FoodHub Data Analysis

Python Foundations : PGP-DSBA

Dec 16,2022

Contents / Agenda

- Business Problem Overview and Solution Approach
- Data Overview
- EDA - Univariate Analysis
- EDA - Multivariate Analysis
- Executive Summary

Business Problem Overview and Solution Approach

- Business Problem :

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.

- Solution Approach / Methodology:

Providing proper offers to attract customer to provide feedback. Proper promotions during weekdays to enhance the revenue. For restaurants that have least customers, make sure to analyze factors such as quality of food or variety of food options or other factors through customer feedback.

Data Overview

- Data of first five rows of the dataset

```
In [8]: df = pd.read_csv('foodhub_order.csv')
        df.head()
```

Out[8]:

	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

- Shape of the dataset

```
In [9]: df.shape
Out[9]: (1898, 9)
```

-> The dataset has 1898 rows and 9 columns.

Data Overview

- Data information

```
In [10]: 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
```

-> 4 object datatype and 5 numerical datatype.

-> Every column has 1898 observation.

Data Overview

- Missing Values

```
In [11]: df.isnull().sum()

Out[11]: 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
```

-> There are no missing values in the dataset.

Data Overview

- Statistical Summary

```
In [12]: df.describe()
```

Out[12]:

	order_id	customer_id	cost_of_the_order	food_preparation_time	delivery_time
count	1.898000e+03	1898.000000	1898.000000	1898.000000	1898.000000
mean	1.477496e+06	171168.478398	16.498851	27.371970	24.161749
std	5.480497e+02	113698.139743	7.483812	4.632481	4.972637
min	1.476547e+06	1311.000000	4.470000	20.000000	15.000000
25%	1.477021e+06	77787.750000	12.080000	23.000000	20.000000
50%	1.477496e+06	128600.000000	14.140000	27.000000	25.000000
75%	1.477970e+06	270525.000000	22.297500	31.000000	28.000000
max	1.478444e+06	405334.000000	35.410000	35.000000	33.000000

```
In [13]: df.describe().T
```

Out[13]:

	count	mean	std	min	25%	50%	75%	max
order_id	1898.0	1.477496e+06	548.049724	1476547.00	1477021.25	1477495.50	1.477970e+06	1478444.00
customer_id	1898.0	1.711685e+05	113698.139743	1311.00	77787.75	128600.00	2.705250e+05	405334.00
cost_of_the_order	1898.0	1.649885e+01	7.483812	4.47	12.08	14.14	2.229750e+01	35.41
food_preparation_time	1898.0	2.737197e+01	4.632481	20.00	23.00	27.00	3.100000e+01	35.00
delivery_time	1898.0	2.416175e+01	4.972637	15.00	20.00	25.00	2.800000e+01	33.00

```
In [14]: df.describe(exclude="number").T
```

Out[14]:

	count	unique	top	freq
restaurant_name	1898	178	Shake Shack	219
cuisine_type	1898	14	American	584
day_of_the_week	1898	2	Weekend	1351
rating	1898	4	Not given	736

-> Time to prepare the food after customer place order is 20mins - 35mins.

-> Cost of the order ranges between 4.47(minimum) - 35.41(maximum)

->Time taken to deliver is between 15mins - 33mins

Data Overview

- Rating count of orders

```
In [15]: df['rating'].value_counts()
Out[15]: Not given    736
         5            588
         4            386
         3            188
         Name: rating, dtype: int64
```

-> 736 orders are not rated

Univariate Analysis

- Unique Order ID

Order ID

```
In [16]: # check unique order ID
         df['order_id'].nunique()
Out[16]: 1898
```

- Unique Customer ID

Customer ID

```
In [17]: # check unique customer ID
         df['customer_id'].nunique()
Out[17]: 1200
```

- Unique restaurant name

Restaurant name

```
In [18]: # check unique Restaurant Name
         df['restaurant_name'].nunique()
Out[18]: 178
```

Univariate Analysis

- Unique cuisine type

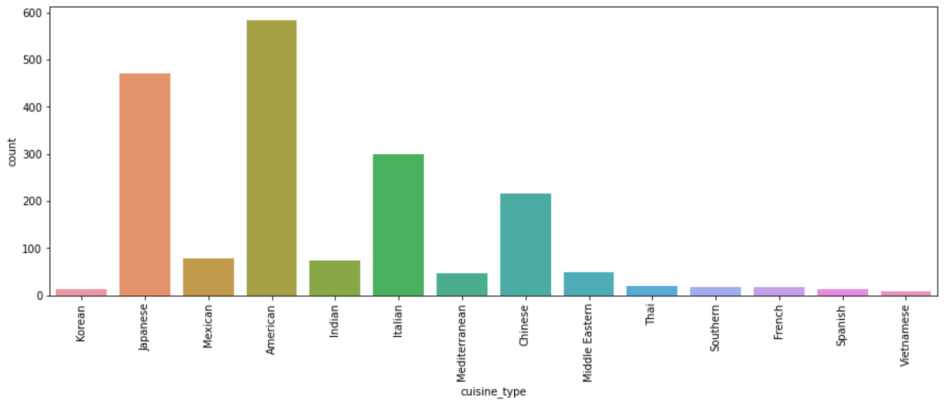
Cuisine type

```
In [19]: # Check unique cuisine type
         df['cuisine_type'].nunique()
```

```
Out[19]: 14
```

