Project Foundations for Data Science: FoodHub Data Analysis

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

```
# Ignoring warnings
import warnings
warnings.filterwarnings('ignore')
```

Double-click (or enter) to edit

```
# 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


def histo(dataframe, col):
    new_df = dataframe
    new_df.fillna({col: df[col].mean()})
    plt.hist(new_df[col], color = "blue", alpha = .8, edgecolor = "g")
    plt.show()
```

Understanding the structure of the data

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

```
Mounted at /content/drive
```

```
# read the data
df = pd.read_csv('/content/drive/MyDrive/foodhub_order.csv')
# returns the first 5 rows
df.head()
```

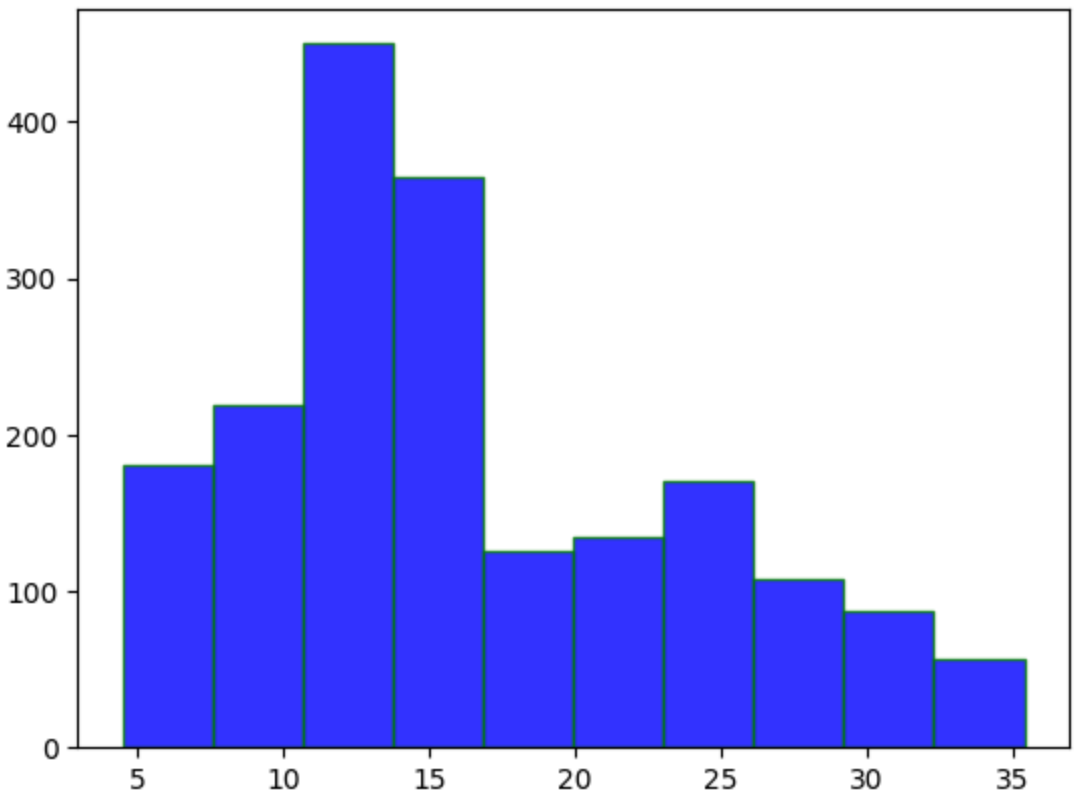
	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of
0	1477147	337525	Hangawi	Korean	30.75	
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	
2	1477070	66393	Cafe Habana	Mexican	12.23	
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	

```
df['Band'] = df.restaurant_name.str.split().str.get(0)
```

Observations:

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

```
histo(df, 'cost_of_the_order')
```



Question 1: How many rows and columns are present in the data?

```
df.shape
```

```
(1898, 9)
```

Observations:

- The dataset has 1898 rows and 9 columns

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

```
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 columns have 1898 observations
- There are no values, meaning each column will always have data
- rating is an 'object' data type as it can contain an integer rating or "Not given" in the case rating isn't given

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

```
df.isna().sum()
```

```
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
```

```
# Replacing "Not given" with 0 rating
new_rating = []
for rate in df.rating:
  if(rate == "Not given"):
    new_rating.append(0)
  else:
    new_rating.append(int(rate))
df['int_rating'] = new_rating
```

Observations:

- There are no missing values as each column has 1898 values

- The rating column would have null values, but it was already taken care of as the data without the values were replaced with "Not given"

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?

```
df['food_preparation_time'].describe()
```

```
count    1898.000000
mean       27.371970
std         4.632481
min        20.000000
25%        23.000000
50%        27.000000
75%        31.000000
max        35.000000
Name: food_preparation_time, dtype: float64
```

Observations:

- Minimum time: 20.00
- Average time: 27.37
- Maximum time: 35.00

```
Start coding or generate with AI.
```

Question 5: How many orders are not rated?

```
df.rating.value_counts()
```

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

Observations:

- 736 orders are not rated.

Exploratory Data Analysis (EDA)

Univariate Analysis

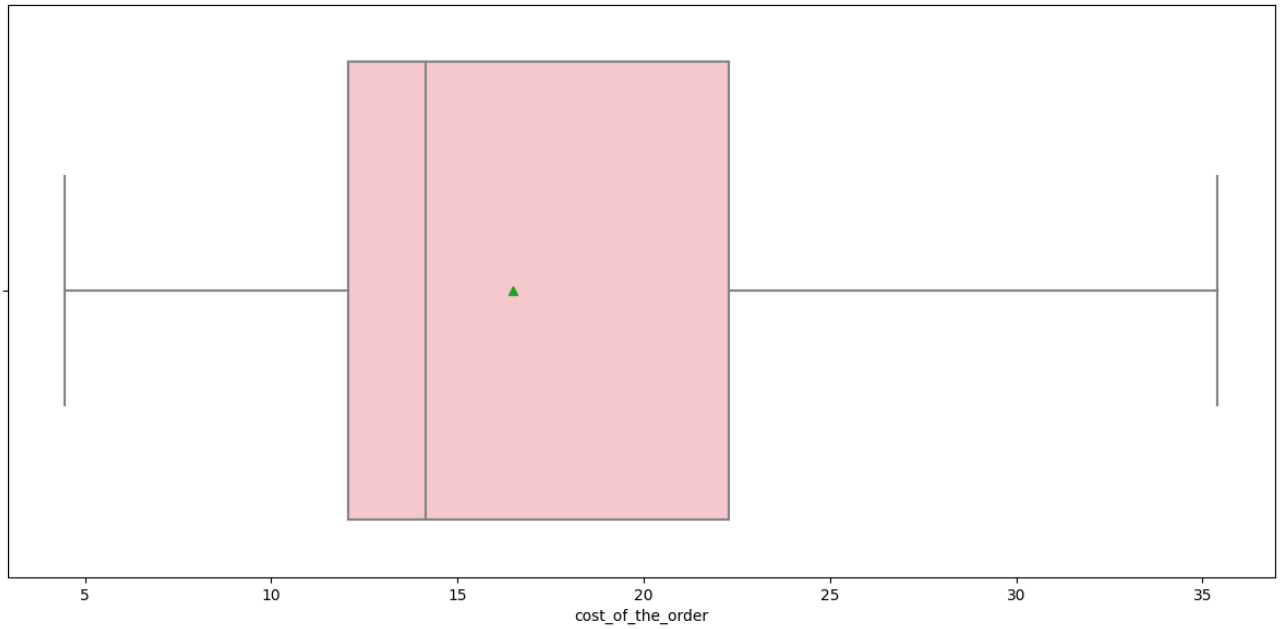
Question 6: Explore all the variables and provide observations on their distributions. (Generally, histograms, boxplots, countplots, etc. are used for univariate exploration)

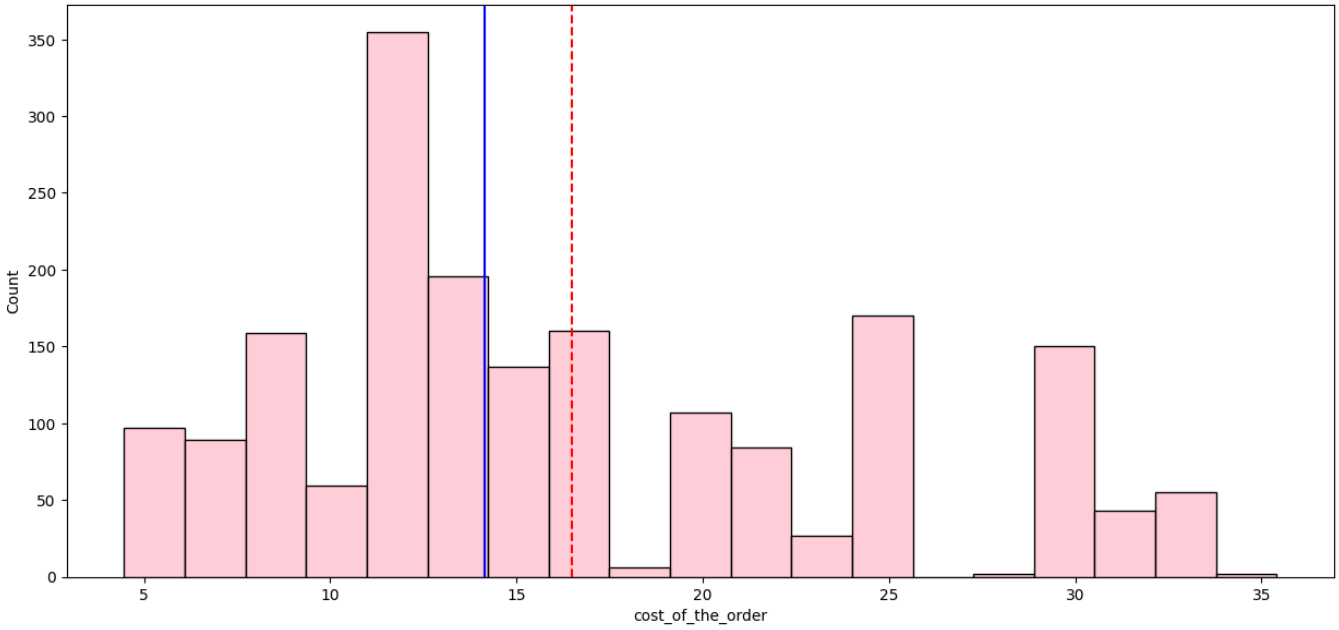
```
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

def histo_boxplot(myData, figsize = (15, 15), bins = "auto"):
    # Boxplot and Histogram function that takes in a 1D array 'myData'
    # and graphs it on a (15, 15) sized graph.
    # The bins is defaulted to "auto" for ease in determining the number of bins
    # needed to graph the histogram.

    f,(ax_box, ax_hist) = plt.subplots(nrows=2, figsize=figsize)
    # The subplot grid will have 2 rows.

    # Subplots
    # Boxplot will show the mean of the data in red
    sns.boxplot(x = myData, ax = ax_box, showmeans = True, color = 'pink')

    # Histogram
    sns.histplot(x = myData, kde=False, ax=ax_hist, bins=bins, color = 'pink')
    # Adding the mean to the histogram
    ax_hist.axvline(np.mean(myData), color ='red', linestyle='--')
    # Adding the median to the histogram
    ax_hist.axvline(np.median(myData), color='blue', linestyle='-')

    # Show graphs
    plt.show()
```

```
def bar_percent(myData, values):
    # Saves the length of the column to a variable 'total'
    total = len(myData[values])
    # Setting the figure size
    plt.figure(figsize = (40, 10))

    # Converts the column to a catergorical datatype
    myData[values] = myData[values].astype('category')

    ax = sns.countplot(x = values, data = myData, palette = 'pastel', order = myData
    # Rotating x Labels by 90 degrees
    plt.xticks(rotation=90)

    for p in ax.patches:
      # Percentage of the data
      percent = '{:.1f}%'.format(100 * p.get_height() / total)\
      # Getting the width of the plot for X and Y coordinates that will be used to p
      x = p.get_x() + p.get_width() / 2
      y = p.get_y() + p.get_height()
      # Adding the percentage onto the plot
      ax.annotate(percent, (x, y), size = 10)

    # Show plot
    plt.show()
```

Observations on cost of the order

```
histo_boxplot(df.cost_of_the_order)
```


5
10
15
20
25
30
35
cost_of_the_order

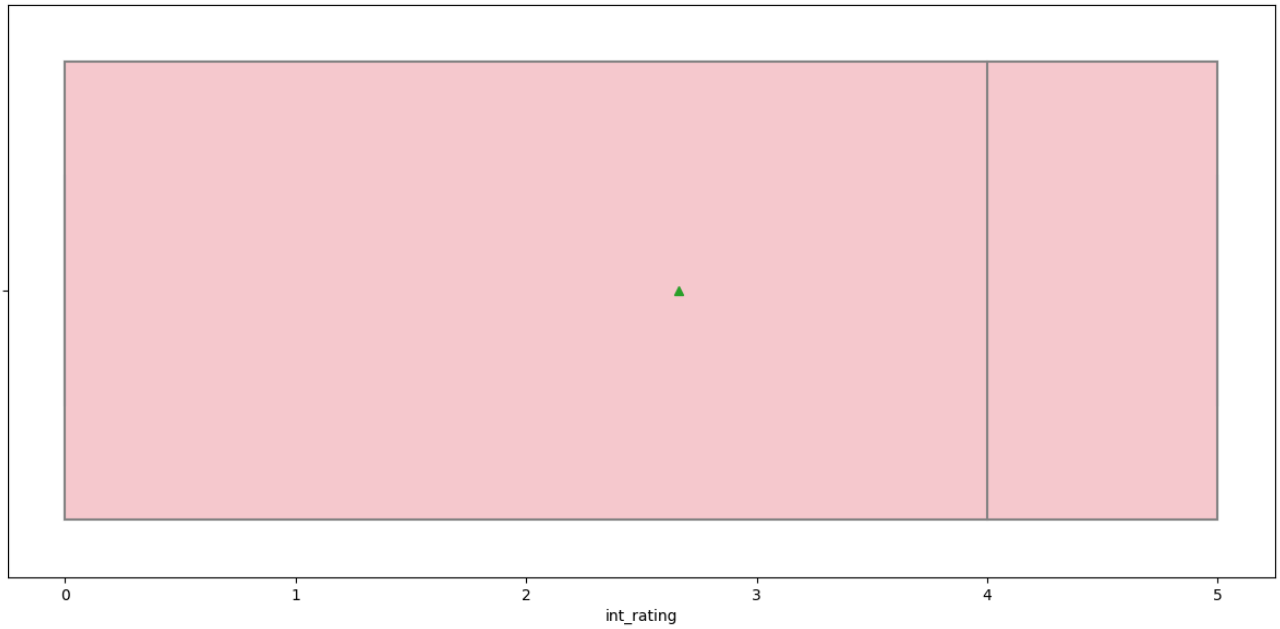

Count
0
50
100
150
200
250
300
350
5
10
15
20
25
30
35
cost_of_the_order

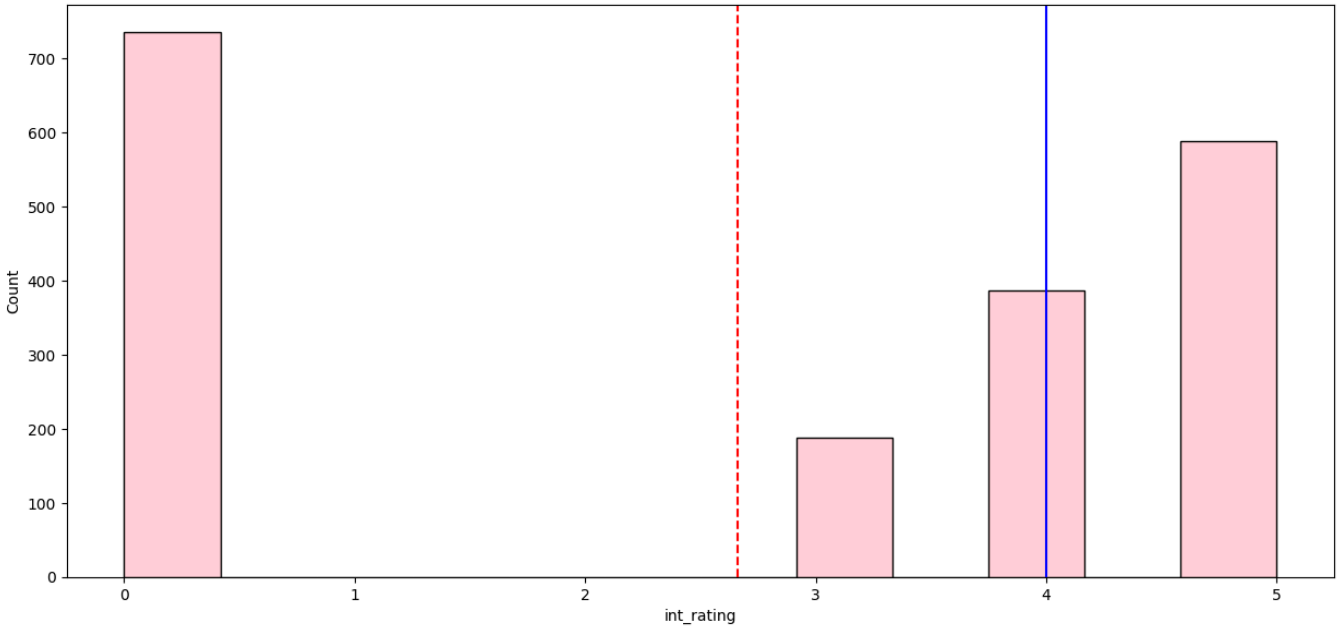
Observations:

- The distribution of 'cost_of_the_order' is right-skewed as seen in both the boxplot and the histogram.
- The mean and the median are centered, meaning most orders can vary, but will generally stay consistant.
- The cost of food will generally be around $10-20, but can vary depending on the time of the day.
- Will need to check the price distribution on weekends compared to weekdays

Observations on the rating

```
histo_boxplot(df.int_rating)
```


0
1
2
3
4
5
int_rating

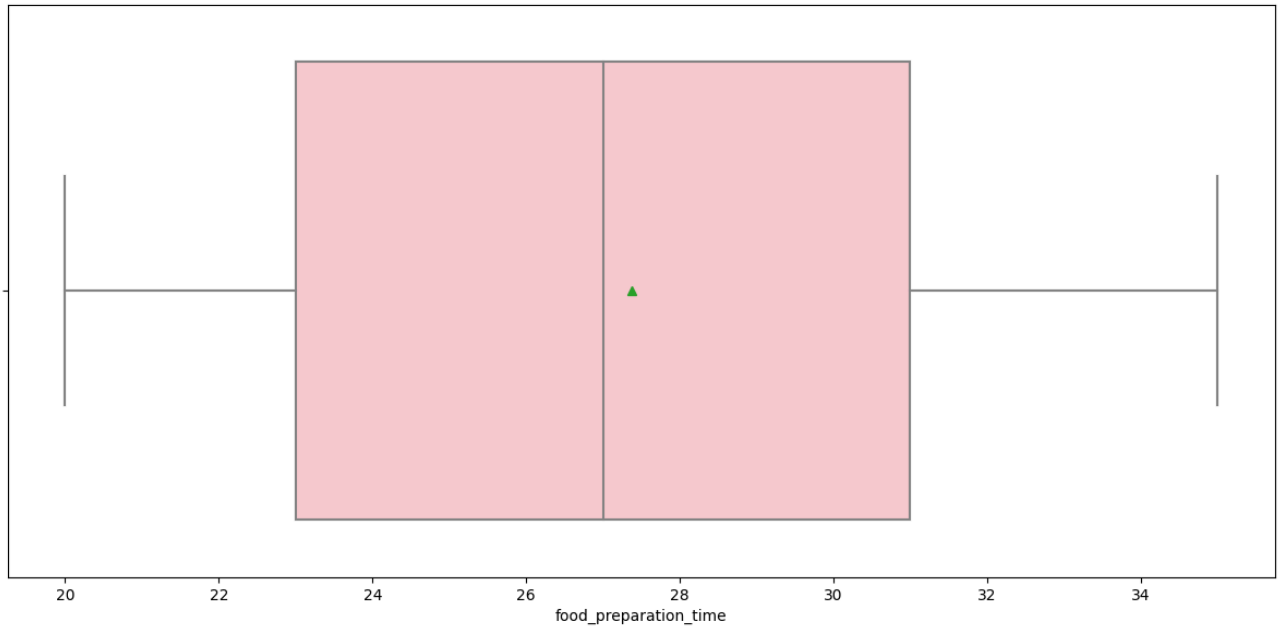

700
600
500
400
300
200
100
0
Count
0
1
2
3
4
5
int_rating

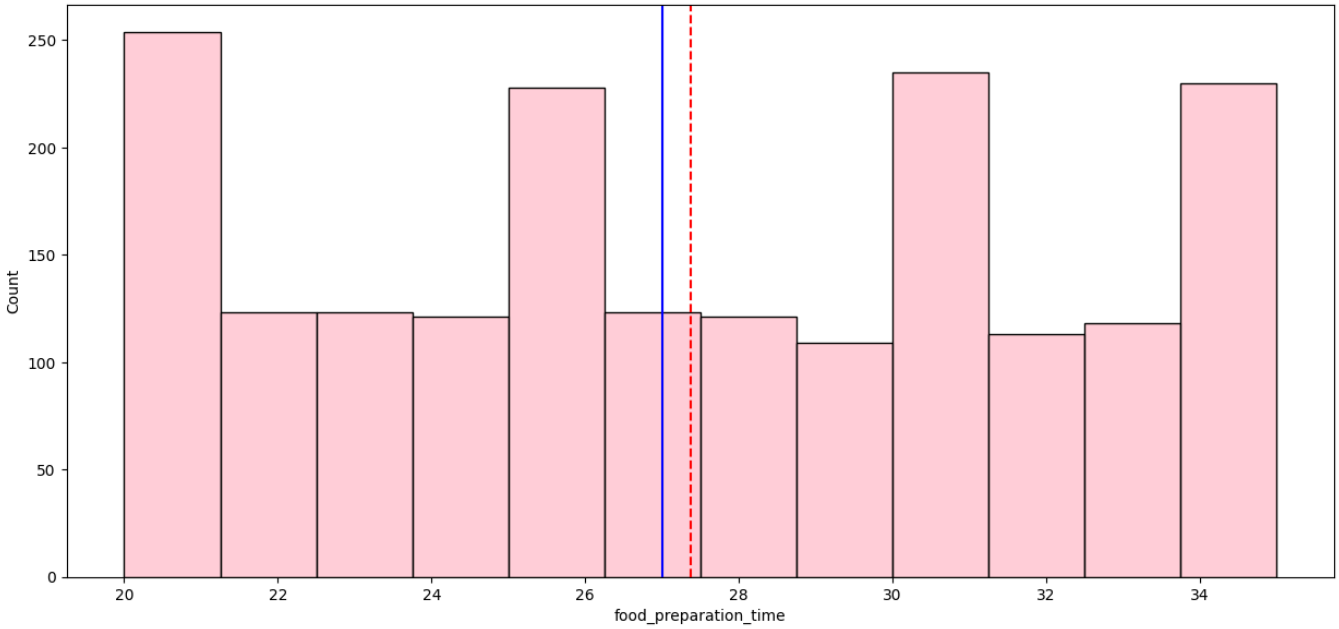
Observations:

- The data is left-skewed due to the higher number of unrated restaurants
- Most of the actual reviews ranged from 3-5 stars

Observations on food preparation time

```
histo_boxplot(df.food_preparation_time)
```


20
22
24
26
28
30
32
34
food_preparation_time

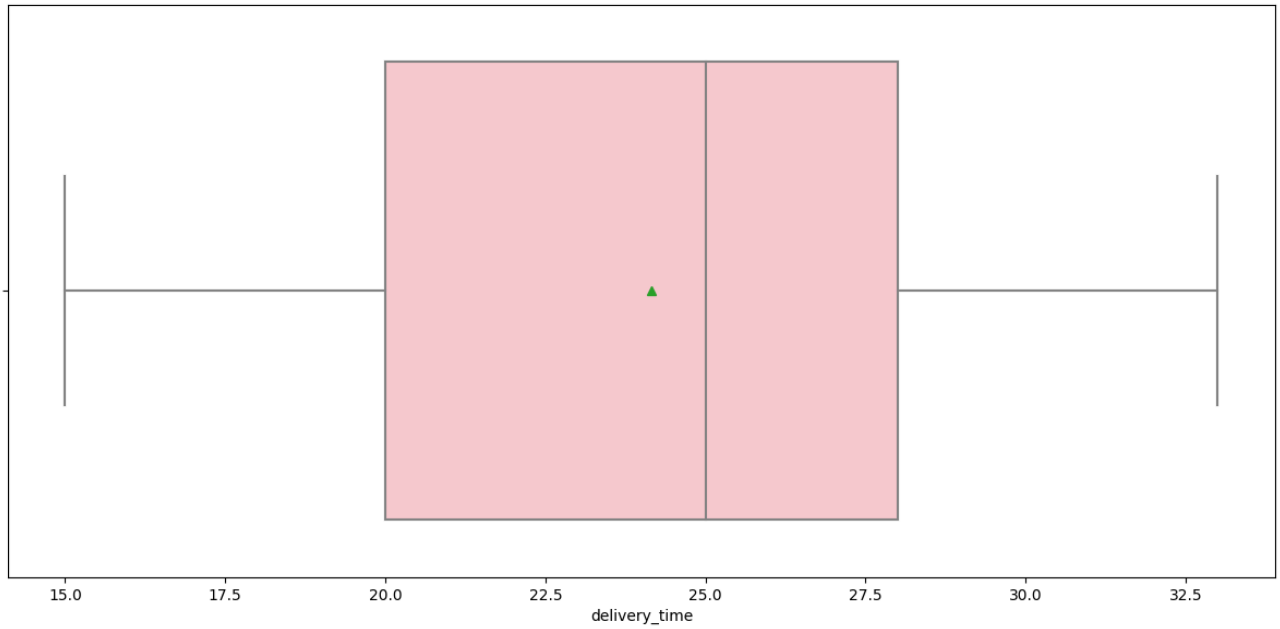

250
200
150
100
50
0
Count
20
22
24
26
28
30
32
34
food_preparation_time

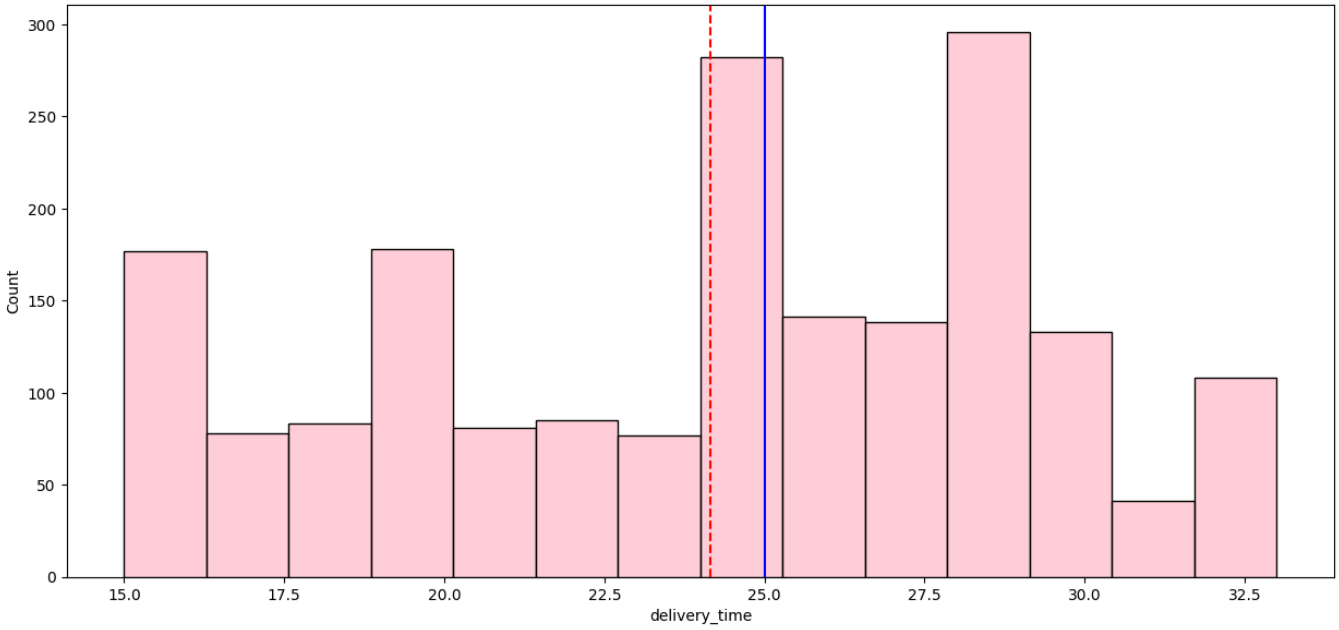
Observations:

- The distribution is slightly right-skewed as the mean > median, but looks symmetric.
- The food preparation will generally take at least 20 minutes and no more than 36 minutes.
- Less people will probably order if it goes over 36 minutes to prepare the food.

Observations on delivery time

```
histo_boxplot(df.delivery_time)
```


15.0
17.5
20.0
22.5
25.0
27.5
30.0
32.5
delivery_time

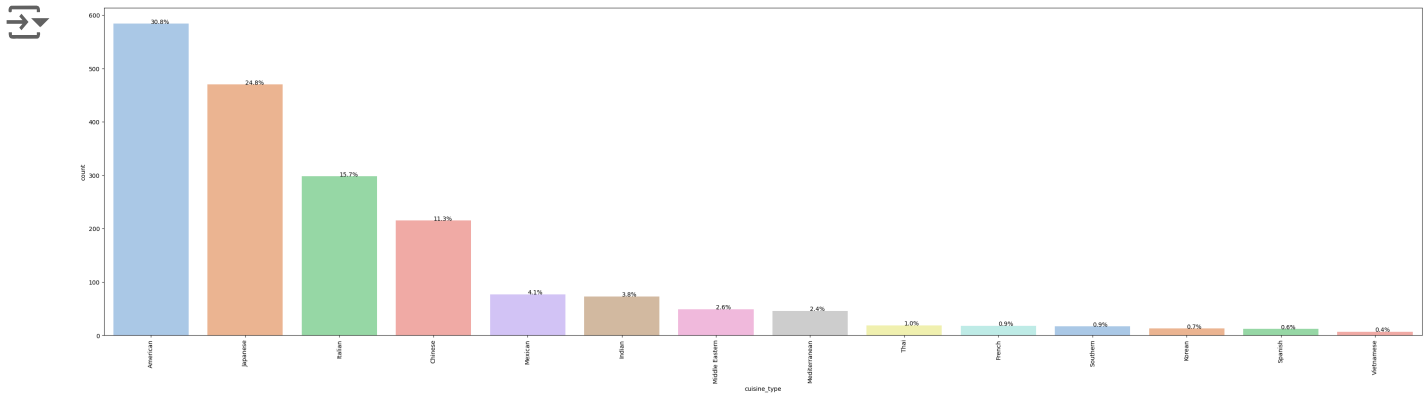

300
250
200
150
100
50
0
Count
15.0
17.5
20.0
22.5
25.0
27.5
30.0
32.5
delivery_time

Observations:

- The distribution is left-skewed based on the boxplot and histogram.
- Delivery times tend to take more than 20 minutes, and will vary depending on the restaurant.
- The delivery time can be affected depending on the cost of the food

Observations on cuisine type

```
bar_percent(df, 'cuisine_type')
```

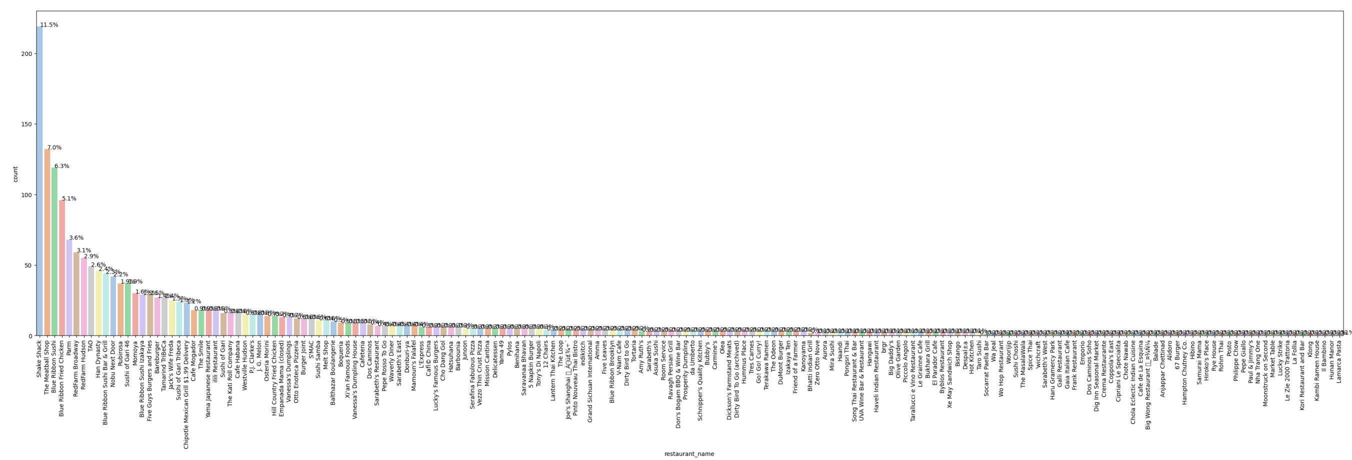


Observations:

- The main cuisines were American, Japanese, Italian, and Chinese
- There were a few outliers that didn't have as much movement as the others.
- Most of the orders were to one of four cuisines.

Observations on restaurant name

```
bar_percent(df, 'restaurant_name')
```

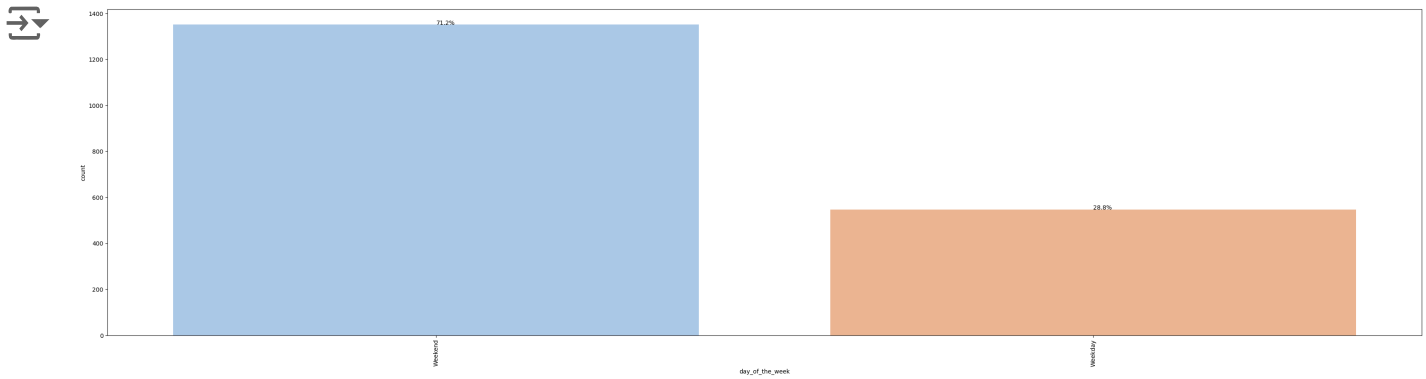


Observations:

- Shake Shack received the most orders
- There was a large variety of restaurants that were included in the data
- All values will have at least 1 order

Observations on day of the week

```
bar_percent(df, 'day_of_the_week')
```



Observations:

- Most of the orders were done on the weekend
- Although not as many orders were done on weekdays, it still provides a considerable amount of data.

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

```
df['restaurant_name'].value_counts().nlargest(5)
```

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

Observations:

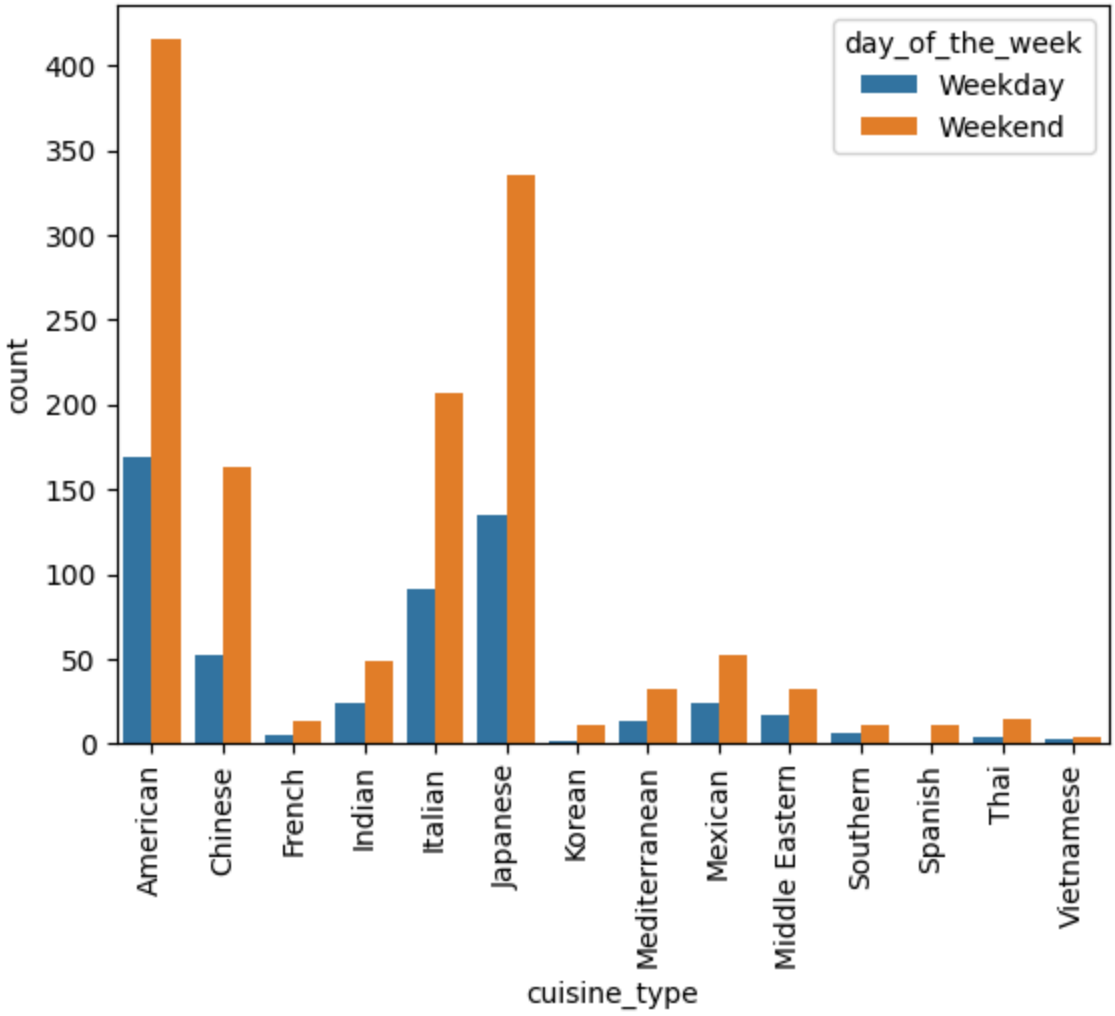
- The top 5 restaurants in terms of number of orders received are:

1. Shake Shack
2. The Meatball Shop
3. Blue Ribbon Sushi
4. Blue Ribbon Fried Chicken
5. Parm

Question 8: Which is the most popular cuisine on weekends?

```
sns.countplot(data=df, x='cuisine_type',hue='day_of_the_week')
plt.xticks(rotation=90)
```

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13]),
 [Text(0, 0, 'American'),
  Text(1, 0, 'Chinese'),
  Text(2, 0, 'French'),
  Text(3, 0, 'Indian'),
  Text(4, 0, 'Italian'),
  Text(5, 0, 'Japanese'),
  Text(6, 0, 'Korean'),
  Text(7, 0, 'Mediterranean'),
  Text(8, 0, 'Mexican'),
  Text(9, 0, 'Middle Eastern'),
  Text(10, 0, 'Southern'),
  Text(11, 0, 'Spanish'),
  Text(12, 0, 'Thai'),
  Text(13, 0, 'Vietnamese')])
```



Observations:

- The most popular cuisine on weekends is American.

Question 9: What percentage of the orders cost more than 20 dollars?

```
over20 = df['cost_of_the_order'] > 20
over20.value_counts(normalize = True)
```

```
False    0.707587
True     0.292413
Name: cost_of_the_order, dtype: float64
```

Observations:

- 29.24% of the orders cost more than 20 dollars.

Question 10: What is the mean order delivery time?

```
df['delivery_time'].mean()
```

```
24.161749209694417
```

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 3 most frequent customers. Find the IDs of these customers and the number of orders they placed

```
df['customer_id'].value_counts().nlargest(3)
```

```
52832    13
47440    10
83287     9
Name: customer_id, dtype: int64
```

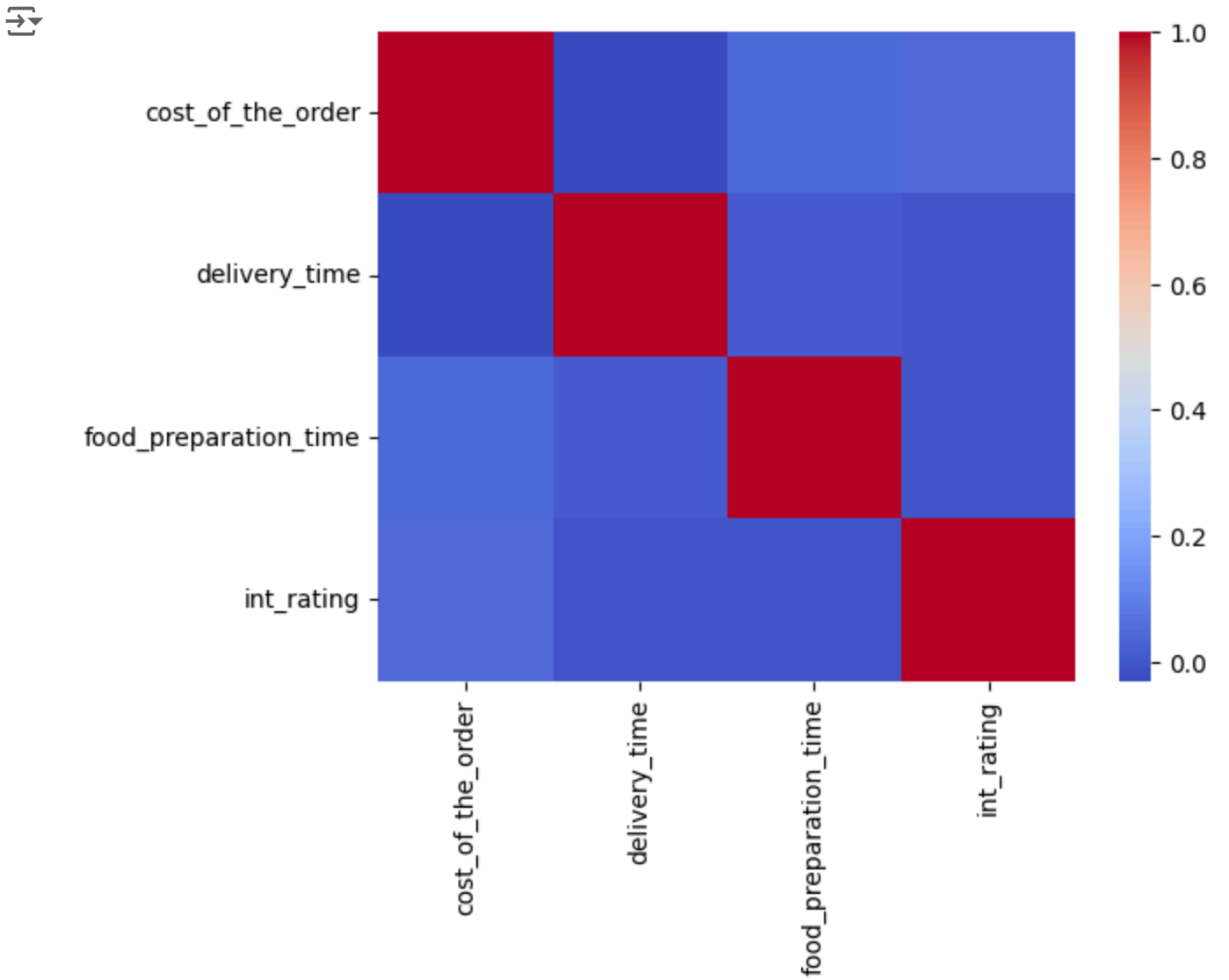
Observations:

- The top 3 most frequent customer's Ids are
    1. 52832
    2. 47440
    3. 83287

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)

```
sns.heatmap(data=df[['cost_of_the_order', 'delivery_time', 'food_preparation_time',
```



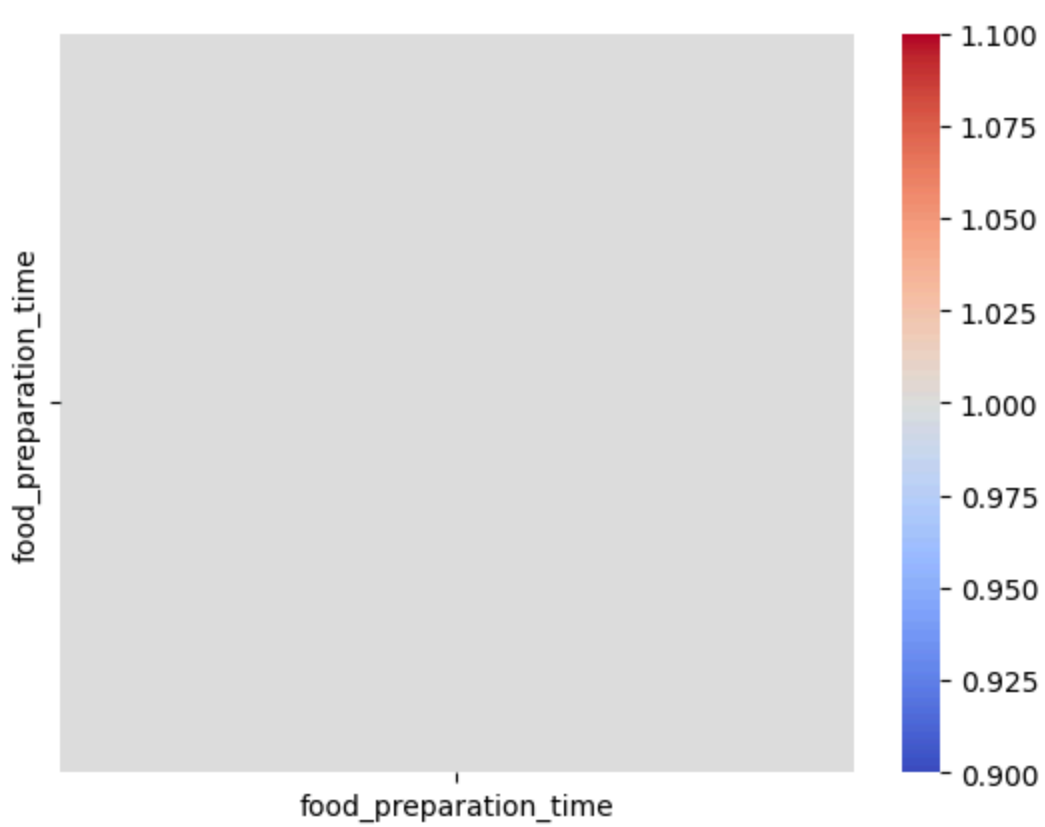
Observations:

- We can see that there is no correlation between the delivery time and the cost of the order

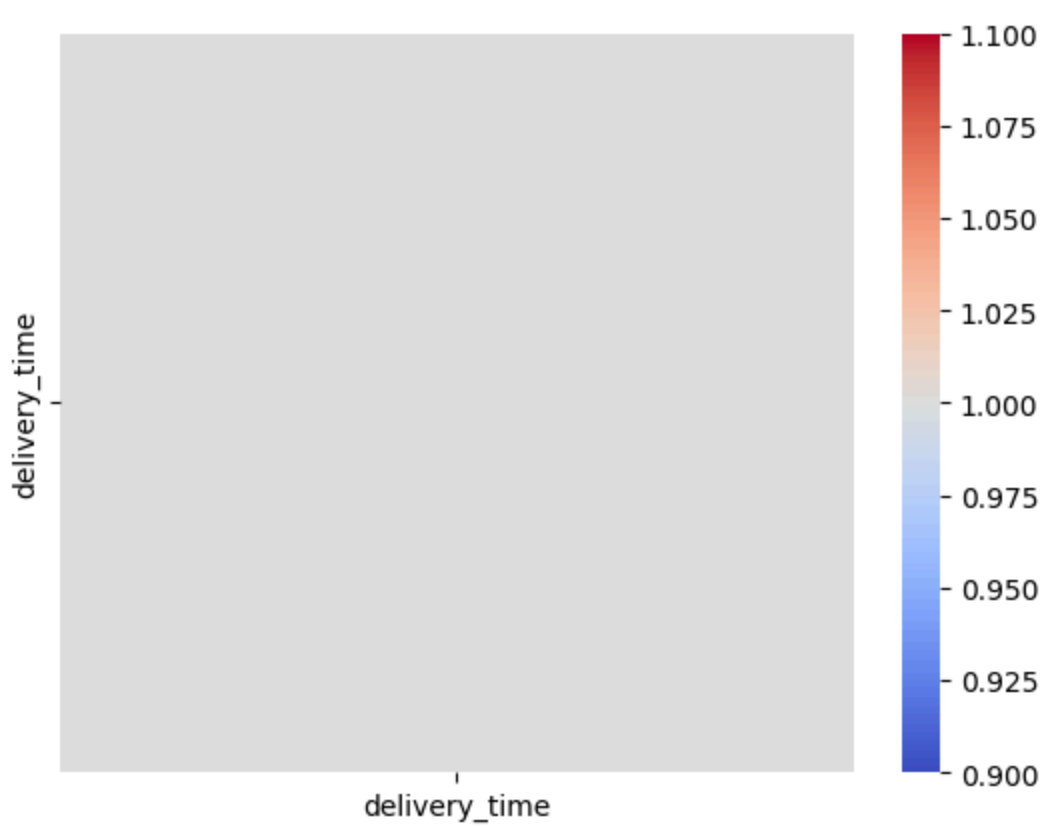
- There is very little correlation food preparation time and the cost of the order, meaning the price doesn't always affect the time it takes to prepare since it could mean a lot of things like more expensive ingredients, faster cook time, and not necessarily larger orders.
- The rating and the cost of the order did have a small correlation

```
sns.heatmap(data=df[['cost_of_the_order', 'day_of_the_week']].corr(), cmap = 'coolwa
```

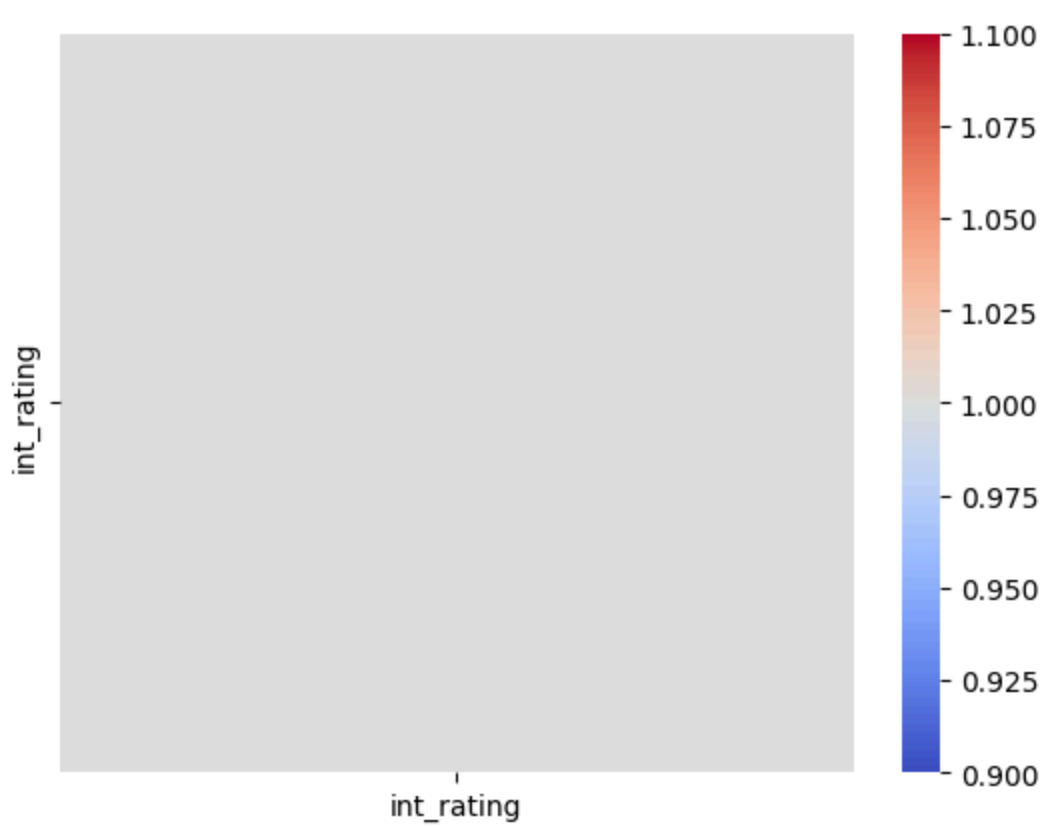

```
sns.heatmap(data=df[['food_preparation_time', 'day_of_the_week']].corr(), cmap = 'co
```




```
sns.heatmap(data=df[['delivery_time', 'day_of_the_week']].corr(), cmap = 'coolwarm')
```




```
sns.heatmap(data=df[['int_rating', 'day_of_the_week']].corr(), cmap = 'coolwarm');
```

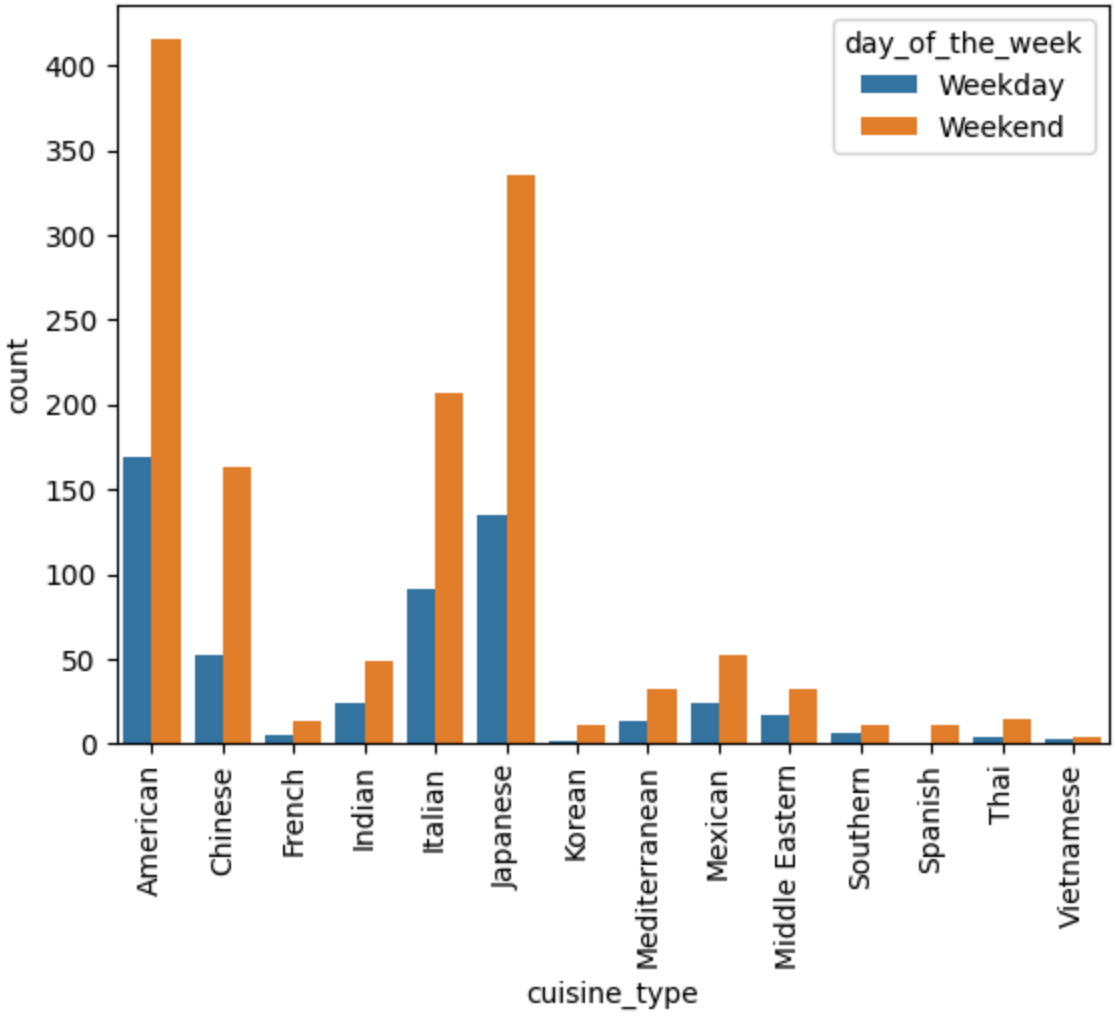


Observations:

- As seen in the heatmaps, there are no correlations between the day of the week and the numeric variables.

```
sns.countplot(data = df, x = 'cuisine_type', hue = 'day_of_the_week')
plt.xticks(rotation=90)
```

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13]),
 [Text(0, 0, 'American'),
  Text(1, 0, 'Chinese'),
  Text(2, 0, 'French'),
  Text(3, 0, 'Indian'),
  Text(4, 0, 'Italian'),
  Text(5, 0, 'Japanese'),
  Text(6, 0, 'Korean'),
  Text(7, 0, 'Mediterranean'),
  Text(8, 0, 'Mexican'),
  Text(9, 0, 'Middle Eastern'),
  Text(10, 0, 'Southern'),
  Text(11, 0, 'Spanish'),
  Text(12, 0, 'Thai'),
  Text(13, 0, 'Vietnamese')])
```



Observations

- We can see that more restaurants will receive more customers on weekends
- Restaurants will do better on weekends than compared to weekdays