PYF_Project_LearnerNotebook_LowCode_Learner (1)

January 5, 2025

1 Project Python Foundations: FoodHub Data Analysis

1.0.1 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.

1.0.2 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.

1.0.3 Data Description

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

1.0.4 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_of_the_order: 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

1.0.5 Please read the instructions carefully before starting the project.

This is a commented Jupyter IPython Notebook file in which all the instructions and tasks to be performed are mentioned. Read along carefully to complete the project. * Blanks '_____' are provided in the notebook that needs to be filled with an appropriate code to get the correct result. Please replace the blank with the right code snippet. With every '_____' blank, there is a comment that briefly describes what needs to be filled in the blank space. * Identify the task to be performed correctly, and only then proceed to write the required code. * Fill the code wherever asked by the commented lines like "# write your code here" or "# complete the code". Running incomplete code may throw an error. * Please run the codes in a sequential manner from the beginning to avoid any unnecessary errors. * You can the results/observations derived from the analysis here and use them to create your final presentation.

1.0.6 Let us start by importing the required libraries

[1]:
```
# Installing the libraries with the specified version.
!pip install numpy==1.25.2 pandas==2.2.2 matplotlib==3.8.0 seaborn==0.13.1 -q␣
 ↪--user --no-warn-script-location
```

```
                         18.2/18.2 MB
71.5 MB/s eta 0:00:00
                         294.8/294.8 kB
20.2 MB/s eta 0:00:00
```

Note: *After running the above cell, kindly restart the notebook kernel and run all cells sequentially from the start again.*

[2]:
```
# 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
```

1.0.7 Understanding the structure of the data

[3]:
```
# uncomment and run the following lines for Google Colab
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

[4]:
```
# Read the data
df = pd.read_csv('/content/gdrive/My Drive/python_files/foodhub.csv') ## Fill␣
 ↪the blank to read the data
```

[5]:
```
# Returns the first 5 rows
df.head()
```

[5]:

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week	rating	food_preparation_time	delivery_time
0	1477147	337525	Hangawi	Korean	30.75	Weekend	Not given	25	20
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend	Not given	25	23
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday	5	23	28
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend	3	25	15
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday	4	25	24

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

There are 1,898 rows and 9 columns

[6]:
```
# Check the shape of the dataset
df.shape ## Fill in the blank
```

[6]:
```
(1898, 9)
```

1.0.9 Question 2: What are the datatypes of the different columns in the dataset? [0.5 mark]

integer, object, and float

dtypes: float64(1), int64(4), object(4)

[7]:
```
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
```

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

[8]:
```
# Checking for missing values in the data
df.isnull().sum()
```

[8]:
```
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
```

1.0.11 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]

Food prep times:

Min: 20 mins

Mean: 27.37 mins

Max: 35 mins

[9]:
```
df.describe(include='all').T
```

[9]:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
order_id	1898.0	NaN	NaN	NaN	1477495.5	548.049724	1476547.0	1477021.25	1477495.5	1477969.75	1478444.0
customer_id	1898.0	NaN	NaN	NaN	171168.478398	113698.139743	1311.0	77787.75	128600.0	270525.0	405334.0
restaurant_name	1898	178	Shake Shack	219	NaN	NaN	NaN	NaN	NaN	NaN	NaN
cuisine_type	1898	14	American	584	NaN	NaN	NaN	NaN	NaN	NaN	NaN
cost_of_the_order	1898.0	NaN	NaN	NaN	16.498851	7.483812	4.47	12.08	14.14	22.2975	35.41
day_of_the_week	1898	2	Weekend	1351	NaN	NaN	NaN	NaN	NaN	NaN	NaN
rating	1898	4	Not given	736	NaN	NaN	NaN	NaN	NaN	NaN	NaN
food_preparation_time	1898.0	NaN	NaN	NaN	27.37197	4.632481	20.0	23.0	27.0	31.0	35.0
delivery_time	1898.0	NaN	NaN	NaN	24.161749	4.972637	15.0	20.0	25.0	28.0	33.0

[10]:
```
# Get the summary statistics of the numerical data
def print_time_stats(df):

    min_time = df['food_preparation_time'].min()
    mean_time = df['food_preparation_time'].mean().round(2)
    max_time = df['food_preparation_time'].max()

    print("Min:", min_time)
    print("Mean:", mean_time)
    print("Max:", max_time)