- Observation cuisine type

```
In [20]: plt.figure(figsize = (15,5))
         sns.countplot(data = df, x = 'cuisine_type');## Create a countplot for cuisine type.
         plt.xticks(rotation=90);
```



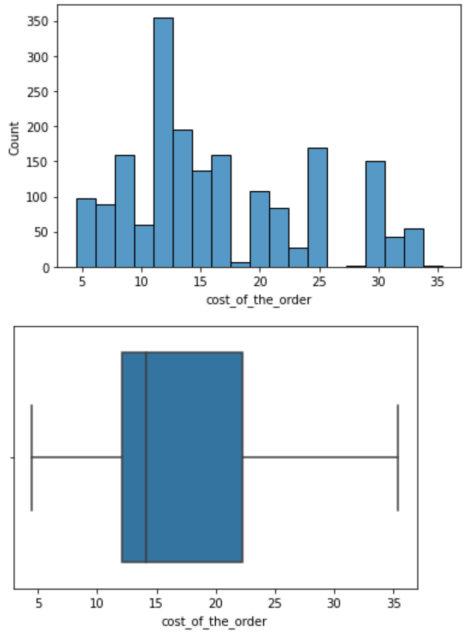
-> American cuisine is the most ordered cuisine

-> Vietnamese is the least ordered cuisine

Univariate Analysis

- Observation on “Cost of the order”

```
In [21]: sns.histplot(data=df,x='cost_of_the_order') ## His
         plt.show()
         sns.boxplot(data=df,x='cost_of_the_order') ## Boxp
         plt.show()
```



->Most of the order cost is approximately 12 dollars.

-> Minimum and Maximum cost of the order are 5 dollars and 35 dollars.

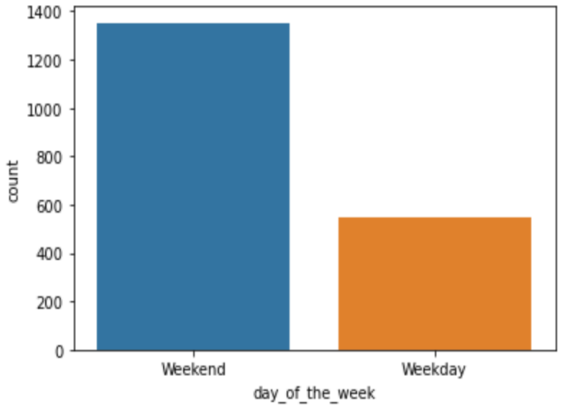
Univariate Analysis

- Observation on day of the week : Majority of the orders are placed on weekend.

```
In [22]: # # Check the unique values
         df['day_of_the_week'].nunique()

Out[22]: 2
```

```
In [23]: sns.countplot(data = df, x = 'day_of_the_week'); ##
```



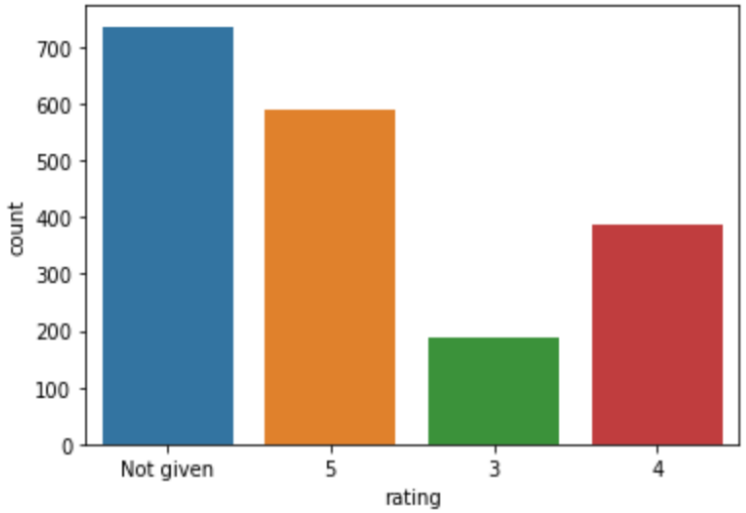
Univariate Analysis

- Observation on the rating : Most of the orders did not recieve rating

```
In [24]: # Check the unique val
         df['rating'].nunique()

Out[24]: 4
```

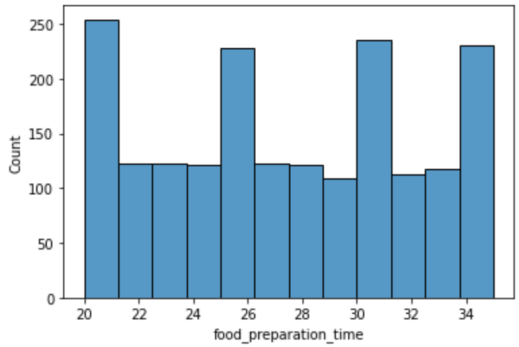
```
In [25]: sns.countplot(data = df, x = 'rating'); ## Comple
```

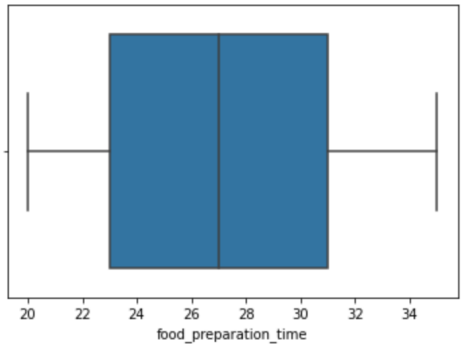


Univariate Analysis

- Observation for food preparation time: There is no skewness in the graph

```
In [26]: sns.histplot(data=df,x='food_preparation_time') ##
         plt.show()
         sns.boxplot(data=df,x='food_preparation_time') ##
         plt.show();
```

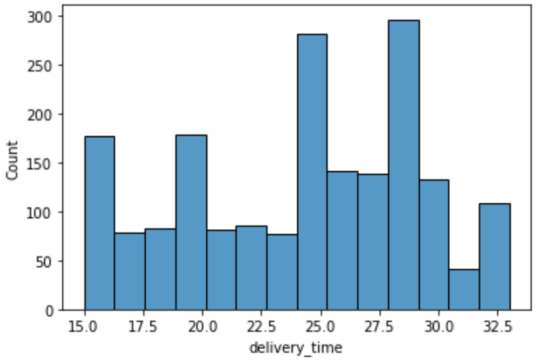


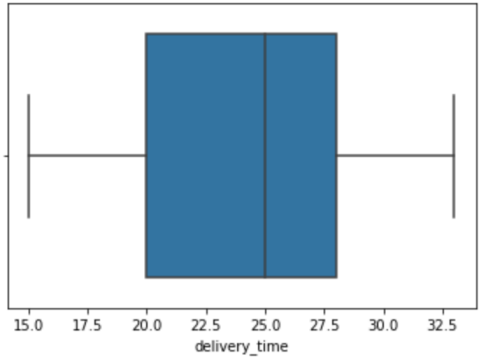


Univariate Analysis

- Observation for delivery time : The distribution for delivery time is left skewed.

```
In [27]: sns.histplot(data=df,x='delivery_time') ## Complet
         plt.show()
         sns.boxplot(data=df,x='delivery_time') ## Complete
         plt.show()
```





Univariate Analysis

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

In [28]:
```
# Get top 5 restaurants with highest number of orders
df['restaurant_name'].value_counts() ## Complete the code
```

Out[28]:
```
Shake Shack                  219
The Meatball Shop            132
Blue Ribbon Sushi            119
Blue Ribbon Fried Chicken     96
Parm                          68
                            ...
Sushi Choshi                   1
Dos Caminos Soho               1
La Follia                      1
Philippe Chow                  1
'wichcraft                     1
Name: restaurant_name, Length: 178, dtype: int64
```

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

In [29]:
```
# 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 va
```

Out[29]:
```
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: cuisine_type, dtype: int64
```

Univariate Analysis

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

In [30]:
```
# Get orders that cost above 20 dollars
df_greater_than_20 = df[df['cost_of_the_order']>20] ## Write the appropriate column name to ge

# 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 %
```

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

In [31]:
```
# Get the mean delivery time
mean_del_time = df['delivery_time'].mean()  ## Write the appropriate fun

print('The mean delivery time for this dataset is', round(mean_del_time,
```

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

Question 11: The company has decided to give 20% discount vouchers to the top 3 most frequent customers. Find the IDs of these customers and the number of orders they placed. [1 mark]

In [32]:
```
# Get the counts of  each customer_id
df['customer_id'].value_counts().head(3)  ## Write the appropriate column name to get the top 3 cmost frequent customers
```

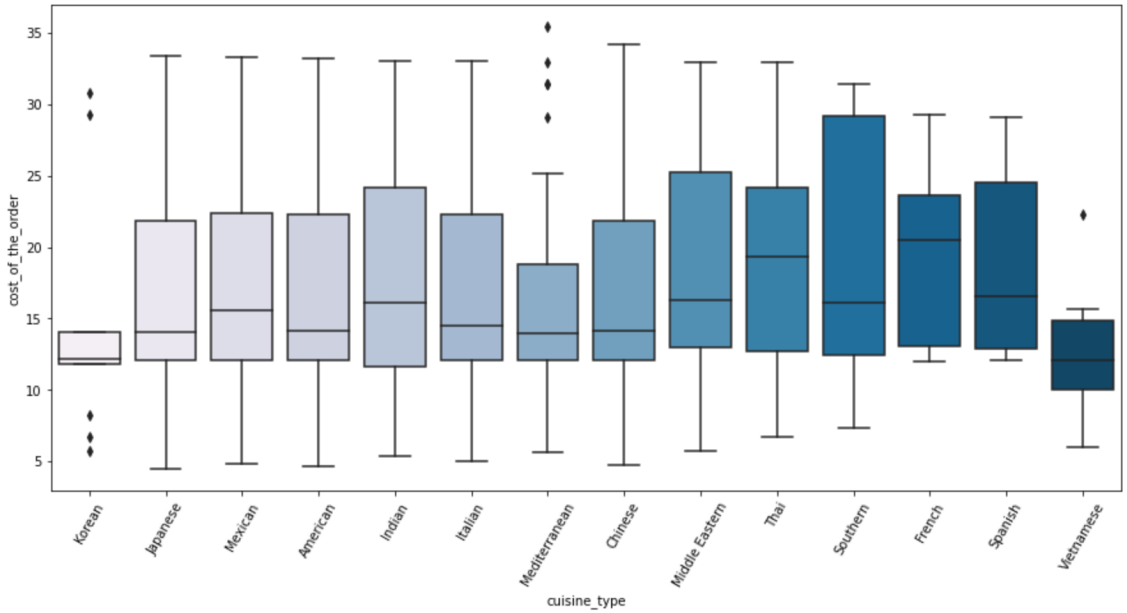
Out[32]:
```
52832    13
47440    10
83287     9
Name: customer_id, dtype: int64
```

Multivariate Analysis

- Relationship between cost of the order and cuisine type

Cuisine vs Cost of the order

```
In [33]: # 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')
plt.xticks(rotation = 60)
plt.show()
```

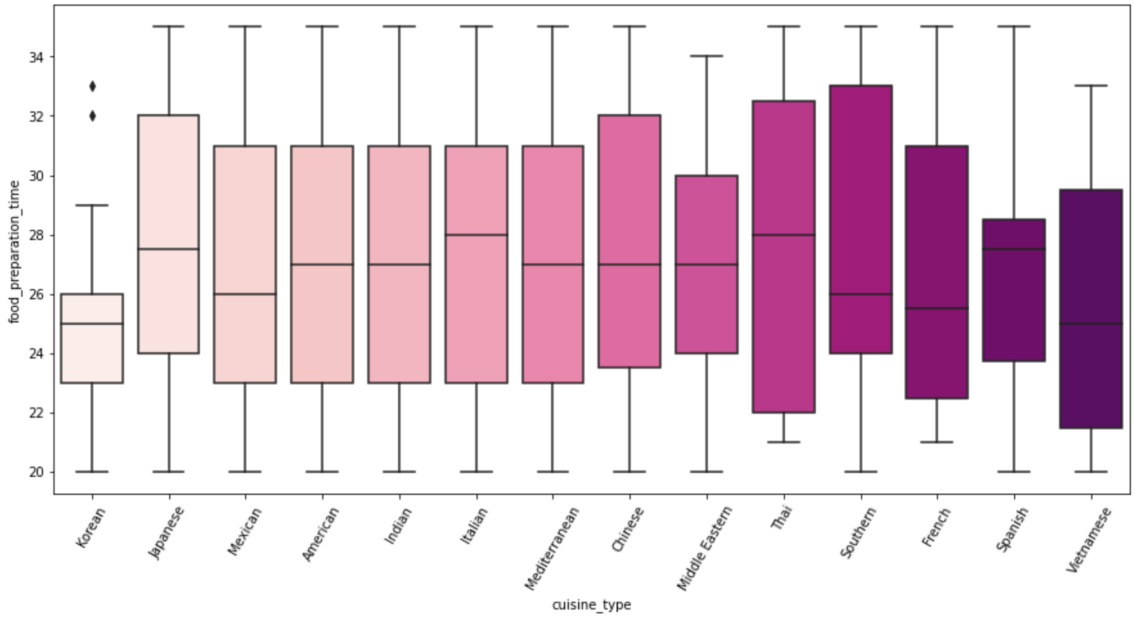


Multivariate Analysis

- Relationship between cuisine type and food preparatiob time

Cuisine vs Food Preparation time

```
In [34]: # 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 = 'RdPu')  ## Complete the code to visu
plt.xticks(rotation = 60)
plt.show()
```

