

Project Python Foundations: FoodHub Data Analysis

Marks: 60

Context

The number of restaurants in New York is increasing day by day. Lots of students and busy professionals rely on those restaurants due to their hectic lifestyles. Online food delivery service is a great option for them. It provides them with good food from their favorite restaurants. A food aggregator company FoodHub offers access to multiple restaurants through a single smartphone app.

The app allows the restaurants to receive a direct online order from a customer. The app assigns a delivery person from the company to pick up the order after it is confirmed by the restaurant. The delivery person then uses the map to reach the restaurant and waits for the food package. Once the food package is handed over to the delivery person, he/she confirms the pick-up in the app and travels to the customer's location to deliver the food. The delivery person confirms the drop-off in the app after delivering the food package to the customer. The customer can rate the order in the app. The food aggregator earns money by collecting a fixed margin of the delivery order from the restaurants.

Objective

The food aggregator company has stored the data of the different orders made by the registered customers in their online portal. They want to analyze the data to get a fair idea about the demand of different restaurants which will help them in enhancing their customer experience. Suppose you are hired as a Data Scientist in this company and the Data Science team has shared some of the key questions that need to be answered. Perform the data analysis to find answers to these questions that will help the company to improve the business.

Data Description

The data contains the different data related to a food order. The detailed data dictionary is given below.

Data Dictionary

- order_id: Unique ID of the order
- customer_id: ID of the customer who ordered the food

- restaurant_name: Name of the restaurant
- cuisine_type: Cuisine ordered by the customer
- cost: Cost of the order
- day_of_the_week: Indicates whether the order is placed on a weekday or weekend (The weekday is from Monday to Friday and the weekend is Saturday and Sunday)
- rating: Rating given by the customer out of 5
- food_preparation_time: Time (in minutes) taken by the restaurant to prepare the food. This is calculated by taking the difference between the timestamps of the restaurant's order confirmation and the delivery person's pick-up confirmation.
- delivery_time: Time (in minutes) taken by the delivery person to deliver the food package. This is calculated by taking the difference between the timestamps of the delivery person's pick-up confirmation and drop-off information

Let us start by importing the required libraries

```
In [1]: # import libraries for data manipulation
import numpy as np
import pandas as pd

# import libraries for data visualization
import matplotlib.pyplot as plt
import seaborn as sns
```

Understanding the structure of the data

```
In [2]: # uncomment and run the following lines for Google Colab
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

Loading the dataset

```
In [3]: # read the data
data = pd.read_csv("/content/drive/MyDrive/PythonProject/foodhub_order.csv")

In [4]: # copying data to another variable to avoid any changes to original data
df = data.copy()
```


Data Overview

Displaying the first few rows of the dataset

```
In [5]: # Looking at head (5 observations)
df.head()
```

Out[5]:

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week
0	1477147	337525	Hangawi	Korean	30.75	Weekend
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday



Observations:

- The DataFrame has 9 columns as mentioned in the Data Dictionary. Data in each row corresponds to the order placed by a customer.

Question 1: How many rows and columns are present in the data? [0.5 mark]

In [6]: `# check the shape of the dataset`
`df.shape`

Out[6]: (1898, 9)

Observations:

- There are 1898 rows and 9 columns in dataset.

Question 2: What are the datatypes of the different columns in the dataset? (The info() function can be used) [0.5 mark]

In [8]: `# Use info() to print a concise summary of the DataFrame`
`df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1898 entries, 0 to 1897
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   order_id              1898 non-null   int64
1   customer_id           1898 non-null   int64
2   restaurant_name       1898 non-null   object
3   cuisine_type          1898 non-null   object
4   cost_of_the_order     1898 non-null   float64
5   day_of_the_week       1898 non-null   object
6   rating                1898 non-null   object
7   food_preparation_time 1898 non-null   int64
8   delivery_time         1898 non-null   int64
dtypes: float64(1), int64(4), object(4)
memory usage: 133.6+ KB
```

Observations:

- All the columns have 1898 observations. There is not any missing values dataset.
- The `food_preparation_time` and `delivery_time` columns show time in int datatype. The `cost_of_the_order` columns have float data type. `Order_id` and `customer_id` are in int datatype.
- The object type columns contain categories in them.
- Total memory usage is approximately 133.6 KB

Question 3: Are there any missing values in the data? If yes, treat them using an appropriate method. [1 mark]

```
In [9]: # checking missing values across each columns
df.isnull().sum()
```

```
Out[9]:
```

	0
order_id	0
customer_id	0
restaurant_name	0
cuisine_type	0
cost_of_the_order	0
day_of_the_week	0
rating	0
food_preparation_time	0
delivery_time	0

dtype: int64

Observations:

- *There are no missing values in dataset.*

```
In [10]: df.duplicated().sum()
```

```
Out[10]: np.int64(0)
```

Observations:

- *There are no duplicated rows in dataset.*

Question 4: Check the statistical summary of the data. What is the minimum, average, and maximum time it takes for food to be prepared once an order is placed? [2 marks]

```
In [11]: # get the summary statistics of the numerical data
df.describe().T.round(2)
```

Out[11]:

	count	mean	std	min	25%	50%
order_id	1898.0	1477495.50	548.05	1476547.00	1477021.25	1477495.50
customer_id	1898.0	171168.48	113698.14	1311.00	77787.75	128600.00
cost_of_the_order	1898.0	16.50	7.48	4.47	12.08	14.14
food_preparation_time	1898.0	27.37	4.63	20.00	23.00	27.00
delivery_time	1898.0	24.16	4.97	15.00	20.00	25.00

Observations:

1. *Order_id* : *Order_id* is the unique identification of customers given orders in the dataset.
2. *Customer_id* : *Customer_id* is the unique identification of customers in the dataset. *Order ID* and *Customer ID* are just identifiers for each order.
3. *Cost_of_the_order* : The cost of the order ranges from 4.47 to 35.41. 75% of the cost of the order is less than 23 dollars. People don't prefer to spend much money on food. A chunk of people prefer to spend money on expensive food dishes.
4. *Food_preparation_time* : The food preparation time ranges from 20 to 35 minutes. The mean is 27 minutes, and 75% is 31 minutes. Restaurants take a lot of time for food preparation.
5. *Delivery_time* : The delivery time ranges from 15 to 33 minutes. The mean is 24 minutes, and 75% is 28 minutes. delivery time is much.
6. More analyses are needed on cost of the order, food preparation time and delivery time etc...

Question 5: How many orders are not rated? [1 mark]

```
In [12]: #write code for value count
df['rating'].value_counts()['Not given']
```

```
Out[12]: np.int64(736)
```

```
In [13]: #write code for value count with percentile.
df['rating'].value_counts(normalize=True)['Not given'].round(2)
```

```
Out[13]: np.float64(0.39)
```

Observations:

- *There are 736 (39%) orders that are not rated.*

More analysis is needed on rating column.

Exploratory Data Analysis (EDA)

Univariate Analysis

Question 6: Explore all the variables and provide observations on their distributions. (Generally, histograms, boxplots, countplots, etc. are used for univariate exploration.) [9 marks]

Order_ID Column

```
In [14]: # check unique order ID
df['order_id'].nunique()
```

Out[14]: 1898

Observation :

- *There are 1898 unique order id.*

Customer_id Column

```
In [15]: # check unique customer ID
df['customer_id'].nunique()
```

Out[15]: 1200

Observations :

- *FoodHub have 1200 customers. Each customer has own unique id number. Order id is customers given orders numbers.*

Restaurent_name column :

```
In [16]: # check unique restaurant name
df['restaurant_name'].nunique()
```

Out[16]: 178

```
In [17]: # importing plotly
import plotly.express as px
# Create plotly histogram plot for restaurent name column
```

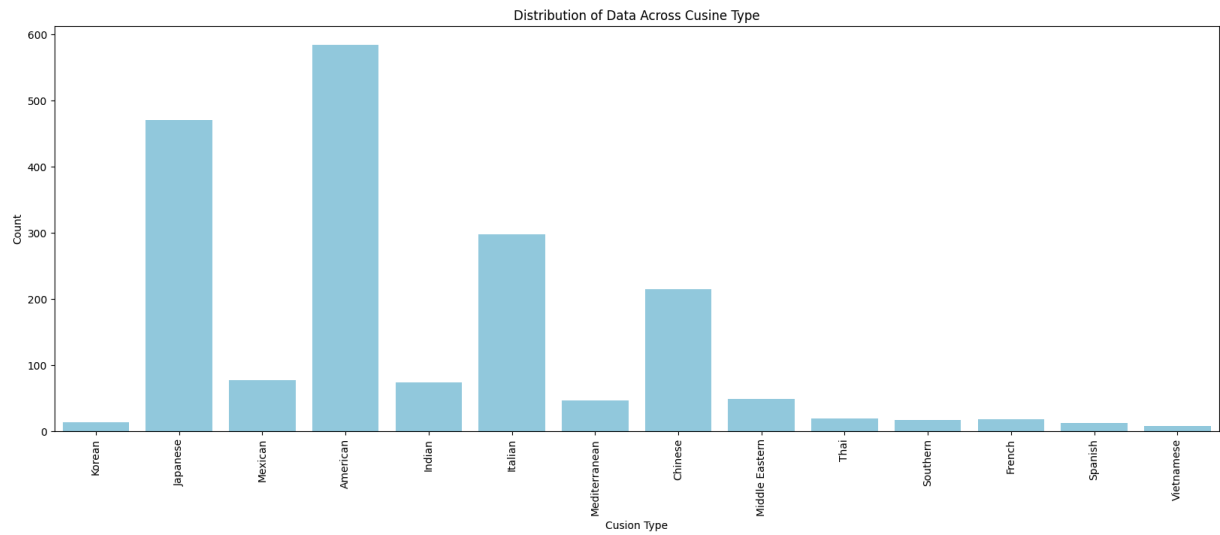
```
his = px.histogram(df, x="restaurant_name")  
# Show plot  
his.show()
```

Observations:

- There are 178 unique restaurants.
- The Shake Shack restaurant has received maximum number of orders.

