur/bhairo-pichola?...	Taj Lake Palace, Pichola, Udaipur	Udaipur	1105

1 rows × 25 columns

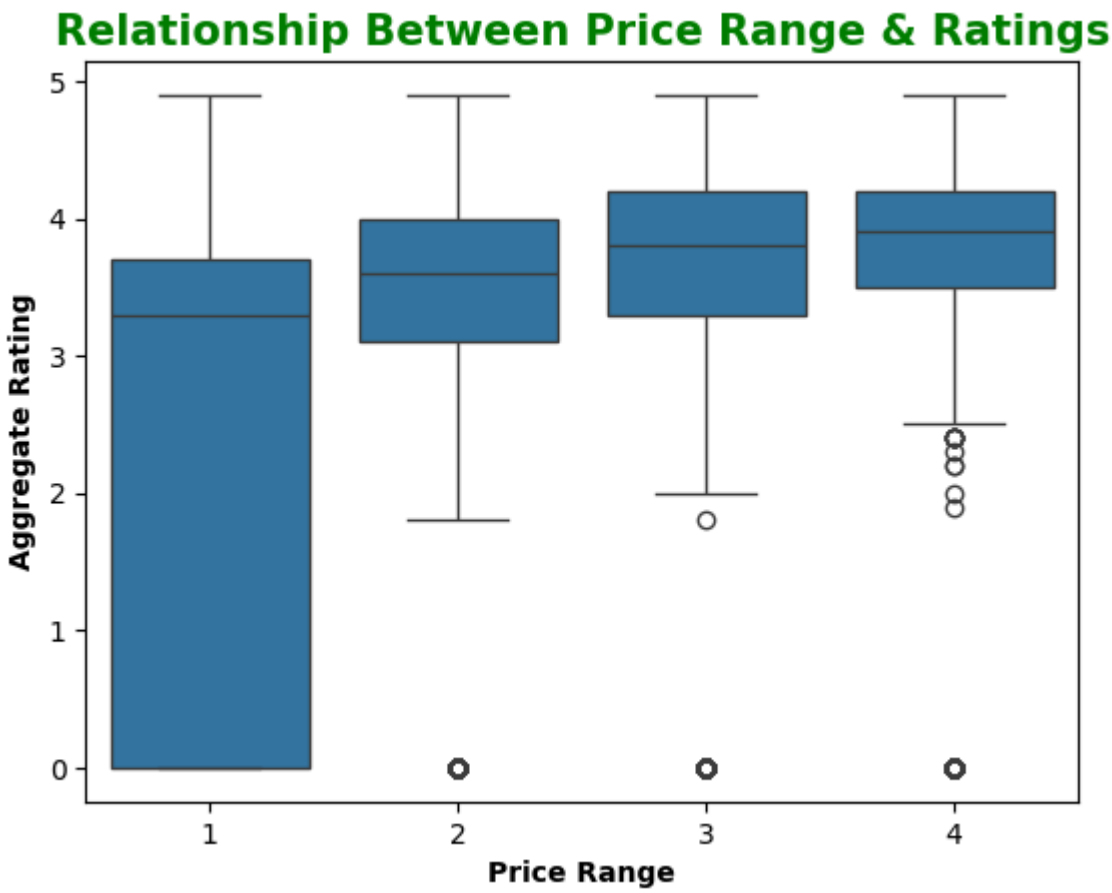
In [ ]:

In [60]:
```
sns.kdeplot(df['aggregate_rating'], shade=True,color='r')

plt.title("Ratings distribution",fontsize=15,fontweight='bold')
plt.xlabel("Average Rating",fontsize=15,fontweight='bold')
plt.show()
```



- There is a spike at 0 which might account for newly opened or unrated restaurants.
- most of the restaurant is rated betweeen 3 to 4
- On average, majority of restaurants have rating between 3 to 4 with fewer restaurants managing to go beyond 4

In [61]:
```
sns.boxplot(x='price_range', y='aggregate_rating', data=df)

plt.title("Relationship Between Price Range & Ratings",fontsize=15,fontweig
plt.xlabel("Price Range",fontsize=10,fontweight='bold')
plt.ylabel("Aggregate Rating",fontsize=10,fontweight='bold')
plt.show()
```



In [62]:
```
# df['aggregate_rating']
```

- This Boxplot Graph shows restaurants in the lowest price category 1 (50-450) have the highest average rating , around 3.5 on the rating scale
- As the price range increases from category 1 to category 2, the average rating declines slightly but highest rating touch 4 on the rating scale.
- Restaurants in the highest price range (category 4) have the lowest average rating, with the biggest variation in ratings as indicated by the larger box plot whiskers.
- As the price range increases, the average rating is decreasing.