print_time_stats(df)
```

```
Min: 20
Mean: 27.37
Max: 35
```

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

736 orders not rated

[11]:
```
df['rating'].value_counts() ## Complete the code
```

[11]:
```
rating
Not given    736
5            588
4            386
3            188
Name: count, dtype: int64
```

1.0.13 Exploratory Data Analysis (EDA)

1.0.14 Univariate Analysis

1.0.15 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

[12]:
```
# check unique order ID
df['order_id'].nunique()
```

[12]:
```
1898
```

Customer ID

[13]:
```
# check unique customer ID
df['customer_id'].nunique()  ## Complete the code to find out number of unique␣
↪Customer ID
```

[13]:
```
1200
```

Restaurant name

[14]:
```
# check unique Restaurant Name
df['restaurant_name'].nunique()  ## Complete the code to find out number of␣
↪unique Restaurant Name
```

[14]:
```
178
```

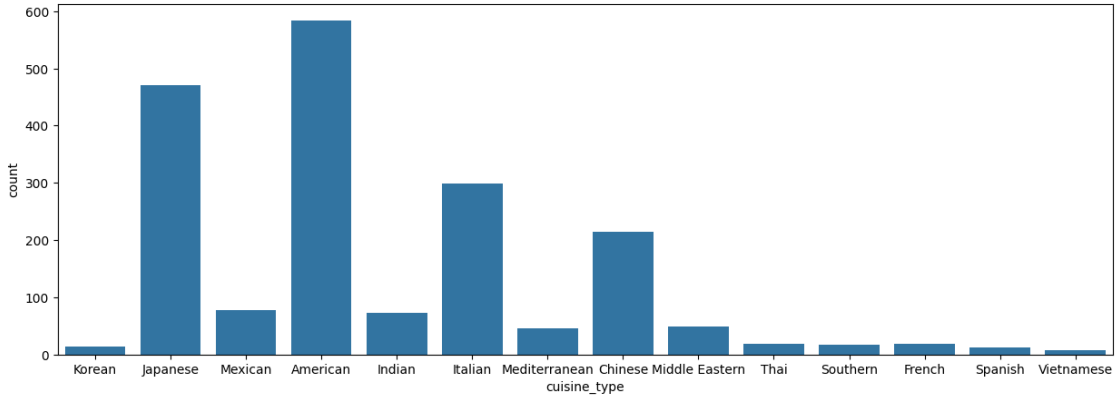
Cuisine type

[15]:
```
# Check unique cuisine type
df['cuisine_type'].nunique() ## Complete the code to find out  number of ␣
↪unique cuisine type
```

[15]: 14

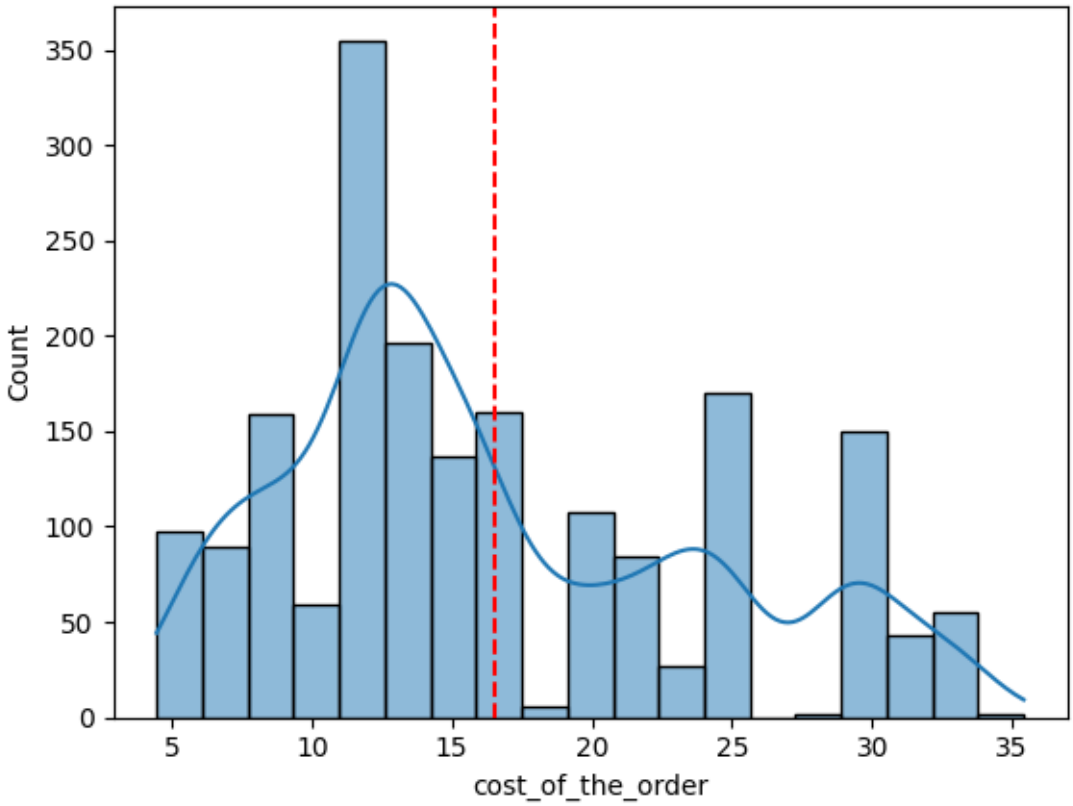
[16]:
```
plt.figure(figsize = (15,5))
sns.countplot(data = df, x = 'cuisine_type') ## Create a countplot for cuisine type.
```

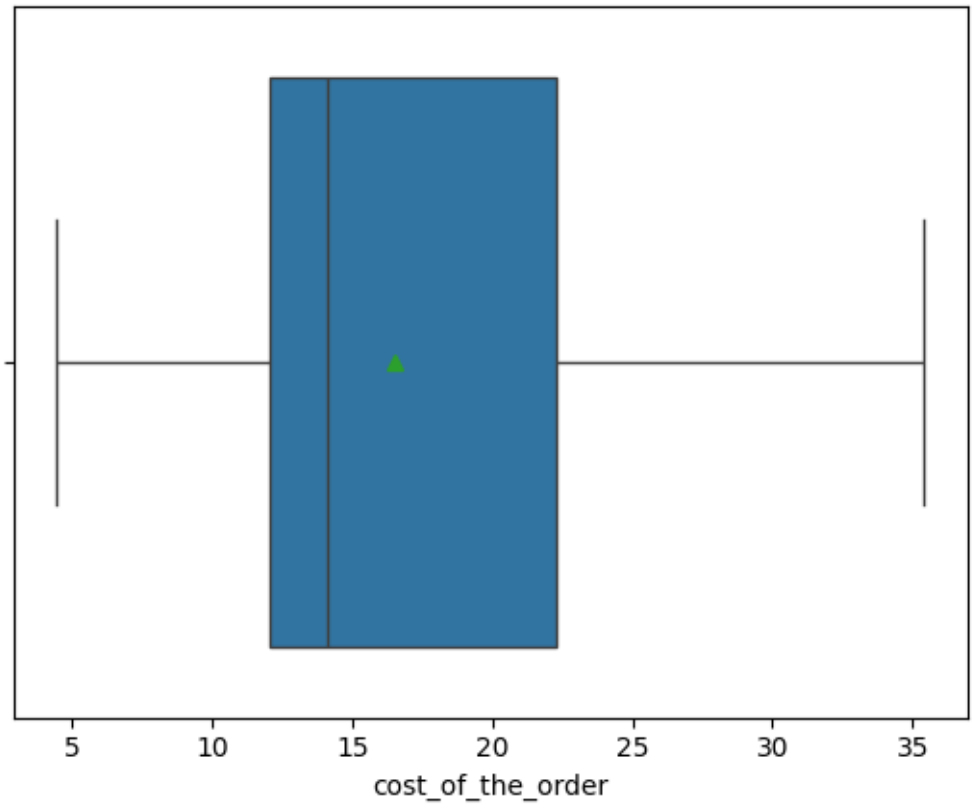
[16]: <Axes: xlabel='cuisine_type', ylabel='count'>



Cost of the order

[17]:
```
sns.histplot(data=df, x='cost_of_the_order', kde=True)