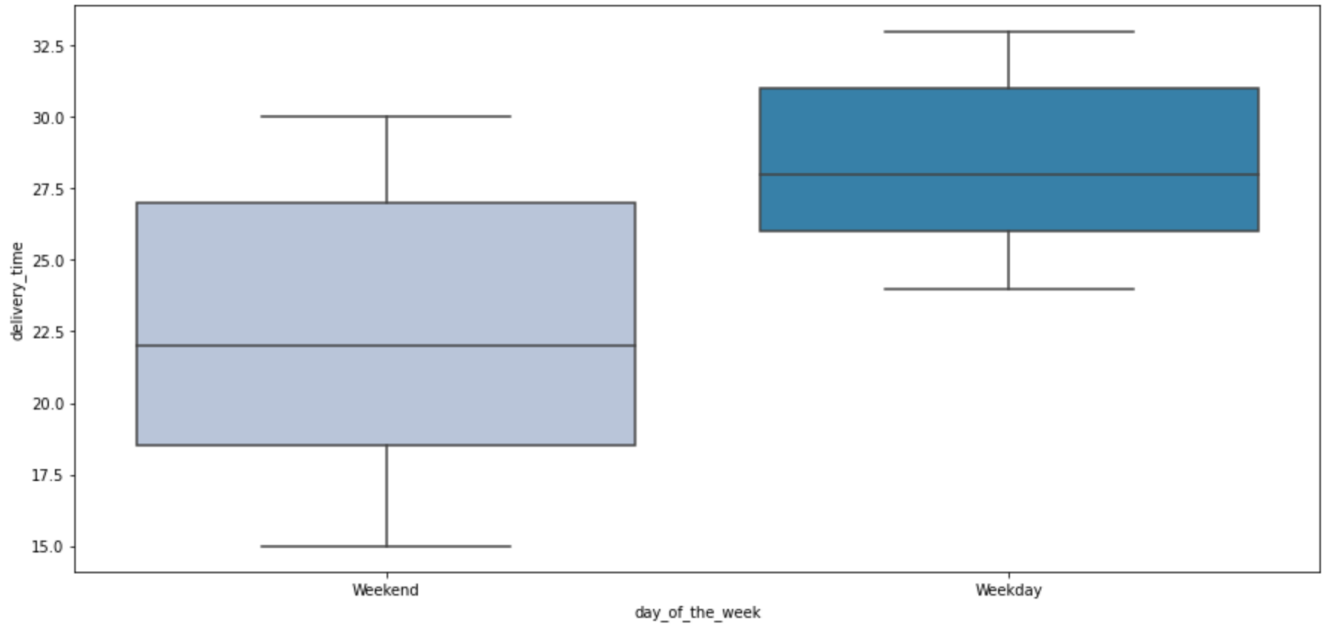


Multivariate Analysis

- Relationship between day of the week and delivery time

Day of the Week vs Delivery time

In [35]:
```
# 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')  ## Complete the code to visua
plt.show()
```



Multivariate Analysis

- Observation on the revenue generated by the restaurants : Highest revenues was generated by Shack Shack

```
In [36]: df.groupby(['restaurant_name'])['cost_of_the_order'].sum().sort_values(ascending = False).head(14)
```