In [63]:
```
def check(input_list):
    if 'Delivery' in input_list:
        return 'Delivery Available'

    elif 'Takeaway Available' in input_list:
        return 'Takeaway Available'

    else:
        return 'No Delivery Available'
```

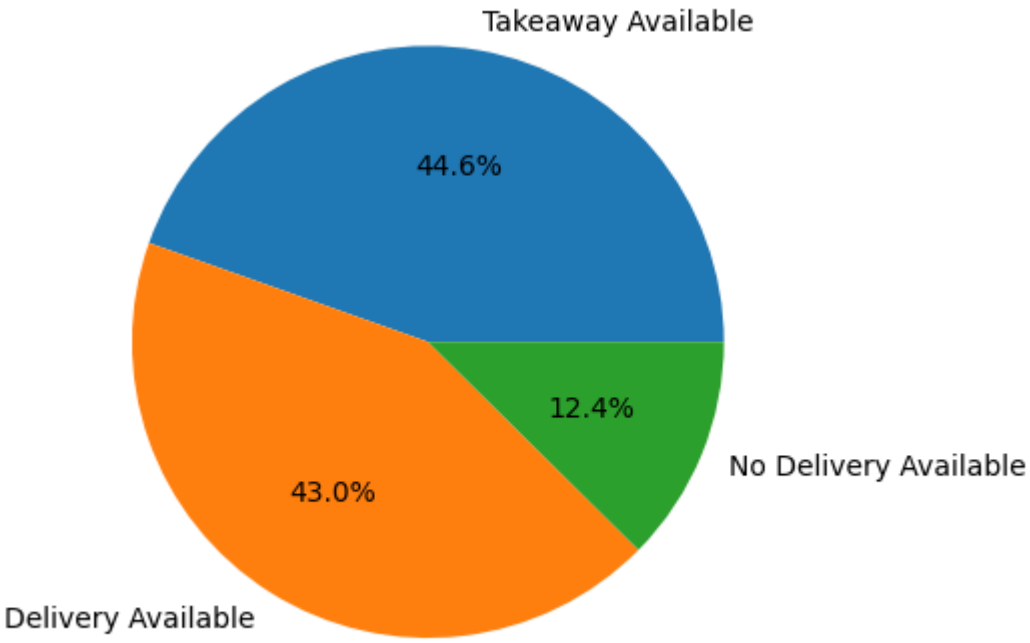
In [64]:
```
ff=df['highlights'].apply(check).value_counts().reset_index()
```