plt.axvline(df['cost_of_the_order'].mean(),color='r',linestyle='--') #mean value of the column indicated with doted line symbol
plt.show()
sns.boxplot(data=df,x='cost_of_the_order', showmeans=True) ## Boxplot for the cost of order
plt.show()
```


Count
cost_of_the_order
0
50
100
150
200
250
300
350
5
10
15
20
25
30
35



Observation:

The histplot is skewed to the right. Average cost of order is around 16 dollars.

The boxplot indicates that the median cost is about 14 dollars and mean is around 16 dollars.

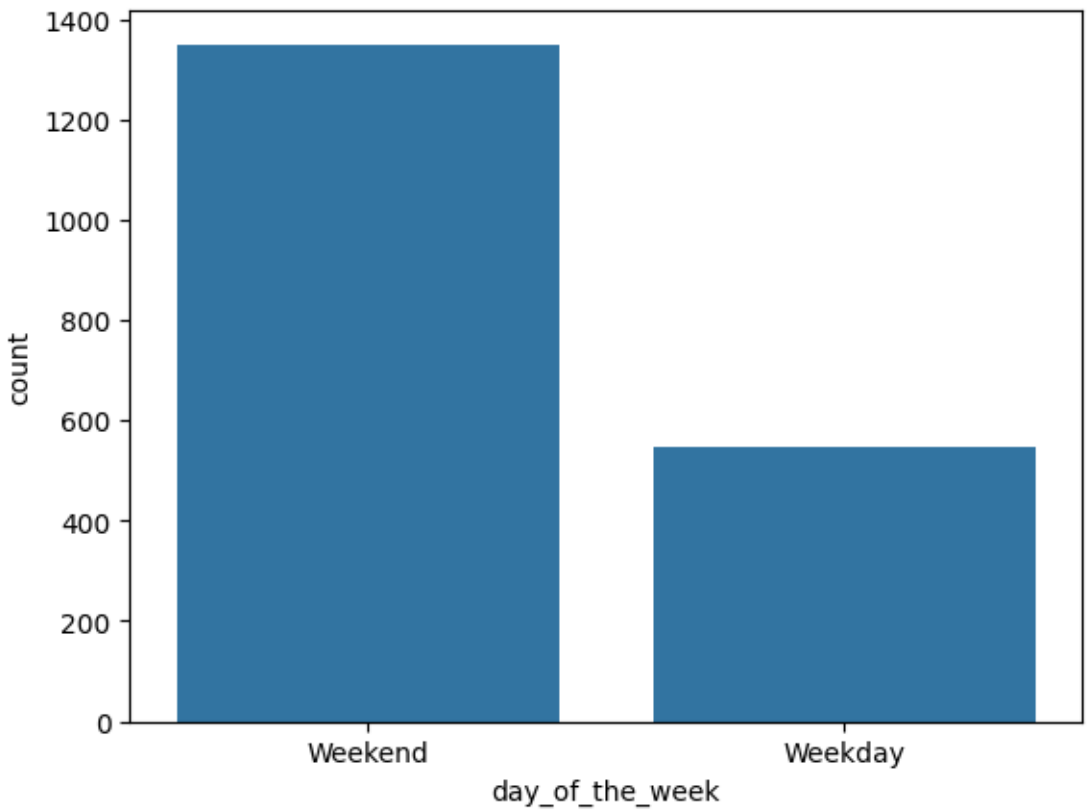
Day of the week

[18]:
```
# # Check the unique values
df['day_of_the_week'].nunique() ## Complete the code to check unique values for␣
↪the 'day_of_the_week' column
```

[18]: 2

[19]:
```
sns.countplot(data = df, x = 'day_of_the_week') ## Complete the code to plot a␣
↪bar graph for 'day_of_the_week' column
```

[19]: <Axes: xlabel='day_of_the_week', ylabel='count'>



Observation:

More than 50% of the orders are made on the weekend

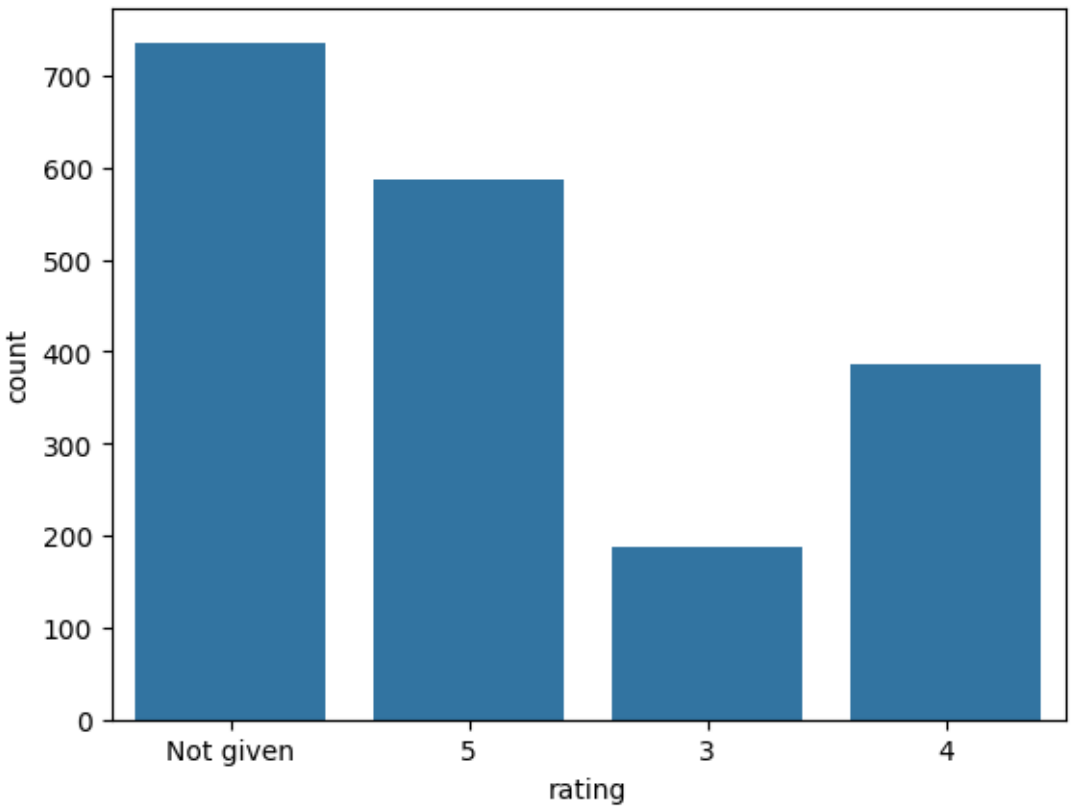
Rating

[20]:
```
# Check the unique values
df['rating'].unique() ## Complete the code to check unique values for the␣
↪'rating' column
```

[20]:
```
array(['Not given', '5', '3', '4'], dtype=object)
```

[21]:
```
sns.countplot(data = df, x = 'rating') ## Complete the code to plot bar graph␣
↪for 'rating' column
```

[21]:
```
<Axes: xlabel='rating', ylabel='count'>
```

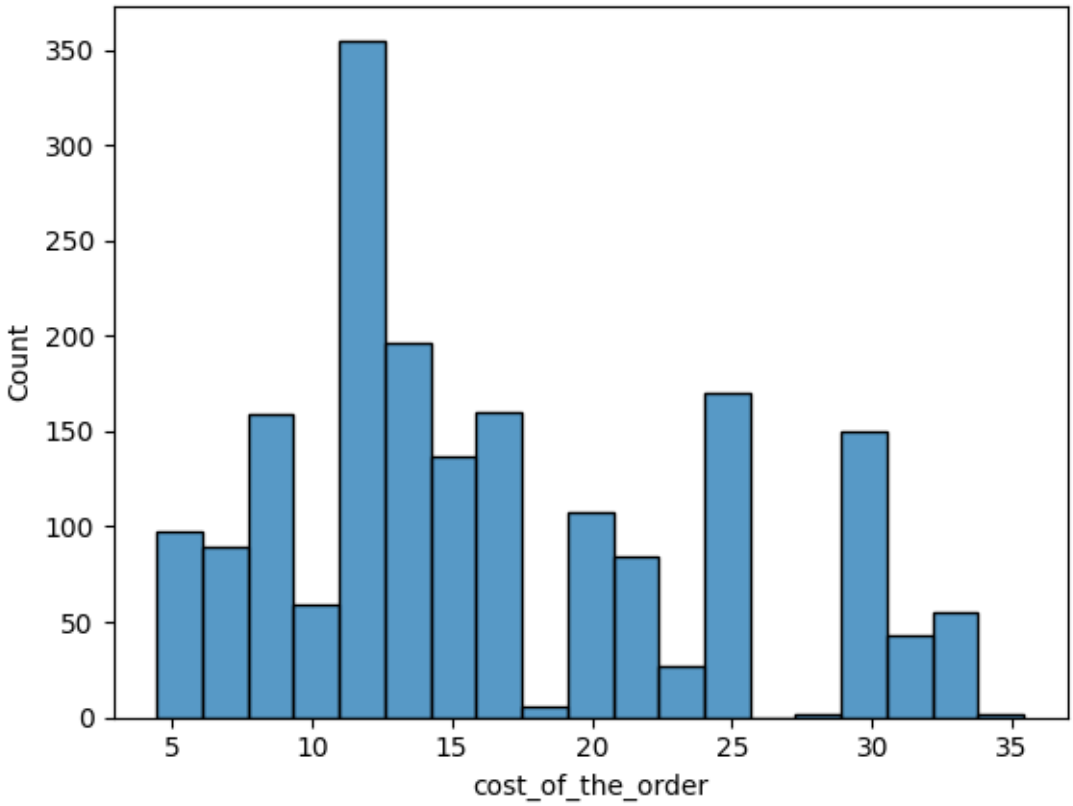


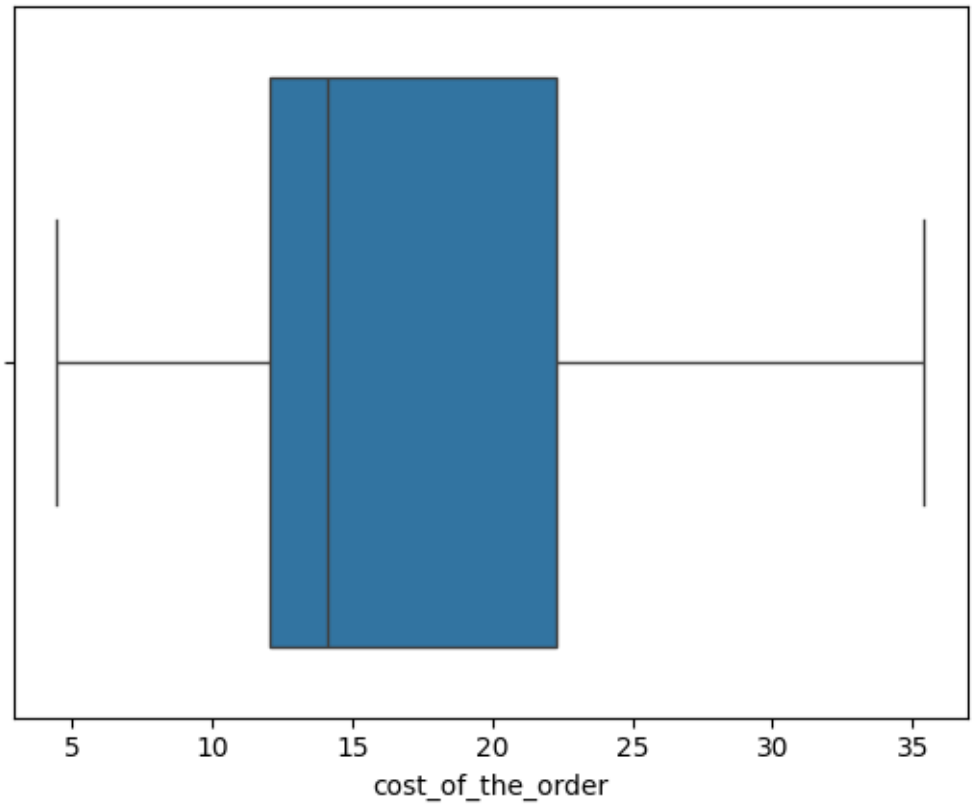
Observation:

Majority of the time a rating is not given

Food Preparation time

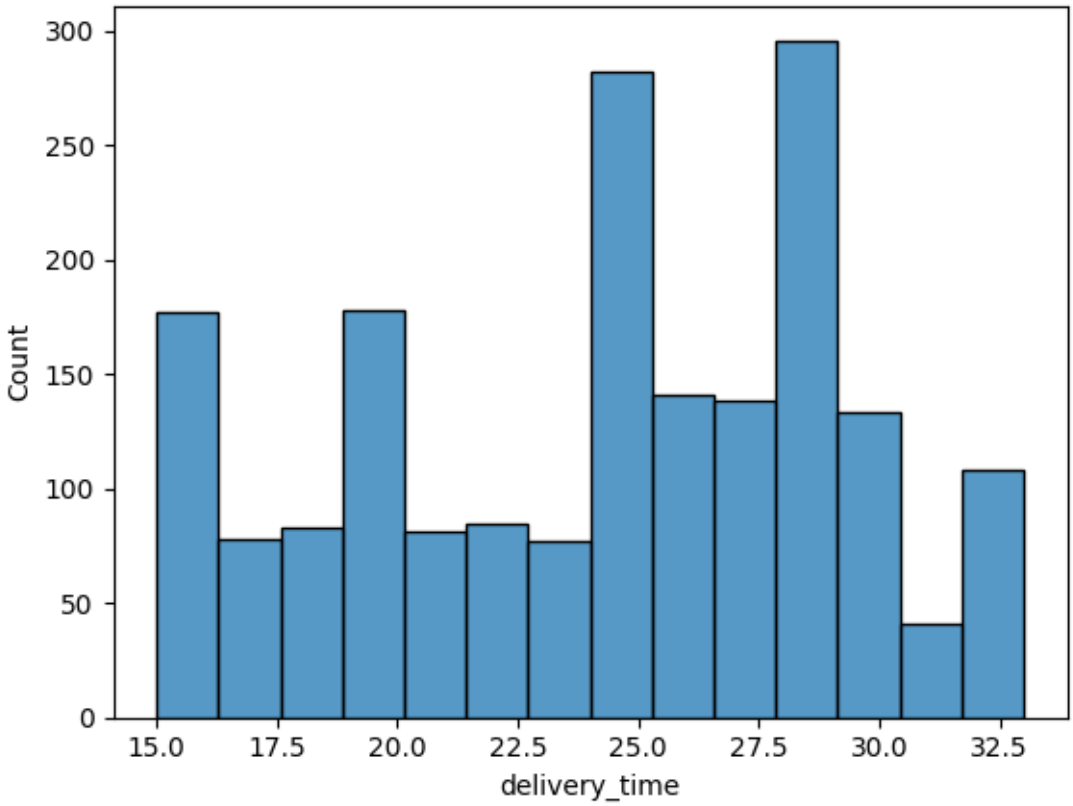
[22]:
```
sns.histplot(data=df,x='cost_of_the_order') ## Complete the code to plot the␣
↪histogram for the cost of order
plt.show()
sns.boxplot(data=df,x='cost_of_the_order') ## Complete the code to plot the␣
↪boxplot for the cost of order
plt.show()
```

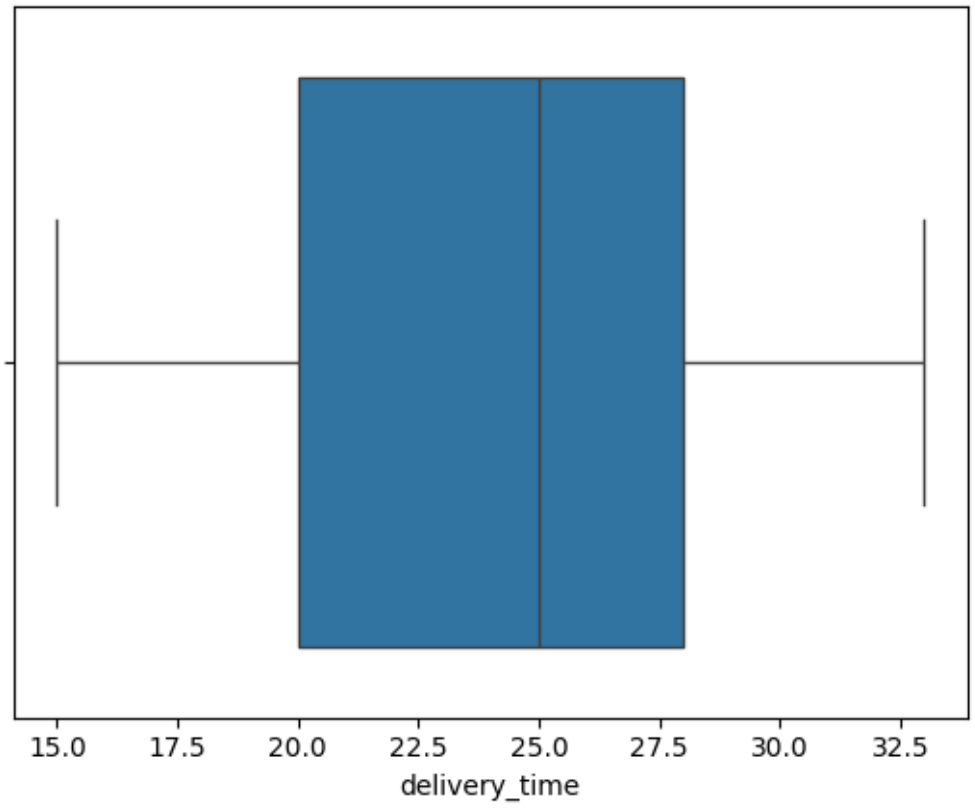

Count
350
300
250
200
150
100
50
0
5
10
15
20
25
30
35
cost_of_the_order



Delivery time

[23]:
```
sns.histplot(data=df,x='delivery_time') ## Complete the code to plot the␣
↪histogram for the delivery time
plt.show()
sns.boxplot(data=df,x='delivery_time') ## Complete the code to plot the boxplot␣
↪for the delivery time
plt.show()
```


300
250
200
150
100
50
0
Count
15.0
17.5
20.0
22.5
25.0
27.5
30.0
32.5
delivery_time



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

[24]:
```
# Get top 5 restaurants with highest number of orders
df['restaurant_name'].value_counts().head(5) ## Complete the code
```

[24]:
```
restaurant_name
Shake Shack                  219
The Meatball Shop            132
Blue Ribbon Sushi            119
Blue Ribbon Fried Chicken     96
Parm                          68
Name: count, dtype: int64
```

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

American is the most popular cuisine ordered on the weekends

[25]:
```
# Get most popular cuisine on weekends
df_weekend = df[df['day_of_the_week'] == 'Weekend']
```

```
df_weekend['cuisine_type'].value_counts() ## Complete the code to check unique␣
 ↪values for the cuisine type on weekend
```

[25]:
```
cuisine_type
American          415
Japanese          335
Italian           207
Chinese           163
Mexican            53
Indian             49
Mediterranean      32
Middle Eastern     32
Thai               15
French             13
Korean             11
Southern           11
Spanish            11
Vietnamese          4
Name: count, dtype: int64
```

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

[26]:
```
# Get orders that cost above 20 dollars
df_greater_than_20 = df[df['cost_of_the_order']>20] ## Write the appropriate␣
 ↪column name to get the orders having cost above $20

# Calculate the number of total orders where the cost is above 20 dollars
print('The number of total orders that cost above 20 dollars is:',␣
 ↪df_greater_than_20.shape[0])

# Calculate percentage of such orders in the dataset
percentage = (df_greater_than_20.shape[0] / df.shape[0]) * 100

print("Percentage of orders above 20 dollars:", round(percentage, 2), '%')
```

```
The number of total orders that cost above 20 dollars is: 555
Percentage of orders above 20 dollars: 29.24 %
```

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

[27]:
```
# Get the mean delivery time
mean_del_time = df['delivery_time'].mean()  ## Write the appropriate function␣
 ↪to obtain the mean delivery time

print('The mean delivery time for this dataset is', round(mean_del_time, 2),␣
 ↪'minutes')
```

```
The mean delivery time for this dataset is 24.16 minutes
```

1.0.20 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]

[28]:
```
# Get the counts of  each customer_id
df['customer_id'].value_counts().head(5)  ## Write the appropriate column name␣
↪to get the top 5 cmost frequent customers
```

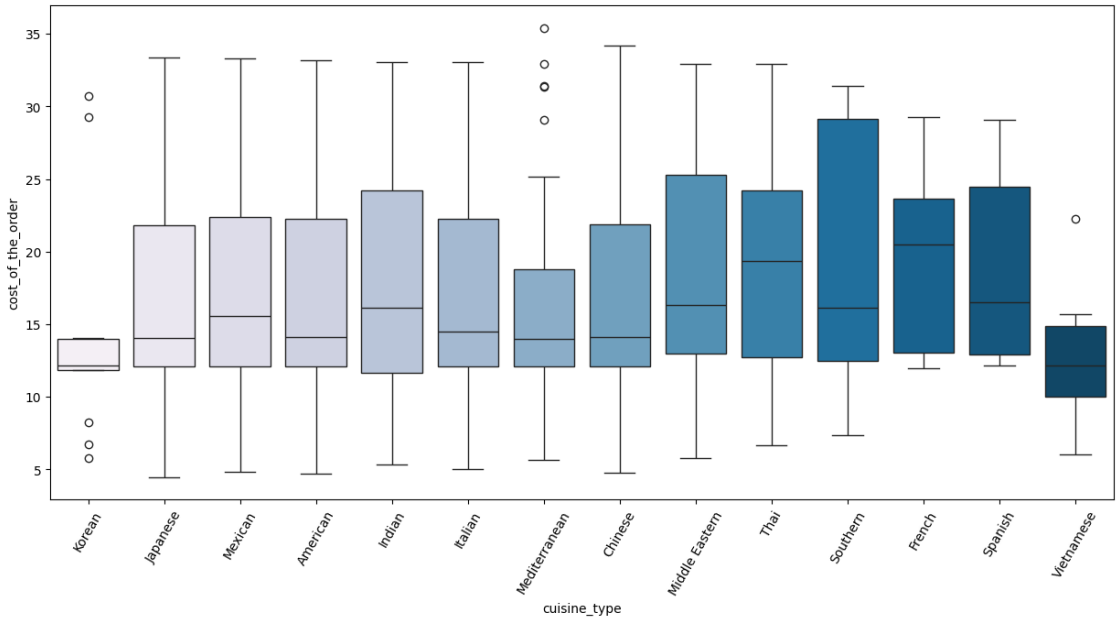
[28]:
```
customer_id
52832     13
47440     10
83287      9
250494     8
259341     7
Name: count, dtype: int64
```

1.0.21 Multivariate Analysis

1.0.22 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]

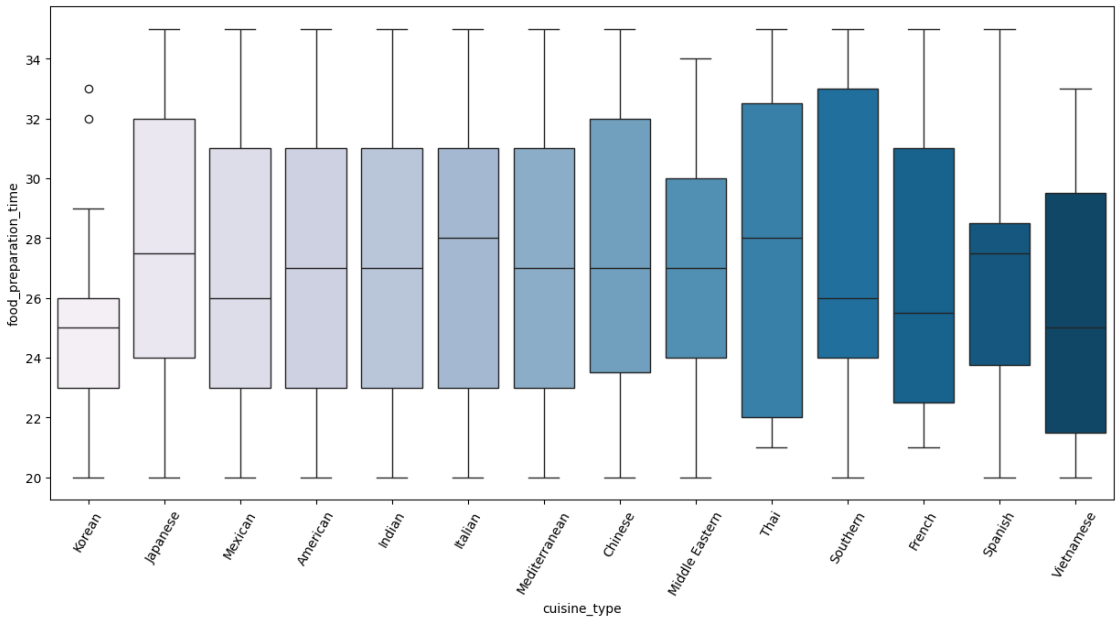
Cuisine vs Cost of the order

[29]:
```
# Relationship between cost of the order and cuisine type
plt.figure(figsize=(15,7))
sns.boxplot(x = "cuisine_type", y = "cost_of_the_order", data = df, palette =␣
↪'PuBu', hue = "cuisine_type")
plt.xticks(rotation = 60)
plt.show()
```



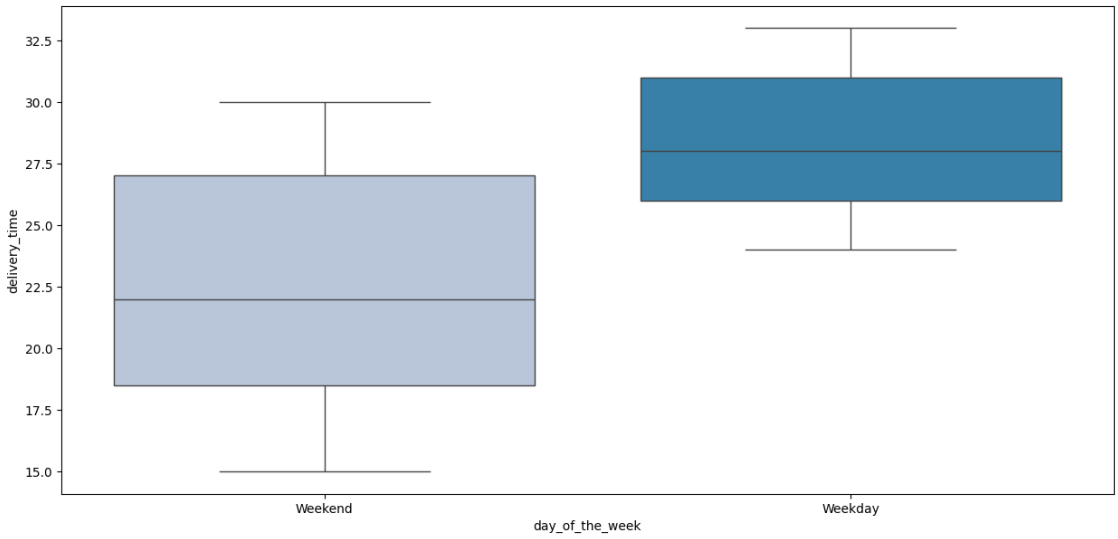
Cuisine vs Food Preparation time

[30]:
```
# Relationship between food preparation time and cuisine type
plt.figure(figsize=(15,7))
sns.boxplot(x = "cuisine_type", y = "food_preparation_time", data = df, palette
 ↪= 'PuBu', hue = "cuisine_type")  ## Complete the code to visualize the
 ↪relationship between food preparation time and cuisine type using boxplot
plt.xticks(rotation = 60)
plt.show()
```



Day of the Week vs Delivery time

[31]:
```
# Relationship between day of the week and delivery time
plt.figure(figsize=(15,7))
sns.boxplot(x = "day_of_the_week", y = "delivery_time", data = df, palette =␣
  ↪'PuBu', hue = "day_of_the_week")  ## Complete the code to visualize the␣
  ↪relationship between day of the week and delivery time using boxplot
plt.show()
```



Run the below code and write your observations on the revenue generated by the restaurants.

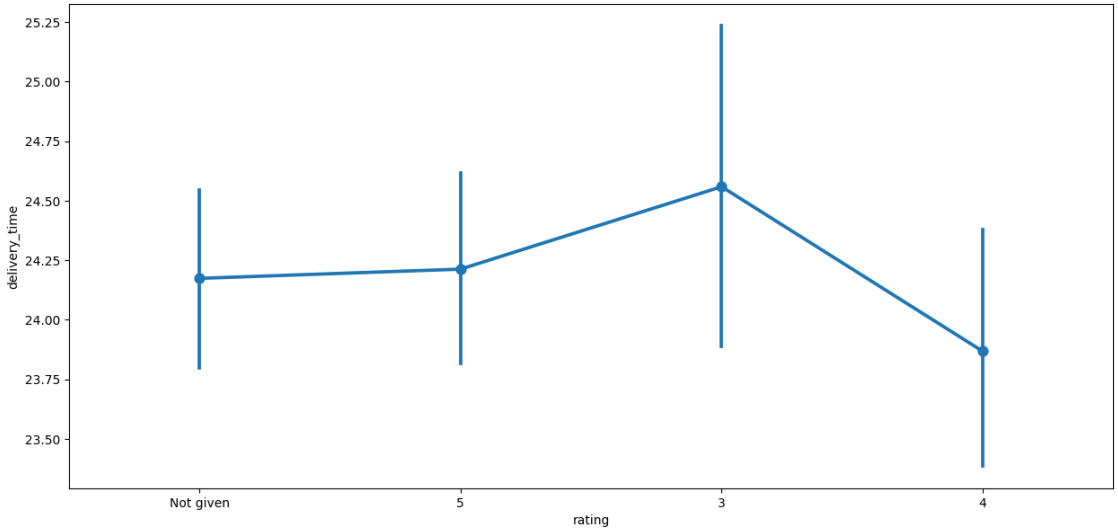
[32]:
```
df.groupby(['restaurant_name'])['cost_of_the_order'].sum().
  ↪sort_values(ascending = False).head(14)
```

[32]:
```
restaurant_name
Shake Shack                      3579.53
The Meatball Shop                2145.21
Blue Ribbon Sushi                1903.95
Blue Ribbon Fried Chicken        1662.29
Parm                             1112.76
RedFarm Broadway                  965.13
RedFarm Hudson                    921.21
TAO                               834.50
Han Dynasty                       755.29
Blue Ribbon Sushi Bar & Grill     666.62
Rubirosa                          660.45
Sushi of Gari 46                  640.87
```

```
Nobu Next Door                    623.67
Five Guys Burgers and Fries       506.47
Name: cost_of_the_order, dtype: float64
```

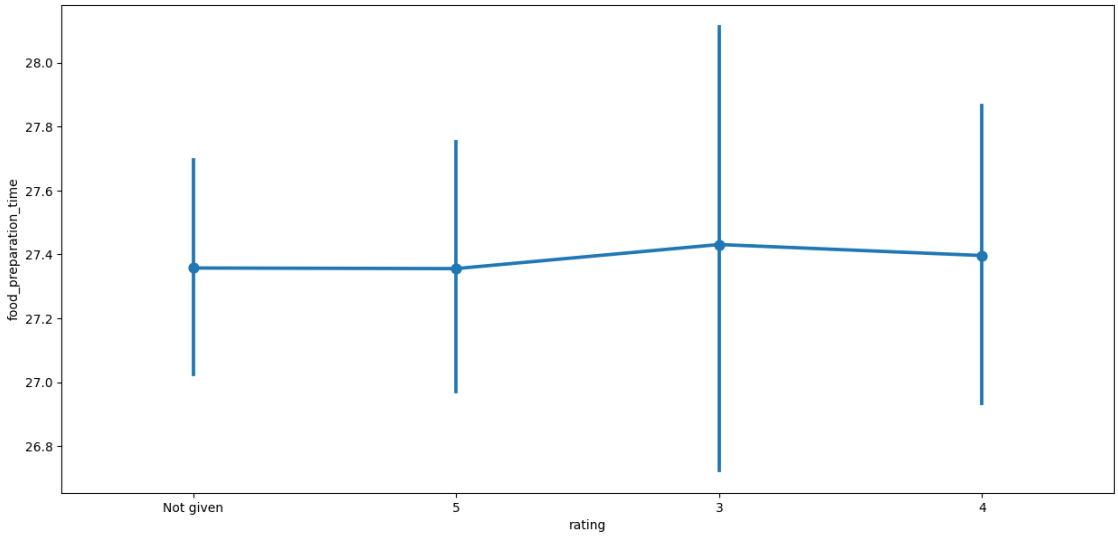
Rating vs Delivery time

[33]:
```
# Relationship between rating and delivery time
plt.figure(figsize=(15, 7))
sns.pointplot(x = 'rating', y = 'delivery_time', data = df)
plt.show()
```



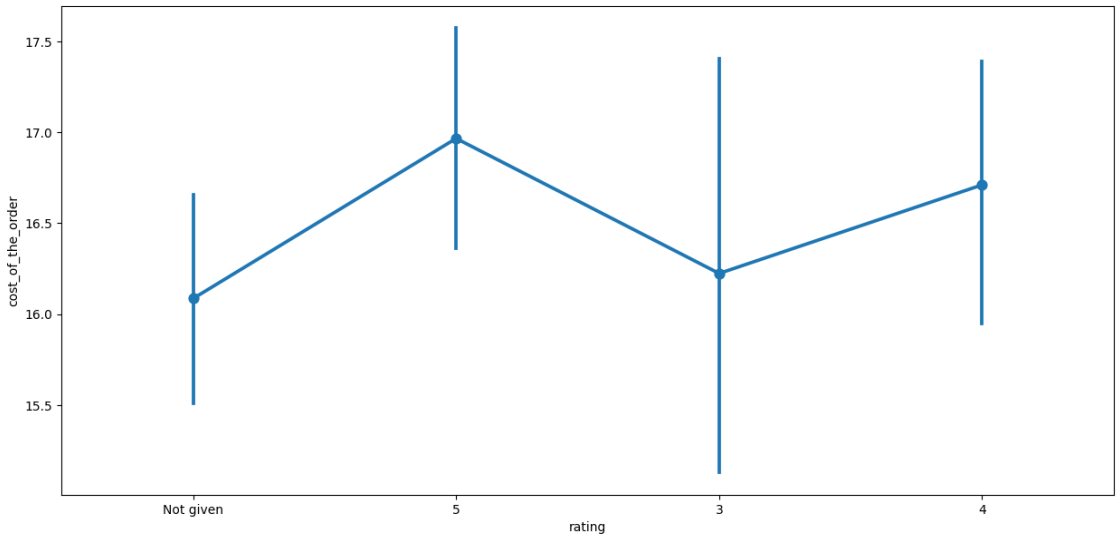
Rating vs Food preparation time

[34]:
```
# Relationship between rating and food preparation time
plt.figure(figsize=(15, 7))
sns.pointplot(x = 'rating', y = 'food_preparation_time', data = df)  ##␣
 ↪Complete the code to visualize the relationship between rating and food␣
 ↪preparation time using pointplot
plt.show()
```



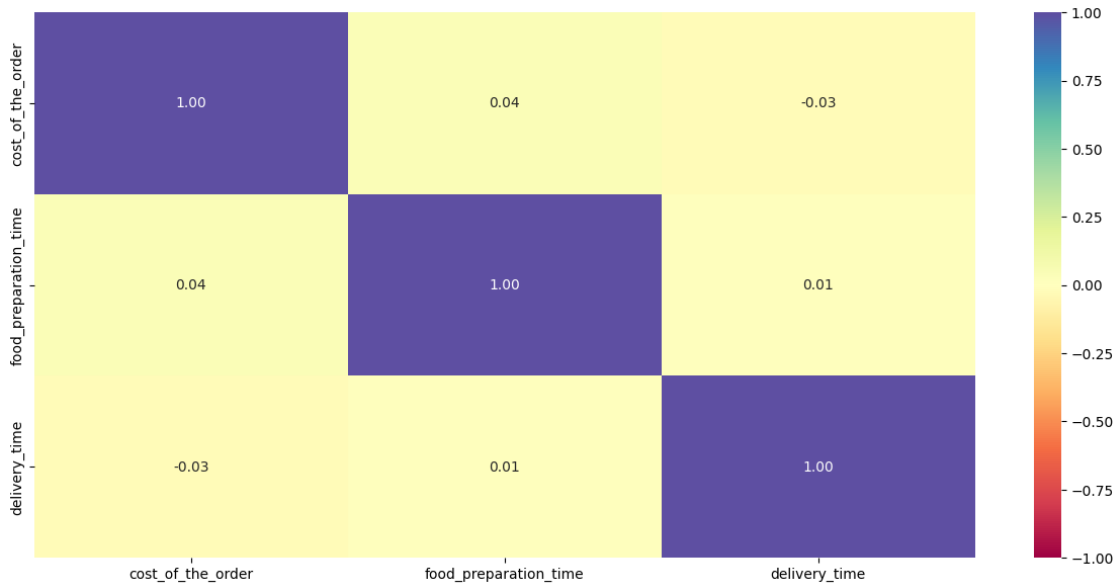
Rating vs Cost of the order

[35]:
```
# Relationship between rating and cost of the order
plt.figure(figsize=(15, 7))
sns.pointplot(x = 'rating', y = 'cost_of_the_order', data = df)    ## Complete␣
↪the code to visualize the relationship between rating and cost of the order␣
↪using pointplot
plt.show()
```



Correlation among variables

[36]:
```
# Plot the heatmap
col_list = ['cost_of_the_order', 'food_preparation_time', 'delivery_time']
plt.figure(figsize=(15, 7))
sns.heatmap(df[col_list].corr(), annot=True, vmin=-1, vmax=1, fmt=".2f",␣
 ↪cmap="Spectral")
plt.show()
```



1.0.23 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]

[37]:
```
# Filter the rated restaurants
df_rated = df[df['rating'] != 'Not given'].copy()

# Convert rating column from object to integer
df_rated['rating'] = df_rated['rating'].astype('int')

# Create a dataframe that contains the restaurant names with their rating counts
df_rating_count = df_rated.groupby(['restaurant_name'])['rating'].count().
 ↪sort_values(ascending = False).reset_index()
df_rating_count.head()
```

[37]:

	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
```

[40]:
```
# Get the restaurant names that have rating count more than 50
rest_names = df_rating_count[df_rating_count['rating']>50]['restaurant_name']␣
 ↪## Complete the code to get the restaurant names having rating count more␣
 ↪than 50

# Filter to get the data of restaurants that have rating count more than 50
df_mean_4 = df_rated[df_rated['restaurant_name'].isin(rest_names)].copy()

# Group the restaurant names with their ratings and find the mean rating of␣
 ↪each restaurant
df_mean_4_rating = df_mean_4.groupby(['restaurant_name'])['rating'].mean().
 ↪sort_values(ascending = False).reset_index().dropna() ## Complete the code␣
 ↪to find the mean rating

# filter for average rating greater than 4
df_avg_rating_greater_than_4 = df_mean_4_rating[df_mean_4_rating['rating'] > 4].
 ↪sort_values(by='rating', ascending=False).reset_index(drop=True)  ##␣
 ↪Complete the code to find restaurants with rating > 4

df_avg_rating_greater_than_4
```

[40]:
```
             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
```

1.0.24 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]

[41]:
```
#function to determine the revenue
def compute_rev(x):
    if x > 20:
        return x*0.25
    elif x > 5:
        return x*0.15
    else:
        return x*0

df['Revenue'] = df['cost_of_the_order'].apply(compute_rev) ## Write the␣
 ↪apprpriate column name to compute the revenue
```

```
df.head()
```

[41]:
```
   order_id  customer_id           restaurant_name cuisine_type  \
0   1477147       337525                   Hangawi       Korean
1   1477685       358141  Blue Ribbon Sushi Izakaya     Japanese
2   1477070        66393               Cafe Habana      Mexican
3   1477334       106968  Blue Ribbon Fried Chicken     American
4   1478249        76942          Dirty Bird to Go     American

   cost_of_the_order day_of_the_week     rating  food_preparation_time  \
0              30.75         Weekend  Not given                     25
1              12.08         Weekend  Not given                     25
2              12.23         Weekday          5                     23
3              29.20         Weekend          3                     25
4              11.59         Weekday          4                     25

   delivery_time  Revenue
0             20   7.6875
1             23   1.8120
2             28   1.8345
3             15   7.3000
4             24   1.7385
```

[42]:
```
# get the total revenue and print it
total_rev = df['Revenue'].sum() ## Write the appropriate function to get the␣
 ↪total revenue
print('The net revenue is around', round(total_rev, 2), 'dollars')
```

```
The net revenue is around 6166.3 dollars
```

1.0.25 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]

[43]:
```
# Calculate total delivery time and add a new column to the dataframe df to␣
 ↪store the total delivery time
df['total_time'] = df['food_preparation_time'] + df['delivery_time']


## Write the code below to find the percentage of orders that have more than 60␣
 ↪minutes of total delivery time (see Question 9 for reference)
# Get orders that take longer than 60 minutes to deliver
df_greater_than_60 = df[df['total_time']>60] ## Write the appropriate column␣
 ↪name to get orders that take longer than 60 minutes to deliver
```

```
# Calculate the number of total orders that take longer than 60 minutes to␣
 ↪deliver
print('The number of total orders that take longer than 60 minutes to deliver␣
 ↪is:', df_greater_than_60.shape[0])

# Calculate percentage of such orders in the dataset
percentage = (df_greater_than_60.shape[0] / df.shape[0]) * 100

print("Percentage of orders that take longer than 60 minutes to deliver is:",␣
 ↪round(percentage, 2), '%')
```

```
The number of total orders that take longer than 60 minutes to deliver is: 200
Percentage of orders that take longer than 60 minutes to deliver is: 10.54 %
```

1.0.26 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]

[44]:
```
# Get the mean delivery time on weekdays and print it
print('The mean delivery time on weekdays is around',
      round(df[df['day_of_the_week'] == 'Weekday']['delivery_time'].mean()),
     'minutes')


## Write the code below to get the mean delivery time on weekends and print it
print('The mean delivery time on weekends is around',
      round(df[df['day_of_the_week'] == 'Weekend']['delivery_time'].mean()),
     'minutes')
```

```
The mean delivery time on weekdays is around 28 minutes
The mean delivery time on weekends is around 22 minutes
```

1.0.27 Conclusion and Recommendations

1.0.28 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]

1.0.29 Conclusions:

-

1.0.30 Recommendations:

-