Cuisine_type column

```
In [18]: # write code for figure size  
plt.figure(figsize=(20,7))  
plt.xlabel('Cusion Type')  
plt.ylabel('Count')  
plt.title('Distribution of Data Across Cusine Type')  
# Rotate x-axis tick labels by 90 degrees  
plt.xticks(rotation=90)  
# Create the plot for 'cuisine_type' column.  
sns.countplot(data=df, x='cuisine_type', color='skyblue')  
# Display the plot  
plt.show()
```

```
In [19]: df['cuisine_type'].value_counts(normalize=True)
```

Out[19]: **proportion**

cuisine_type	
American	0.307692
Japanese	0.247629
Italian	0.157007
Chinese	0.113277
Mexican	0.040569
Indian	0.038462
Middle Eastern	0.025817
Mediterranean	0.024236
Thai	0.010011
French	0.009484
Southern	0.008957
Korean	0.006849
Spanish	0.006322
Vietnamese	0.003688

dtype: float64

Observation :

- *Americen cuisine is the most popular among customers followed by Japanese, Italian cusines.*

- 80% orders get from American, Japanese, Italian and Chinese cuisines.

Day of the week column

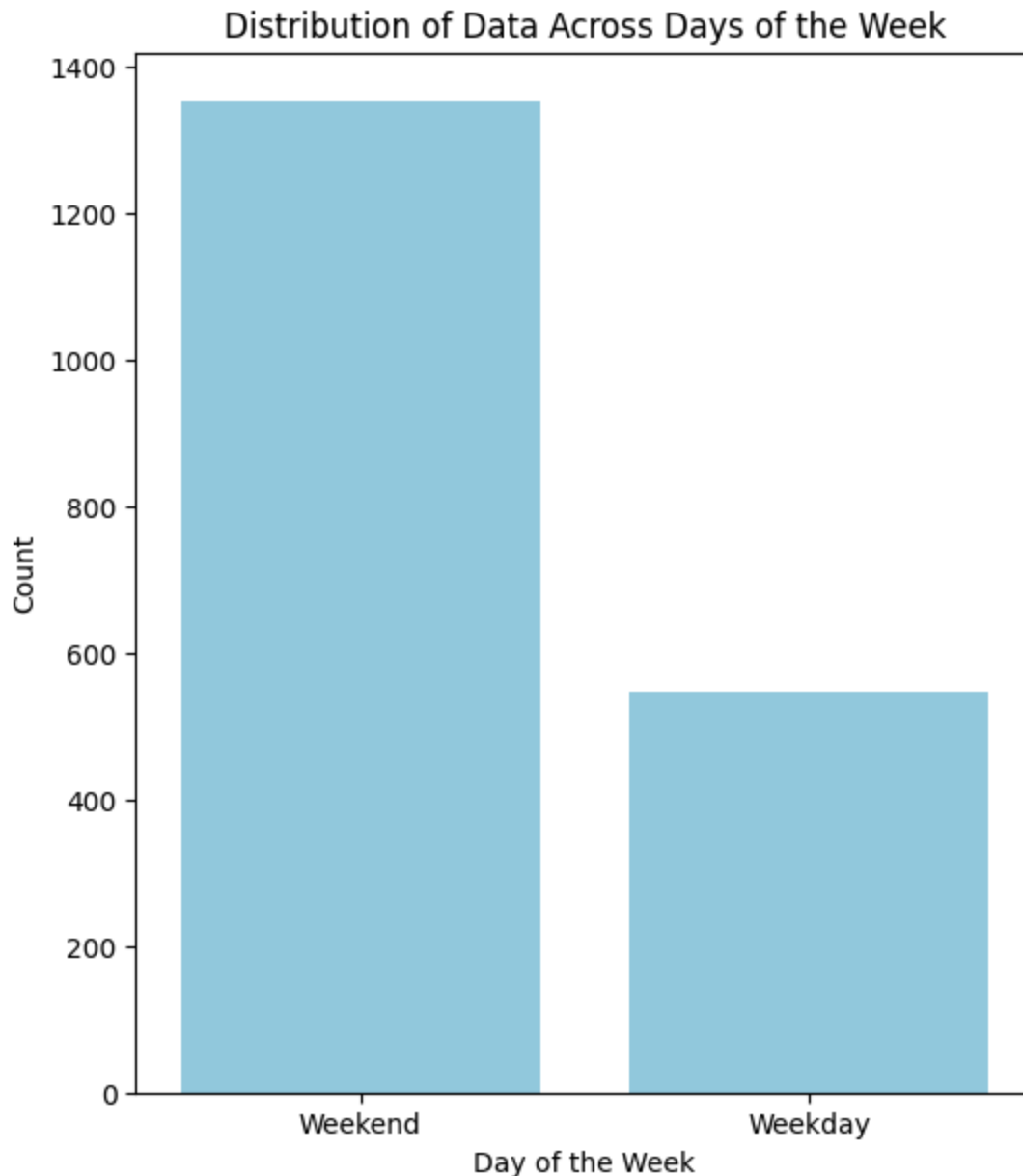
```
In [20]: df['day_of_the_week'].value_counts(normalize=True)
```

```
Out[20]:
```

	proportion
day_of_the_week	
Weekend	0.711802
Weekday	0.288198

dtype: float64

```
In [21]: # write code for figure size
plt.figure(figsize=(6,7))
# Adding labels and title for better readability
plt.xlabel('Day of the Week')
plt.ylabel('Count')
plt.title('Distribution of Data Across Days of the Week')
# Create countplot for 'day_of_the_week' column.
sns.countplot(data=df, x='day_of_the_week', color='skyblue')
# Display the plot
plt.show()
```

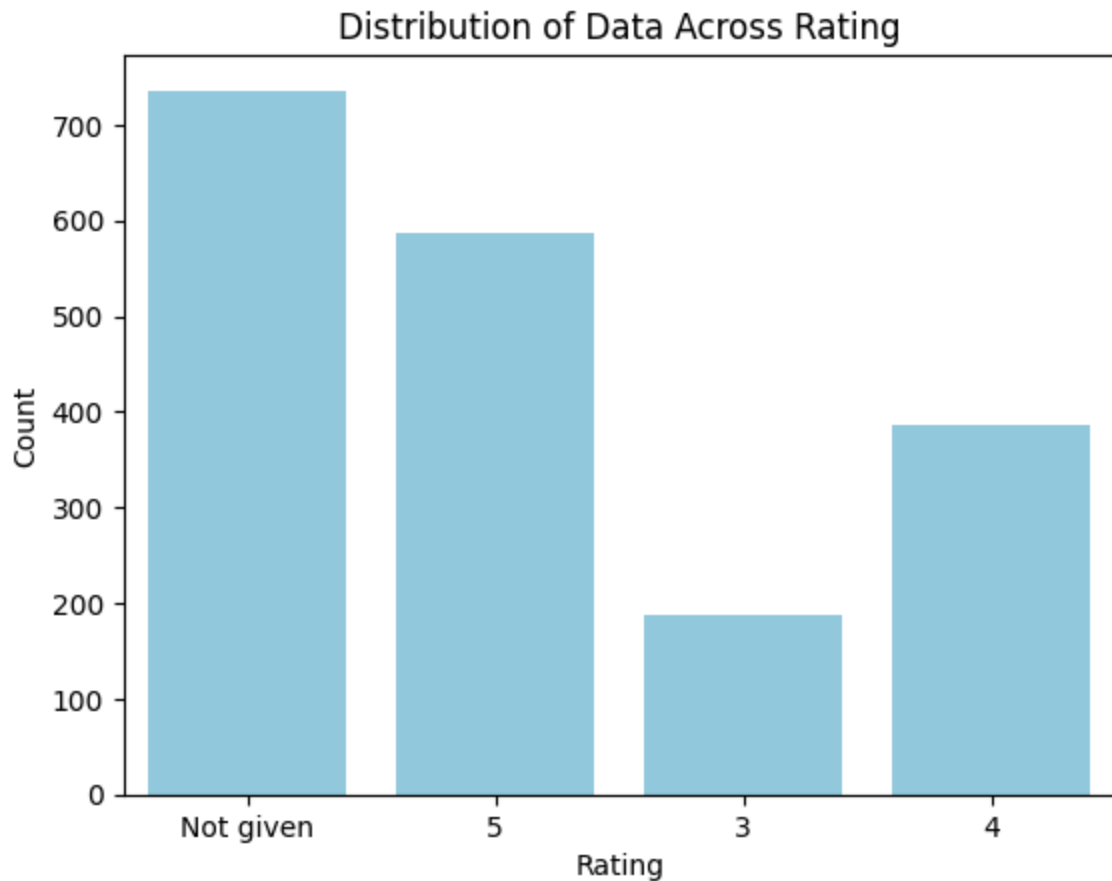


Observations:

- 71% Of orders are received in weekends.
- 29% of oredrs are received in weekdays.