In [65]:
```
ff
```

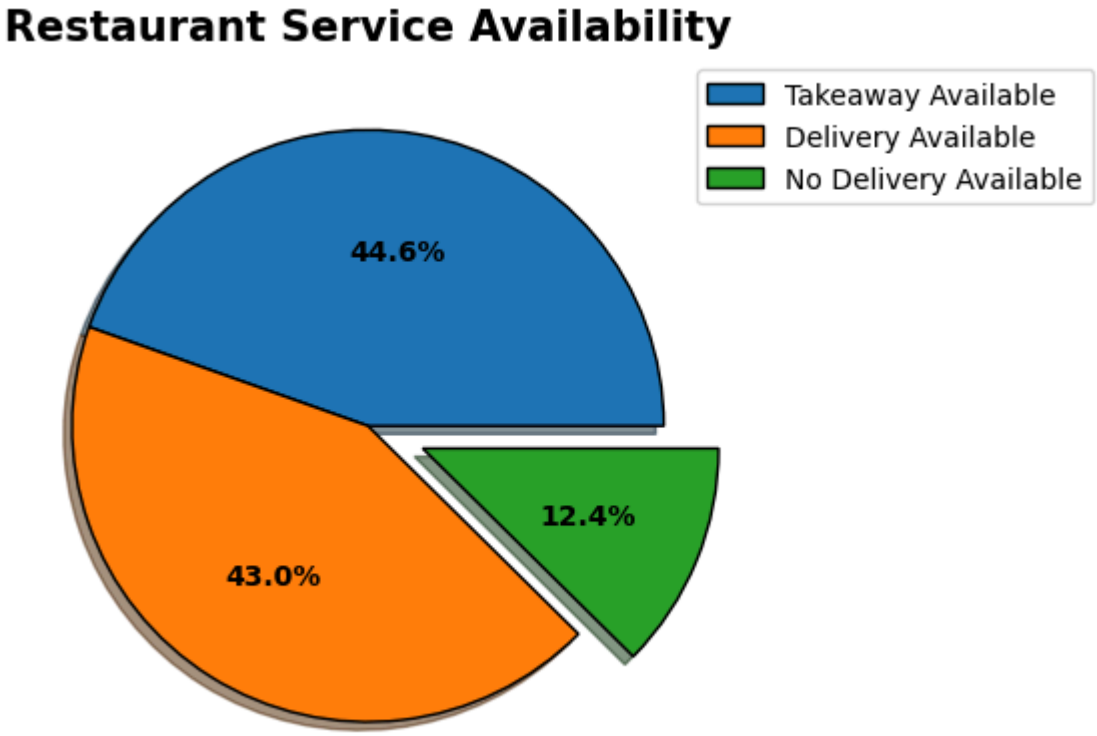
Out[65]:

	highlights	count
0	Takeaway Available	24769
1	Delivery Available	23907
2	No Delivery Available	6892

In [66]:
```
plt.pie(ff['count'],
        labels = ff['highlights'],
        autopct='%.1f%%')

plt.show()
```



In [67]:
```
ex=[0.0,0.0,0.2]

plt.pie(ff['count'],
        explode=ex,
        autopct="%1.1f%%",
        shadow=True,
        radius=1,

        textprops={"fontsize": 10,"fontweight":'bold'},

        wedgeprops={"linewidth":1,"edgecolor" :'black'},
        rotatelabels=False)

plt.title('Restaurant Service Availability',fontsize=15,fontweight='bold')
plt.legend(labels=ff['highlights'],loc='upper right',bbox_to_anchor=(1.5,1)
plt.show()
```



- The pie chart shows the Restaurant Service Availability. the pie chart is divided ito three slices.
- “Take away Availability”, “Delivery Available” & “no delivery Available”.
- the slice labelled with “Take away Availability” accounts for 44.6% of restaurants. This shows that almost half of the restaurants offering takeaway service.
- The slice labelled with “Delivery Available” accounts for 43.0% of restaurants. This shows that almost half of the restaurants provide Delivery services.
- The slice labelled with “no delivery Available” accounts for 12.4% of restaurants. Are no providing takeaways and delivery services maybe these restaurants are in the category of most expensive restaurants.