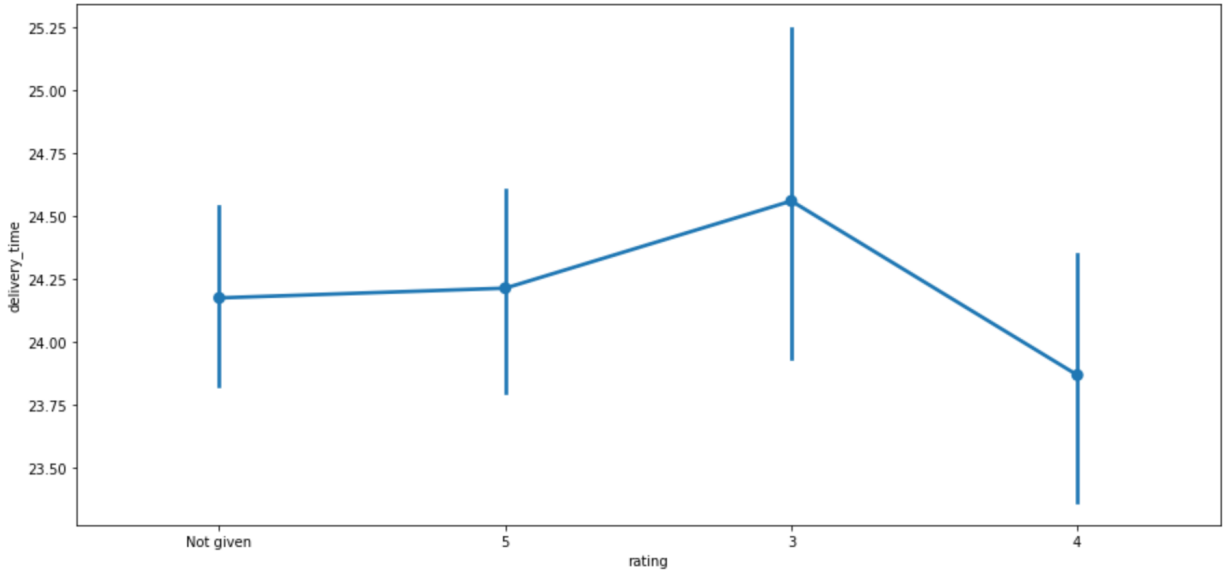
```
Out[36]: restaurant_name
         Shake Shack                      3579.53
         The Meatball Shop                2145.21
         Blue Ribbon Sushi                1903.95
         Blue Ribbon Fried Chicken        1662.29
         Parm                             1112.76
         RedFarm Broadway                  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
```

Multivariate Analysis

- Relationship between rating and delivery time : Customer gave low rating when delivery time is high

Rating vs Delivery time

```
In [37]: # Relationship between rating and delivery time
         plt.figure(figsize=(15, 7))
         sns.pointplot(x = 'rating', y = 'delivery_time', data = df)
         plt.show()
```

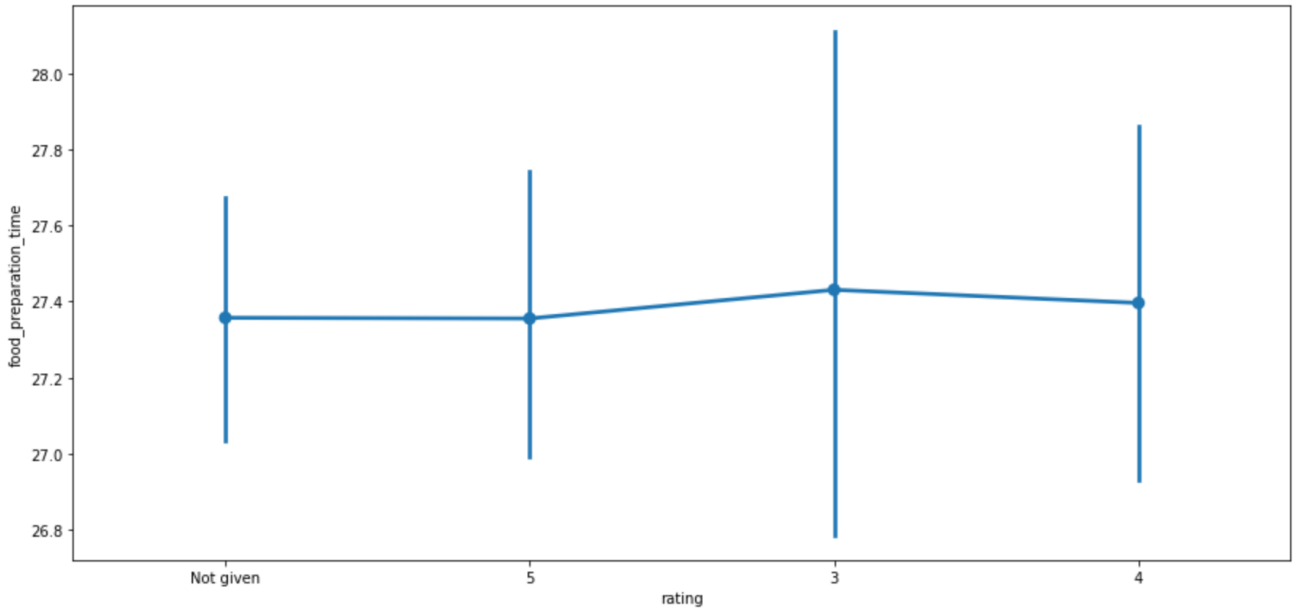


Multivariate Analysis

- Relationship between rating and food preparation time : customer satisfaction is lower when food preparation time is high

Rating vs Food preparation time

```
In [38]: # 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 relatic
plt.show()
```

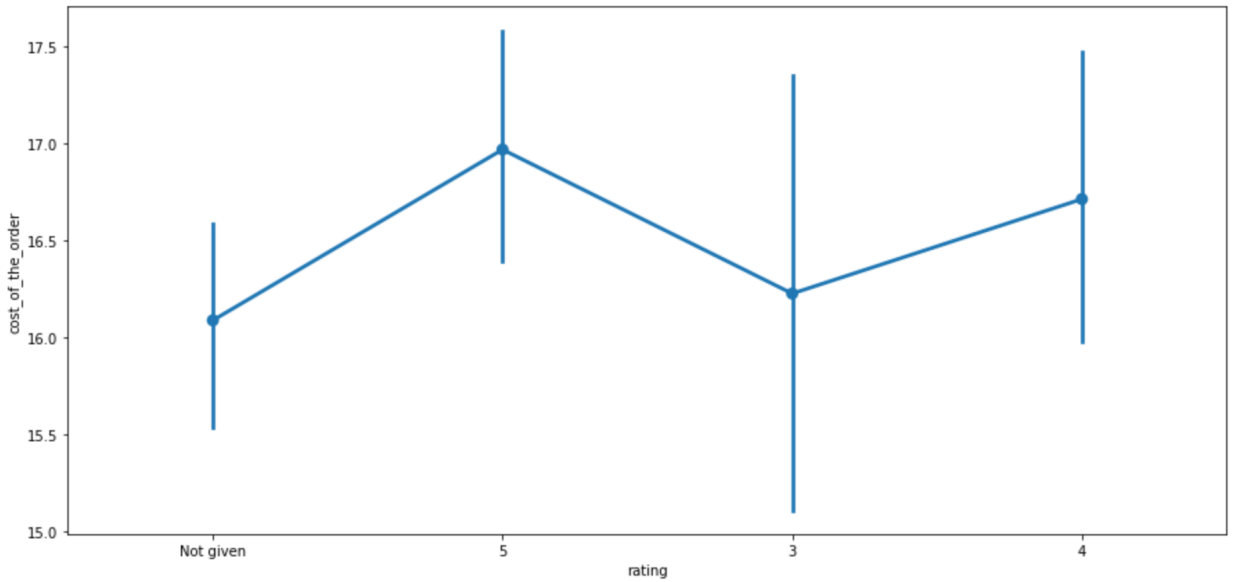


Multivariate Analysis

- Relationship between rating and cost of the order : Higher cost of order leads to higher ratings

Rating vs Cost of the order

```
In [39]: # 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 bet
         plt.show()
```

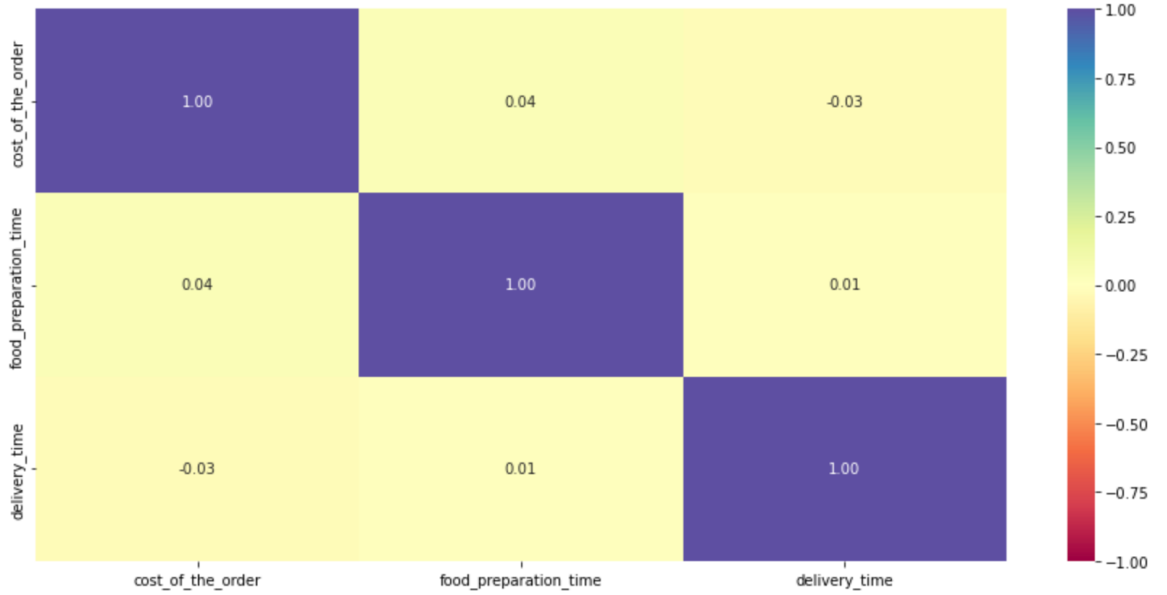


Multivariate Analysis

- Heatmap for correlation among variables

Correlation among variables

In [40]:
```
# 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()
```



Multivariate Analysis

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 [41…

```
# 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()
```

Out[41]:

	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

Multivariate Analysis

```
In [...
# Get the restaurant names that have rating count more than 50
rest_names = df_rating_count[df_rating_count['rating']>4]['restaurant_name'] ## Complete the code to get the re

# 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.groupby(['restaurant_name'])['rating'].mean().sort_values(ascending = False).reset_index().dropna() #
```

Out[42]:

	restaurant_name	rating
0	Vanessa's Dumpling House	5.000000
1	Sushi Samba	4.875000
2	Otto Enoteca Pizzeria	4.857143
3	Sushi of Gari	4.714286
4	S'MAC	4.714286
5	The Kati Roll Company	4.700000
6	Vanessa's Dumplings	4.666667
7	Sushi of Gari Tribeca	4.615385
8	Cho Dang Gol	4.600000
9	Blue Ribbon Sushi Bar & Grill	4.590909
10	Westville Hudson	4.555556
11	Five Guys Burgers and Fries	4.555556
12	The Meatball Shop	4.511905
13	Yama Japanese Restaurant	4.500000
14	Han Dynasty	4.434783
15	J. G. Melon	4.416667
16	The Smile	4.416667
17	Tamarind TriBeCa	4.400000
18	Hill Country Fried Chicken	4.363636
19	TAO	4.357143
20	Nobu Next Door	4.347826
21	Empanada Mama (closed)	4.333333
22	Melt Shop	4.333333
23	Blue Ribbon Sushi Izakaya	4.333333
24	Blue Ribbon Fried Chicken	4.328125
25	Jack's Wife Freda	4.315789
26	P.J. Clarke's	4.300000
27	Xi'an Famous Foods	4.285714
28	Shake Shack	4.278195
29	Momoya	4.272727
30	Cafe Habana	4.272727
31	Burger Joint	4.250000
32	RedFarm Broadway	4.243902
33	Sushi of Gari 46	4.235294
34	Blue Ribbon Sushi	4.219178
35	Balthazar Boulangerie	4.200000
36	Chipotle Mexican Grill $1.99 Delivery	4.200000
37	RedFarm Hudson	4.176471
38	Cafe Mogador	4.153846
39	Ilili Restaurant	4.153846
40	Parm	4.128205
41	Rubirosa	4.125000
42	Bareburger	4.058824
43	Pepe Rosso To Go	4.000000
44	Osteria Morini	4.000000
45	Hatsuhana	4.000000

Multivariate Analysis

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 [...]:
```
#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()
```

Out[43]:

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week	rating	food_preparation_time	delivery_time	Revenue
0	1477147	337525	Hangawi	Korean	30.75	Weekend	Not given	25	20	7.6875
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend	Not given	25	23	1.8120
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday	5	23	28	1.8345
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend	3	25	15	7.3000
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday	4	25	24	1.7385

In [...]:
```
# get the total revenue and print it
total_rev = df['Revenue'].sum() ## Write the appropriate function t
print('The net revenue is around', round(total_rev, 2), 'dollars')
```

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

Observation:

Total revenue for orders having cost greater than 20 dollars is 6166.3 dollars

Multivariate Analysis

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 [...

```
# 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)
df_greater_than_60 = df[df['total_time']>60]

print('The number of total orders that takes above 60 minutes is:', df_greater_than_60.shape[0])

percentage = (df_greater_than_60.shape[0] / df.shape[0]) * 100

print("Percentage of orders that takes more than 60 minutes:", round(percentage, 2), '%')
```

```
The number of total orders that takes above 60 minutes is: 200
Percentage of orders that takes more than 60 minutes: 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 [...

```
# 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 Weekend 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 Weekend is around 22 minutes
```

Executive Summary

Conclusions:

- Most orders are placed during the weekends.
- Approximately 38% of orders are unrated.
- Increased food prices are correlated with higher customer satisfaction.
- The most orders are placed at American, Japanese, Italian, and Chinese restaurants, which are the most well-liked.

Executive Summary

Recommendations:

- Proper offers during weekdays can be provided to enhance the revenue
- An increase in cost of order during weekend can increase revenue
- Feedback is an important tool in business to enhanc growth of a company. So, make sure customer give feedback for orders placed. In foodhub case, we can see most customers did not give any feedback.

Great Learning

Happy Learning !