Rating column

```
In [22]: # Adding Labels and title for better readability
plt.xlabel('Rating')
plt.ylabel('Count')
plt.title('Distribution of Data Across Rating')
# Create the plot for 'day_of_the_week' column.
sns.countplot(data=df, x='rating', color='skyblue')
# Display the plot
plt.show()
```



```
In [23]: df['rating'].value_counts(normalize=True)
```

```
Out[23]:
```

	proportion
--	------------

rating	
Not given	0.387777
5	0.309800
4	0.203372
3	0.099052

dtype: float64

Observations:

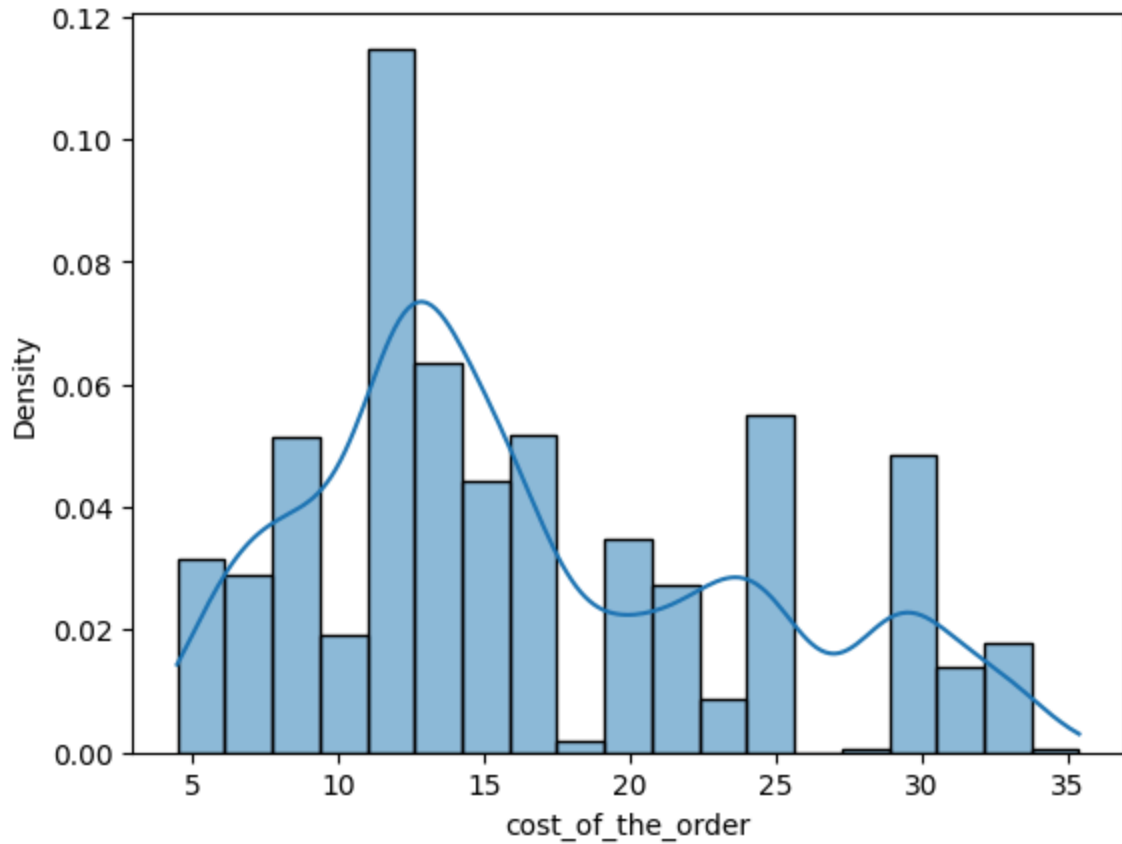
- Customers do not prefer to give less rating.
- 31% of Customers have given highest 5 rating. and 20% of customers have given 4 rating.

Cost of the order, Food preparation time and Delivery time Numerical columns

```
In [24]: # Making a list of all numerical variables
numerical_cols = ['cost_of_the_order', 'food_preparation_time', 'delivery_time']
```

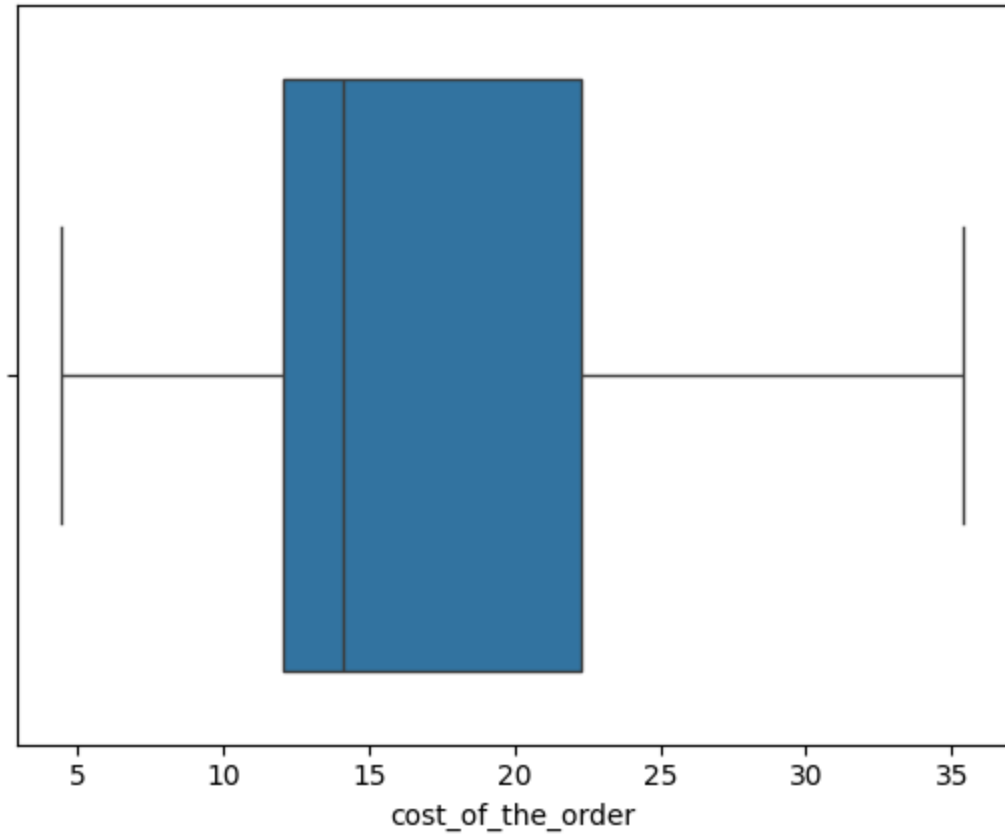
```
# Loop through numerical columns and create plots
for column in numerical_cols:
    print(sns.histplot(data=df, x=column, stat='density', kde=True)) # Use df, not
    print(plt.show())
    print(sns.boxplot(data=df, x=column)) # Use df, not numerical_cols
    print(plt.show())
```

Axes(0.125,0.11;0.775x0.77)



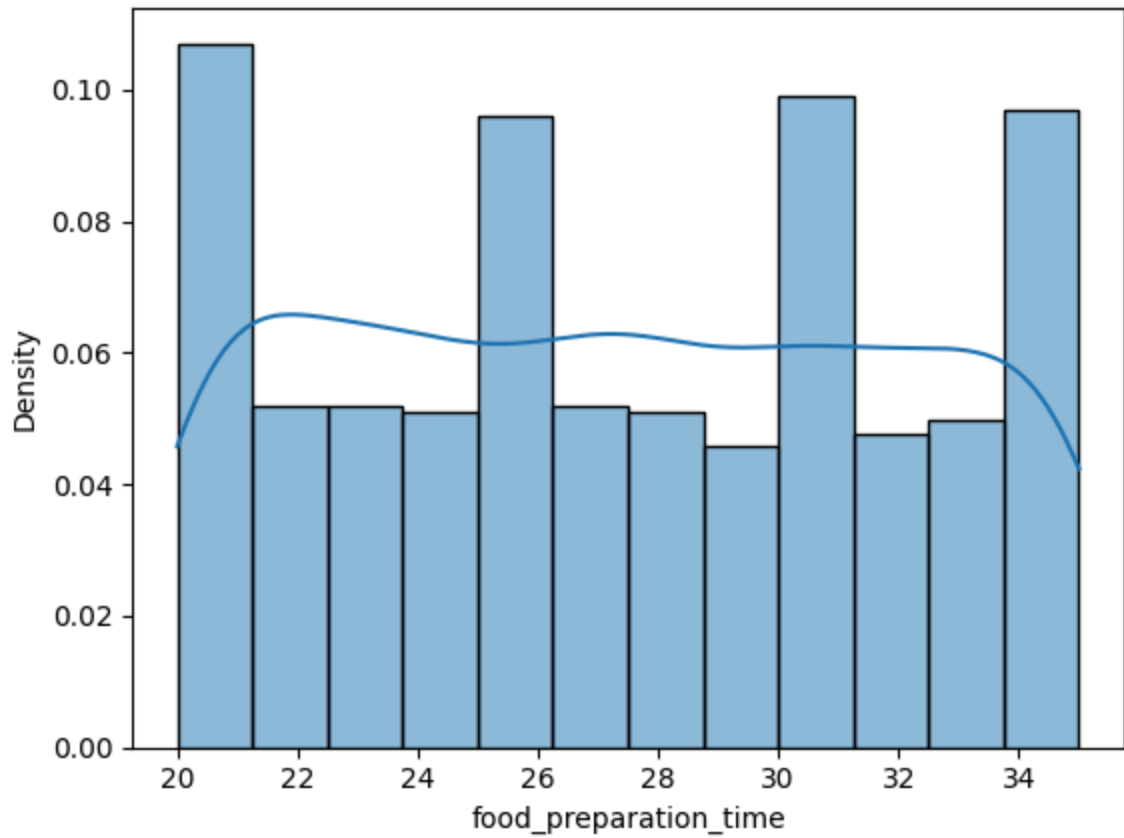
None

Axes(0.125,0.11;0.775x0.77)



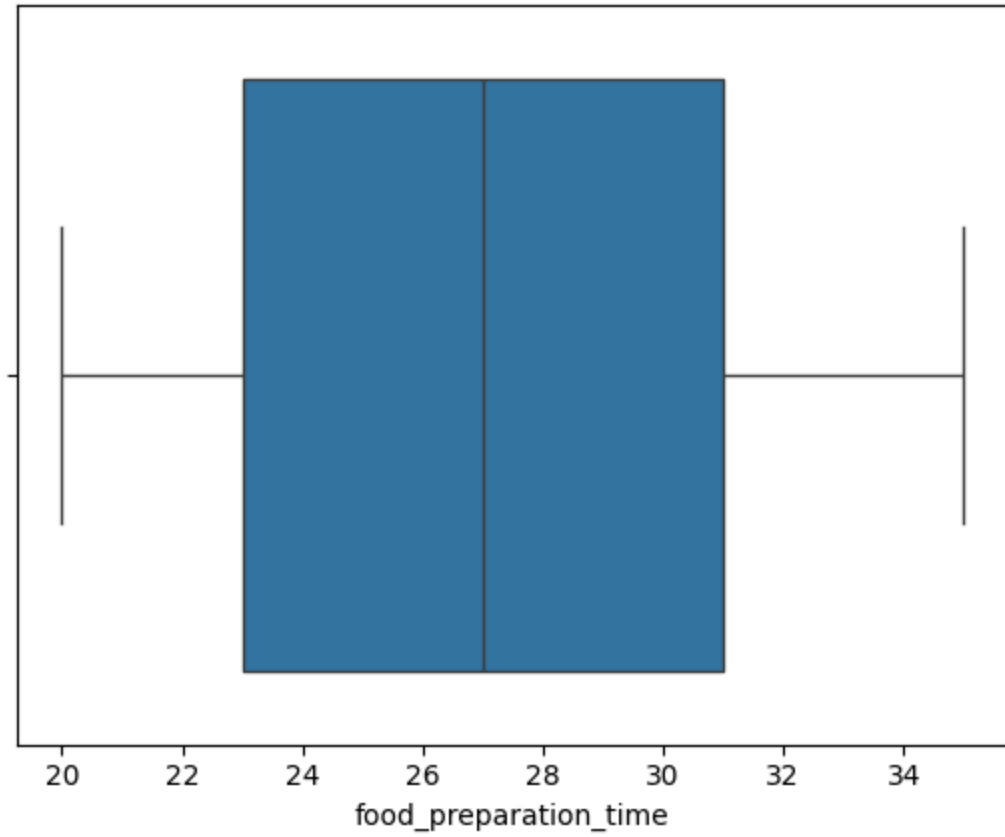
None

Axes(0.125,0.11;0.775x0.77)

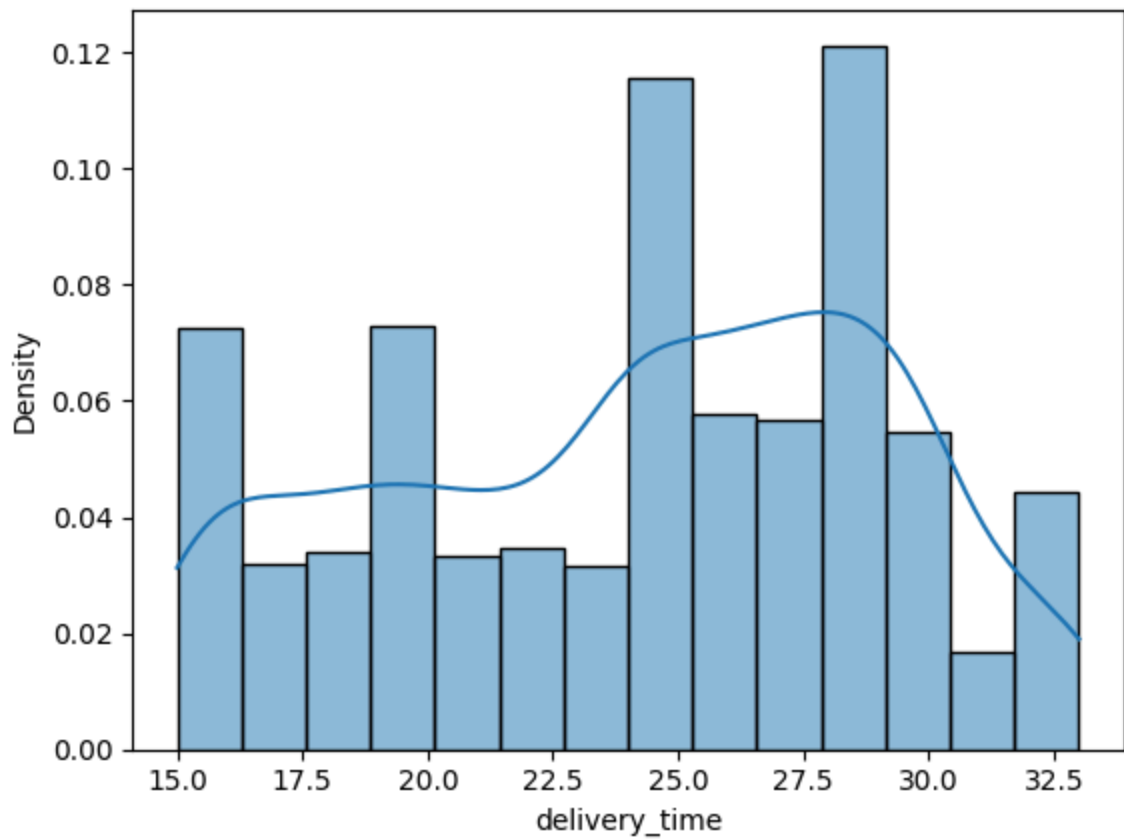


None

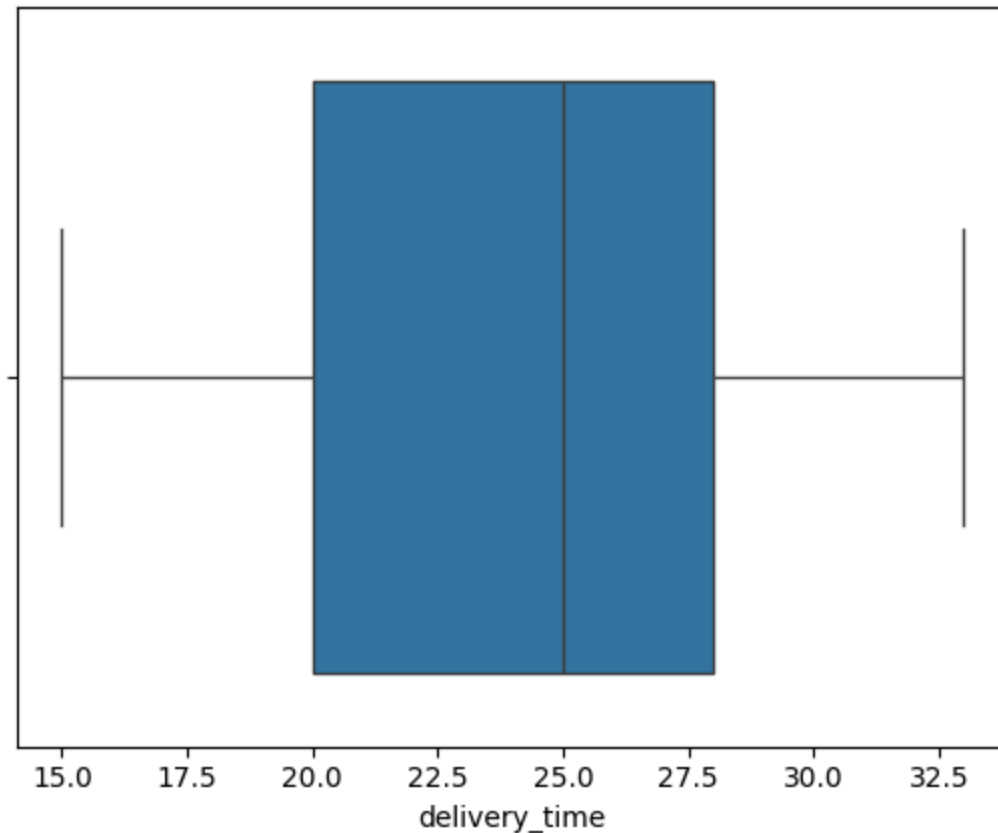
Axes(0.125,0.11;0.775x0.77)



None
Axes(0.125,0.11;0.775x0.77)



None
Axes(0.125,0.11;0.775x0.77)



None

Observations :

cost of the order :

- The distribution of cost of the order is slightly right-skewed and mean is higher than median. There are no any outliers in this column.
- A large chunk of people prefer to order food that costs around 11-13 dollars.
- A few orders cost is greater than 30 dollars. These orders might be for some expensive meals.

Food preparation time :

- The distribution of the food preparation time is nearly symmetrical and quite. Food preparation time mean and median almost similar. There are no any outliers in this column.
- The food preparation time is pretty evenly distributed between 20 and 35 minutes.

Delivery time :

- The distribution of delivery time is a bit left-skewed. The distribution delivery time mean is lower than median. There are no any outliers in delivery time column.
- Comparatively more number of orders have delivery time around between 24 and 31 minutes.

Question 7: Which are the top 5 restaurants in terms of the number of orders received? [1 mark]

```
In [25]: #group by restaurant name and get top 5 restaurants with highest number of orders d
df.groupby(['restaurant_name'])['order_id'].count().head(5)
```

```
Out[25]:
```

order_id	
restaurant_name	
'wichcraft	1
12 Chairs	4
5 Napkin Burger	5
67 Burger	1
Alidoro	1

dtype: int64

```
In [26]: #group by restaurant name and get top 5 restaurants with highest number of orders d
df['restaurant_name'].value_counts(normalize=True).sort_values(ascending=False).hea
```

```
Out[26]:
```

proportion	
restaurant_name	
Shake Shack	0.115385
The Meatball Shop	0.069547
Blue Ribbon Sushi	0.062698
Blue Ribbon Fried Chicken	0.050580
Parm	0.035827

dtype: float64

Observations:

- Shake Shack, The Meatball Shop, Blue Ribbon Sushi, Blue Ribbon Fried Chicken, and Parm are the top 5 restaurants with the highest orders received. Almost, 33% of orders are received from those five restaurants in dataset.

Question 8: Which is the most popular cuisine on weekends? [1 mark]

```
In [27]: # Get most popular cuisine on weekends
weekends = df[df['day_of_the_week'] == 'Weekend']
weekends
weekends.groupby(['cuisine_type'])['order_id'].count().sort_values(ascending=False)
```

Out[27]:

cuisine_type	order_id
American	415
Japanese	335
Italian	207
Chinese	163
Mexican	53
Indian	49
Middle Eastern	32
Mediterranean	32
Thai	15
French	13
Korean	11
Southern	11
Spanish	11
Vietnamese	4

dtype: int64

Observations:

- The most popular cuisine type on weekends is American. but Japanese italian cusiones get more than 200 orders in weekends in dataset.

Question 9: What percentage of the orders cost more than 20 dollars? [2 marks]

```
In [28]: percentage_above_20 = df[df['cost_of_the_order'] > 20]
proportion_above_20 = percentage_above_20['cost_of_the_order'].count() / len(df)*10
print(f"The Proportion of orders costing more than 20 dollars: {proportion_above_20}
```

The Proportion of orders costing more than 20 dollars: 29.24%

Observations:

- Percentage of orders costing more than 20 dollars is 29.24%

Question 10: What is the mean order delivery time? [1 mark]

```
In [29]: # get the mean delivery time
round(df['delivery_time'].mean(), 2)
```

```
Out[29]: np.float64(24.16)
```

Observations :

- The mean order delivery time is around 24.16 minutes.

Question 11: The company has decided to give 20% discount vouchers to the top 5 most frequent customers. Find the IDs of these customers and the number of orders they placed. [1 mark]

```
In [30]: # Write the code here
df.groupby(['customer_id'])['order_id'].count().sort_values(ascending=False).head(5)
```

```
Out[30]:
```

order_id	
customer_id	
52832	13
47440	10
83287	9
250494	8
276192	7

dtype: int64

Observations:

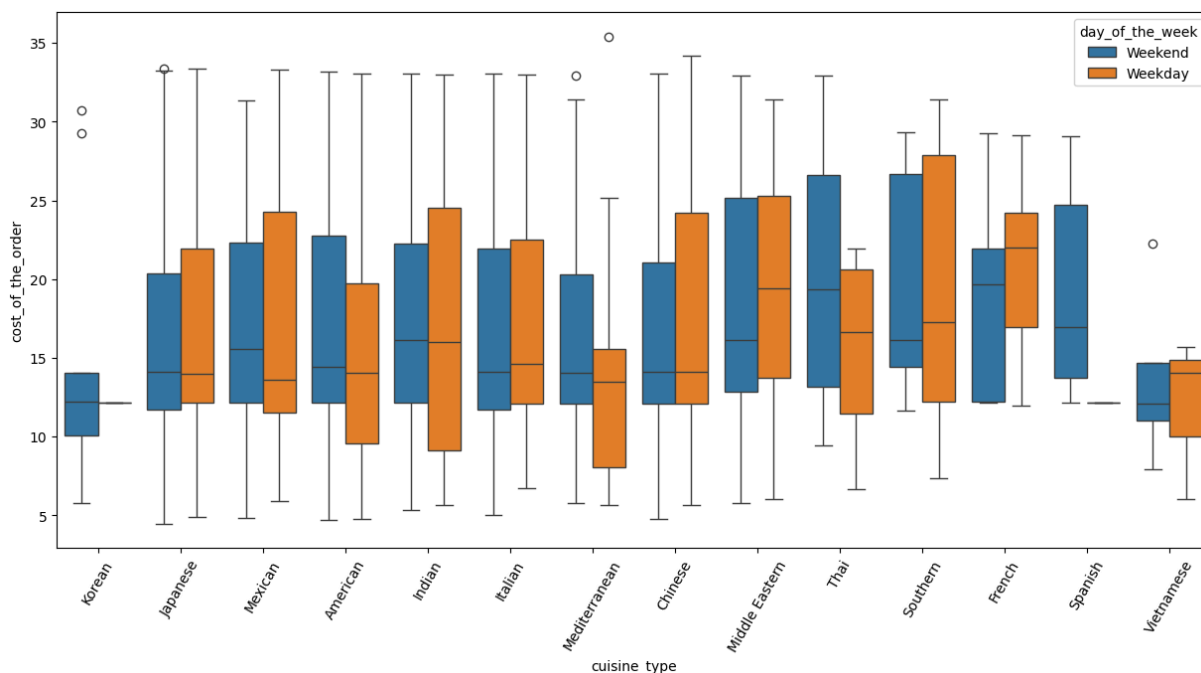
- Customer with ID 52832 has ordered food 13 times from foodHub app followed by Customer with ID 47440, 83287, 250494, 259341. FoodHub has decided to give 20% to those customers.

Multivariate Analysis

Question 12: Perform a multivariate analysis to explore relationships between the important variables in the dataset. (It is a good idea to explore relations between numerical

variables as well as relations between numerical and categorical variables) [10 marks]

```
In [31]: # Relationship between cost of the order and cuisine type
plt.figure(figsize=(15,7))
plt.xticks(rotation = 60)
sns.boxplot(data = df, x = "cuisine_type", y = "cost_of_the_order", hue='day_of_the_week')
plt.show()
```



Observation :

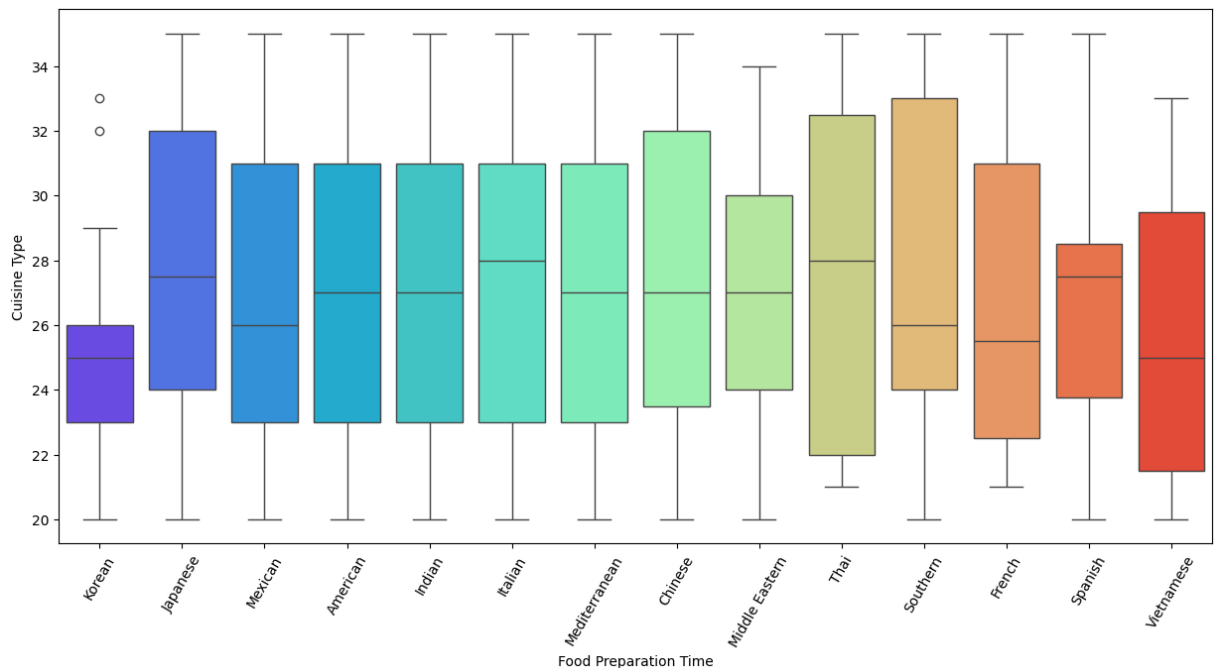
- On weekends, the cost of orders for Mexican, Middle Eastern, and Thai cuisines is high, and those cuisines may cook special dishes.
- Vietnamese, Spanish, and Korean cuisines cost less than other weekday cuisines.
- The boxplots for Italian, American, Chinese, and Japanese cuisines are quite similar, which indicates that their quartile costs are similar.
- French, Thai, and Spanish cuisines are costlier compared to other cuisines.
- French and Middle Eastern cuisine's cost of order is high on weekdays compared to weekends. Maybe, those cuisines give special offers or cook special dishes in weekdays. Or maybe those cuisines are located near to office areas.

```
In [35]: # Relationship between food preparation time and cuisine type
# write code for figure size
plt.figure(figsize=(15,7))
# Create the boxplot for "cuisine_type" and "food_preparation_time" columns.
sns.boxplot(data = df, x = "cuisine_type", y = "food_preparation_time", palette = '
# Adding labels and title for better readability
```

```
plt.ylabel('Cuisine Type')
plt.xlabel('Food Preparation Time')
plt.xticks(rotation = 60)
# Display the plot
plt.show()
```

/tmp/ipython-input-502866046.py:5: FutureWarning:

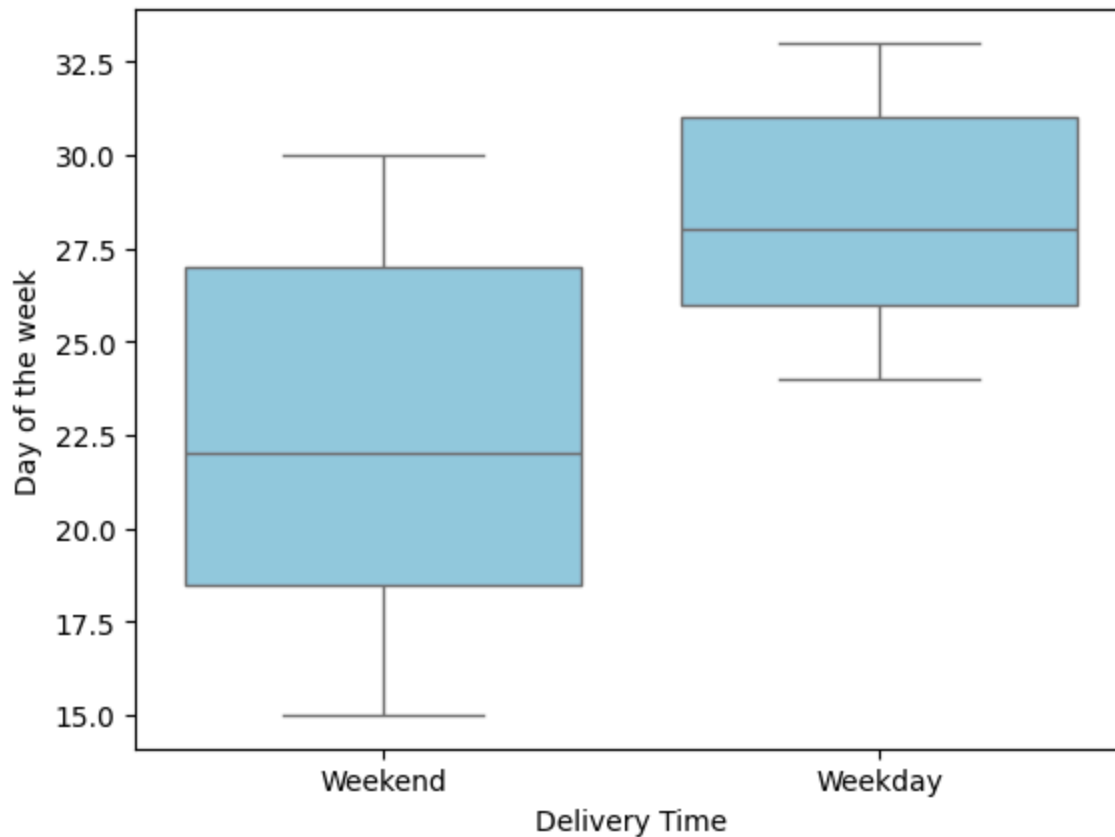
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.



Observation

- The median food preparation time lies between 24 and 30 minutes for all the cuisines.
- Outliers are present for the food preparation time of Korean cuisine.
- Korean cuisine takes less time compared to the other cuisines.
- Food preparation time is very consistent for most of the cuisines.

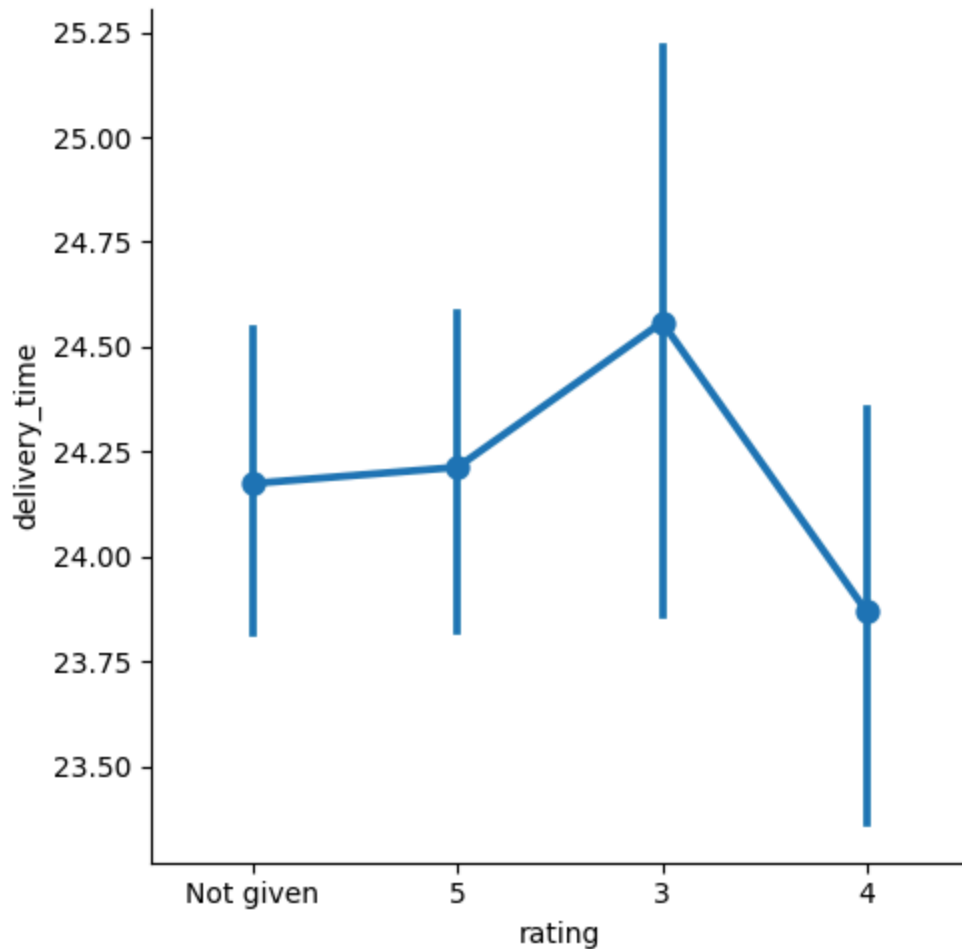
```
In [36]: # Relationship between day of the week and delivery time
# Create the boxplot for "day_of_the_week" and "delivery_time" columns.
sns.boxplot(data = df, x = "day_of_the_week", y = "delivery_time", color='skyblue')
# Adding Labels for better readability
plt.ylabel('Day of the week')
plt.xlabel('Delivery Time')
# Display the plot
plt.show()
```



Observations :

- All delivery time is higher on weekends as compared to the weekdays.because, over all traffic rises in the weekdays.

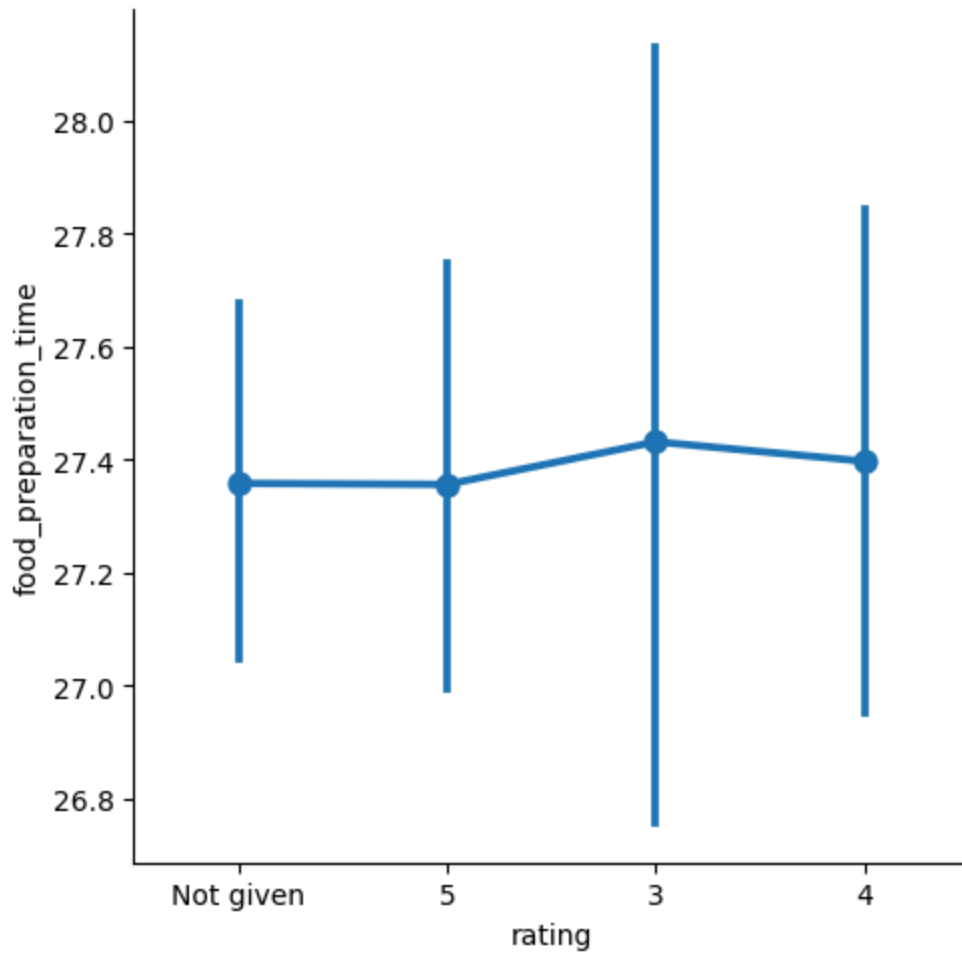
```
In [37]: # Relationship between rating and delivery time
# Create the catplot with point for 'rating' and "delivery_time" columns.
sns.catplot(data=df, x = 'rating', y = 'delivery_time', kind='point');
# Display the plot
plt.show()
```



Observations :

- We can see that ratings are lower. Delivery time plays big role in low-rating of the orders.

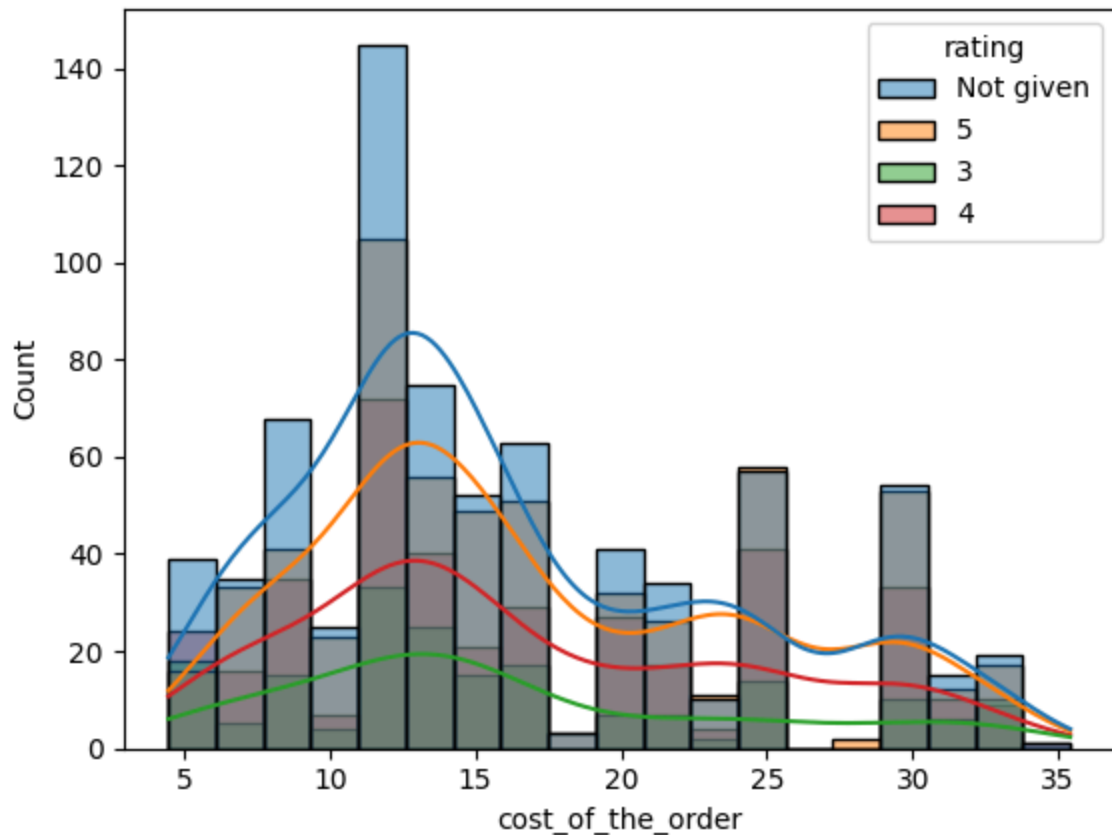
```
In [38]: # Relationship between rating and food preparation time
# Create the catplot with point for 'rating' and 'food_preparation_time' columns.
sns.catplot(data=df, x='rating', y='food_preparation_time', kind='point');
# Display the plot
plt.show()
```



Observations :

- As we can see that, food preparation time does not effect in the low-rating of orders.

```
In [39]: # Relationship between rating and cost of the order
# Create the histplot for 'cost_of_the_order' column and hue = 'rating'.
sns.histplot(data=df, x='cost_of_the_order', hue='rating', kde=True)
# Display the plot
plt.show()
```

Observation :

- Customers with pay costs of orders `~11\$, those customers like to give ratings.
- Customers do not like to give less rating.
- Customer with high paying costs of orders prefer to give ratings.
- Customeres with pay costs of orders is between **11and13**, those orders counts is higher then others bins.

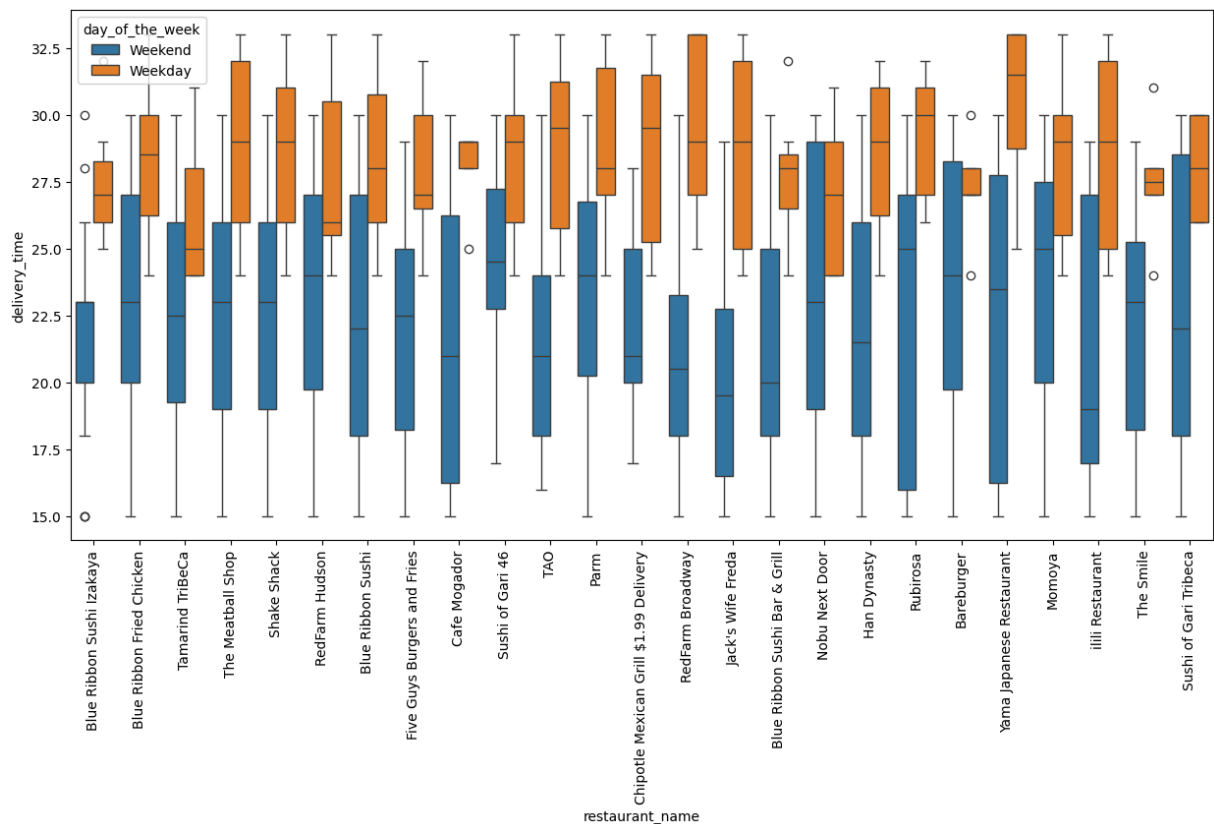
```
In [40]: # importing plotly
import plotly.express as px
# Create plotly histogram plot for restaurant name column
his = px.histogram(df, x="restaurant_name", y="order_id", color="cuisine_type") # R
# Show plot
his.show()
```

Observation :

- *The Meatball Shop is serving Italian and American cuisines food.*
- *FoodHub have maximum number of American cuisine type restaurants followed by Japanese, Italian and Chinese .*
- *Shake Shack has gotten maximum numbers of orders.*

```
In [41]: #the top 25 restaurants in terms of the number of orders received list and make box
#the top 25 restaurants list
Restaurant_list = ['Shake Shack', 'The Meatball Shop', 'Blue Ribbon Sushi', 'Blue R
                'TAO', 'Han Dynasty', 'Blue Ribbon Sushi Bar & Grill', 'Nobu Nex
                'Blue Ribbon Sushi Izakaya', 'Bareburger', 'Tamarind TriBeCa', "Jac
                'The Smile', 'Cafe Mogador', 'Yama Japanese Restaurant', 'ilili Res
new_restaurant_list=[]
for restaurant in Restaurant_list:
    if restaurant in ['Shake Shack', 'The Meatball Shop', 'Blue Ribbon Sushi', 'Blu
                'Han Dynasty', 'Blue Ribbon Sushi Bar & Grill', 'Nobu Next Do
                'Blue Ribbon Sushi Izakaya', 'Bareburger', 'Tamarind TriBeCa', "
                'The Smile', 'Cafe Mogador', 'Yama Japanese Restaurant', 'ilili
        new_restaurant_list.append(restaurant)
filtered_df = df[df['restaurant_name'].isin(new_restaurant_list)]
plt.figure(figsize=(15,7))
```

```
sns.boxplot(data=filtered_df, x='restaurant_name', y="delivery_time", hue='day_of_the_week')
plt.xticks(rotation=90)
plt.show()
```

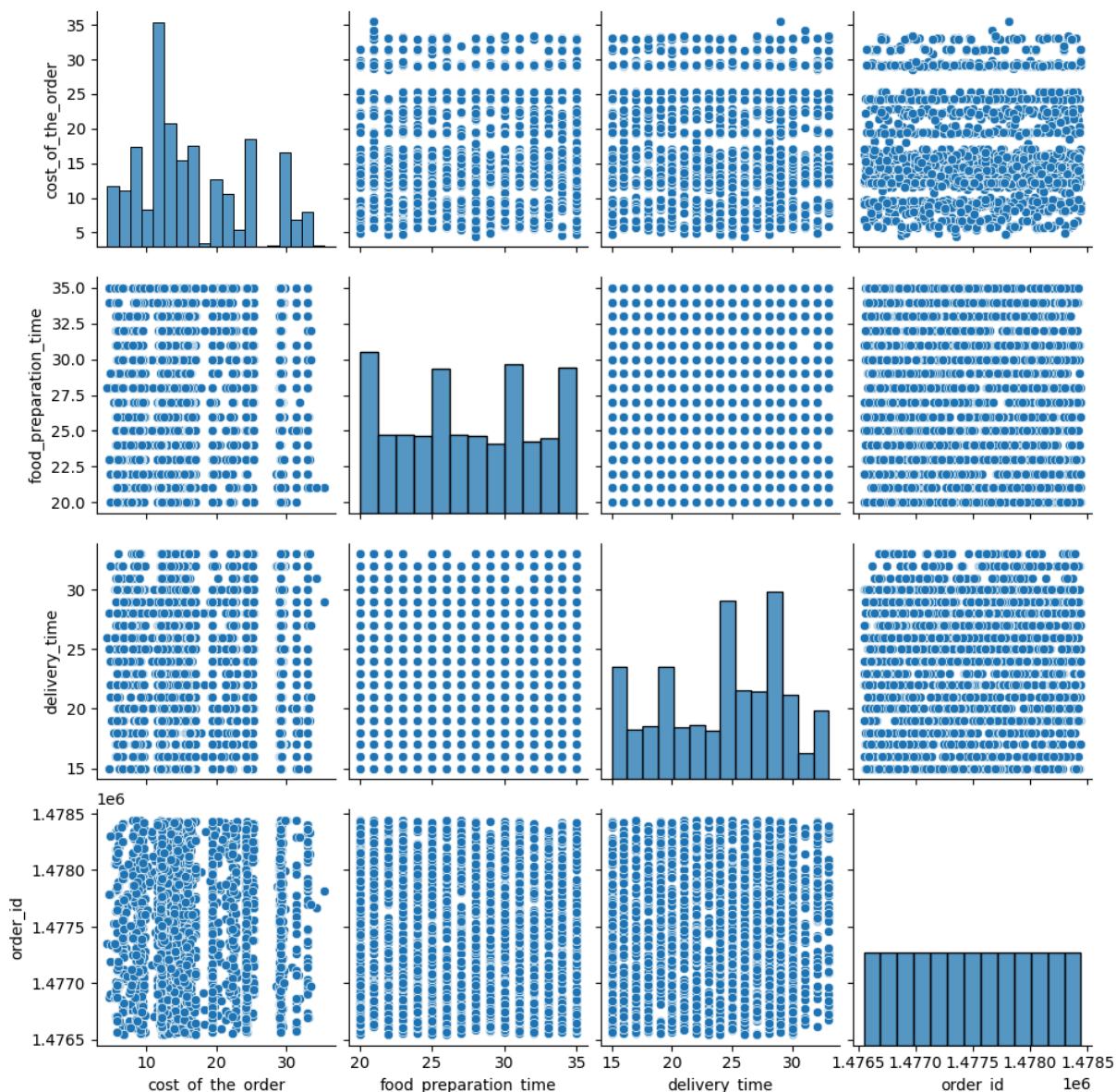


Observation :

- Rubirosa and Momoya restaurent mediam is around 25 minutes in weekends. Yama Japanese Restaurent mediam is around 31 minutes. but We can see others restaurents, there are many variability in delivery time. Some restaurents have outer line also.

```
In [42]: # Check corelation between numerics columns.
# write pair plot cord for 'cost_of_the_order', 'food_preparation_time', 'delivery_time'
sns.pairplot(data=df[['cost_of_the_order', 'food_preparation_time', 'delivery_time']])
```

```
Out[42]: <seaborn.axisgrid.PairGrid at 0x79220486d7d0>
```



Obvferstation :

- *There are no corelation between numerical columns.*

Question 13: The company wants to provide a promotional offer in the advertisement of the restaurants. The condition to get the offer is that the restaurants must have a rating count of more than 50 and the average rating should be greater than 4. Find the restaurants fulfilling the criteria to get the promotional offer. [3 marks]

```
In [43]: df_rated = df[df['rating'] != 'Not given'].copy()
df_rated['rating'] = df_rated['rating'].astype('int')
df_rating_count = df_rated.groupby(['restaurant_name'])['rating'].count().sort_valu
df_rating_count.head()
```

Out[43]:

	restaurant_name	rating
0	Shake Shack	133
1	The Meatball Shop	84
2	Blue Ribbon Sushi	73
3	Blue Ribbon Fried Chicken	64
4	RedFarm Broadway	41

```
In [44]: rest_names = df_rating_count[df_rating_count['rating'] > 50]['restaurant_name']
df_mean_4 = df_rated[df_rated['restaurant_name'].isin(rest_names)].copy()
df_mean_4.groupby(df_mean_4['restaurant_name'])['rating'].mean().sort_values(ascending=True)
```

Out[44]:

	restaurant_name	rating
0	The Meatball Shop	4.511905
1	Blue Ribbon Fried Chicken	4.328125
2	Shake Shack	4.278195
3	Blue Ribbon Sushi	4.219178

Observations:

- The restaurants fulfilling the criteria to get the promotional offer are: 'The Meatball Shop', 'Blue Ribbon Fried Chicken', 'Shake Shack' and 'Blue Ribbon Sushi'.

Question 14: The company charges the restaurant 25% on the orders having cost greater than 20 dollars and 15% on the orders having cost greater than 5 dollars. Find the net revenue generated by the company across all orders. [3 marks]

```
In [45]: #create function to determine the net revenue
def compute_rev(x):
    if x > 20:
        return x*0.25
    elif x > 5:
        return x*0.15
    else:
        return x*0
# Add new revenue column
df['Revenue'] = df['cost_of_the_order'].apply(compute_rev)
# Get 5 Rows
df.head()
```

Out[45]:

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week
0	1477147	337525	Hangawi	Korean	30.75	Weekend
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday

In [46]:

```
# get the total revenue and print it
total_rev = df['Revenue'].sum()
print('The net revenue is around', round(total_rev, 2), 'dollars')
```

The net revenue is around 6166.3 dollars

Observations:

- The net revenue is around 6166.3 dollars.

Question 15: The company wants to analyze the total time required to deliver the food. What percentage of orders take more than 60 minutes to get delivered from the time the order is placed? (The food has to be prepared and then delivered.) [2 marks]

In [47]:

```
# add a new column to the dataframe df to store the total food delivery time
df['total_food_delivery_time'] = df['food_preparation_time'] + df['delivery_time']
round(df[df['total_food_delivery_time'] > 60].shape[0] / df.shape[0] * 100, 2)
```

Out[47]: 10.54

Observations:

- The percentage of orders that have more than 60 minutes of total food delivery time is 10.54 %.

Question 16: The company wants to analyze the delivery time of the orders on weekdays and weekends. How does the mean delivery time vary during weekdays and weekends? [2 marks]

In [48]:

```
df_weekend = df[df['day_of_the_week'] == 'Weekend']
print('The mean delivery time on weekends is around', round(df_weekend['delivery_time'].mean(), 2))
```

The mean delivery time on weekends is around 22.47 minutes

```
In [49]: df_weekday = df[df['day_of_the_week'] == 'Weekday']
print('The mean delivery time on weekdays is around', round(df_weekday['delivery_ti
```

The mean delivery time on weekdays is around 28.34 minutes

Observations :

- The mean delivery time on weekdays is around 22.47 minutes.
- The mean delivery time on weekdays is around 28.34 minutes.

Conclusion and Recommendations

Question 17: What are your conclusions from the analysis? What recommendations would you like to share to help improve the business? (You can use cuisine type and feedback ratings to drive your business recommendations.) [6 marks]

Conclusions:

- FoodHub has 178 unique restaurants. The Shake Shack restaurant is very popular among the customers followed by The Meatball Shop, Blue Ribbon Sushi, Blue Ribbon Fried Chicken restaurants.
- Some cuisines are very popular among customers. FoodHub has 14 unique cuisines. FoodHub gets 80% orders from American, Japanese, Italian and Chinese cuisines.
- FoodHub gets 70% orders in weekends.
- High cost paying customers prefer to give rating. 38% of orders are not rating.

Recommendations:

- Customers rating is most important factor for Business. High cost paying customers prefer to give rating. FoodHub should improve app, so other customers start to give rating.
- FoodHub gets 80% orders from American, Japanese, Italian and Chinese cuisines. FoodHub should integrate with restaurants serving with those cuisines.
- FoodHub should provide promotional offers to top-rated popular restaurants like Shake Shack, The Meatball Shop, Blue Ribbon Fried Chicken and Blue Ribbon Sushi.
- The Shake Shack, The Meatball Shop, Blue Ribbon Sushi, Blue Ribbon Fried Chicken restaurants are more popular. FoodHub should arrange more delivery boys for those restaurants so delivery time can reduce.
- FoodHub gets 70% of orders in weekends. FoodHub should hire higher delivery boys for weekends if needs.
- There are many variability in food delivery time in weekdays and weekends. FoodHub should make sure, Delivery boys give delivery on time.

- *The percentage of orders that have more than 60 minutes of total delivery time is 10.54 %. FoodHub should take strong decision like instant delivery. Delivery time affects customer reating.*
 - *The company has decided to give 20% discount vouchers to the top 5 most frequent customers to encourang customers.*
-