wordcloud Example

In [68]:
```
# pip install wordcloud
```

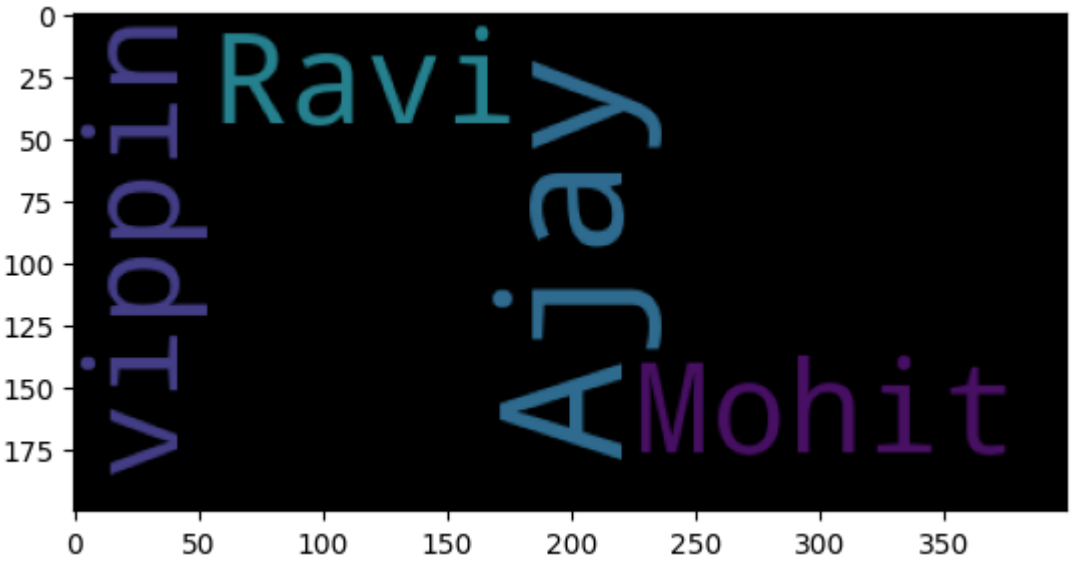
In [69]:
```
from wordcloud import WordCloud
print('done')
```

```
done
```

In [70]:
```
# ' '.join(df['highlights'])
```

In [71]:
```
data = 'Ajay Ajay vippin Mohit Ravi'

plt.imshow(WordCloud().generate(data))
```

Out[71]:
```
<matplotlib.image.AxesImage at 0x17da4c68e90>
```



In [72]:

```
# Sample DataFrame with text data

# Combine text data into a single string
text = ' '.join(df['highlights'])

# Generate word cloud
wc = WordCloud(width=800, height=400, background_color='white').generate(te

# Display the word cloud
#plt.figure(figsize=(10, 5))
plt.imshow(wc)
plt.title('Highlighted Words',fontsize=10,fontweight='bold')
plt.axis('off')
plt.show()
```

Word Cloud is the visual representation of words which show most frequently used ones and highlighted those words the word cloud. It is used for to identifying key topics, data Exploration The above Word cloud shows the highlighted words like Takeaway, Debit card, Indoor Seating, Cash, debit card, credit card, pure veg. outdoor seating Over all its highlight the variety of services offered by the restaurant to make it a convenient dining option for customers.

Limitations

- **Limited Geographic Coverage**: The dataset is covering the only major cities of India. but its reach in smaller towns and rural areas is limited. Its not cover rural areas the Data set mainly focus on urban and big cities
- **Incomplete Data**: some columns like zip code, cuisines types, timings of restaurants had missing which impact the accuracy of analysis.
- **Outdated Data**: The Dataset is made by Web scrapping and Zomato API and its outdated. Which not reflected the new restaurants, cities it also affects the accuracy of analysis
- **Lack of Additional Features**: The data set have lack of additional features we have no deeper information like restaurant menu from which we can identify most ordered dish.

Future Scope

- **Incorporation of Customer Reviews**: Integrating customer reviews and sentiment analysis help to provide valuable information about the customer preference, satisfaction level which helps the company to make decisions.
- **Business Expansion**: Extending the data collection to other region allows the comparison and facilitate a understanding about restaurant industry in India.
- **Predictive Analysis**: Developing predictive models based on the dataset can help the restaurant owners and entrepreneurs to identify the opportunities in restaurant industry by predictive analysis helps to take decision regarding restaurant location, menu, cuisine offering.
- **Incorporation of Online Ordering and Delivery Data**: The Dataset helps to enhancing delivery service system through analysis. we can find out from which region the maximum orders are coming.

Conclusion

- Analyzed the dataset to uncover insights about the restaurant landscape in India.
- Identified the top 20 cities and most popular cuisines.
- Examined various price categories and their relationship with restaurant types.
- Highlighted the most expensive restaurants.
- Analyzed service availability, including takeaway and delivery options.
- Provided a comprehensive understanding of the Indian restaurant industry.
- Aided str- ategic decision-making for stakeholders.
- Revealed significant patterns and trends regarding geographic distribution, establishment types, cuisines, and pricing structures.

DS Trainer:- Ankit Mishra

In [ ]: