Project Python Foundations: 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_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

Let us start by importing the required libraries

In [1]:
```
# Installing the libraries with the specified version.
#!pip install numpy==1.25.2 pandas==1.5.3 matplotlib==3.7.1 seaborn==0.13.1 -q --user
```

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

In [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
```

Understanding the structure of the data

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

In [4]:
```
# Read the data from the CSV file
data = pd.read_csv('foodhub_order.csv')

# Display the first few rows of the dataframe to verify
data.head()
```

Out[4]:

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

In [5]:
```
# View the first 5 rows of the dataframe
data.head()
```

Out[5]:

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

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

In [6]:
```
# Get the number of rows and columns in the data
num_rows, num_columns = data.shape

# Print the result
print(f'The data has {num_rows} rows and {num_columns} columns.')
```

```
The data has 1898 rows and 9 columns.
```

Observations:

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

In [7]:
```
# Check the datatypes of the different columns in the dataset
data.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:

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

In [8]:
```
# Check for missing values in the data
missing_values = data.isnull().sum()

# Print the result
print("Missing values in each column:")
print(missing_values)
```

```
Missing values in each column:
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:

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 [9]:
```
# Calculate the minimum, average, maximum, median, and quartile food preparation time
min_prep_time = data['food_preparation_time'].min()
avg_prep_time = data['food_preparation_time'].mean()
max_prep_time = data['food_preparation_time'].max()
median_prep_time = data['food_preparation_time'].median()
q1_prep_time = data['food_preparation_time'].quantile(0.25)
q3_prep_time = data['food_preparation_time'].quantile(0.75)

# Print the results
print(f'Minimum food preparation time: {min_prep_time} minutes')
print(f'Average food preparation time: {avg_prep_time:.2f} minutes')
print(f'Maximum food preparation time: {max_prep_time} minutes')
print(f'Median food preparation time: {median_prep_time} minutes')
print(f'1st Quartile (Q1) food preparation time: {q1_prep_time} minutes')
print(f'3rd Quartile (Q3) food preparation time: {q3_prep_time} minutes')
```

```
Minimum food preparation time: 20 minutes
Average food preparation time: 27.37 minutes
Maximum food preparation time: 35 minutes
Median food preparation time: 27.0 minutes
1st Quartile (Q1) food preparation time: 23.0 minutes
3rd Quartile (Q3) food preparation time: 31.0 minutes
```

Observations:

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

In [10]:
```
# Find the number of orders that are not rated
not_rated_orders = data[data['rating'] == 'Not given'].shape[0]

# Print the result
print(f'The number of orders that are not rated: {not_rated_orders}')
```

```
The number of orders that are not rated: 736
```

Observations:

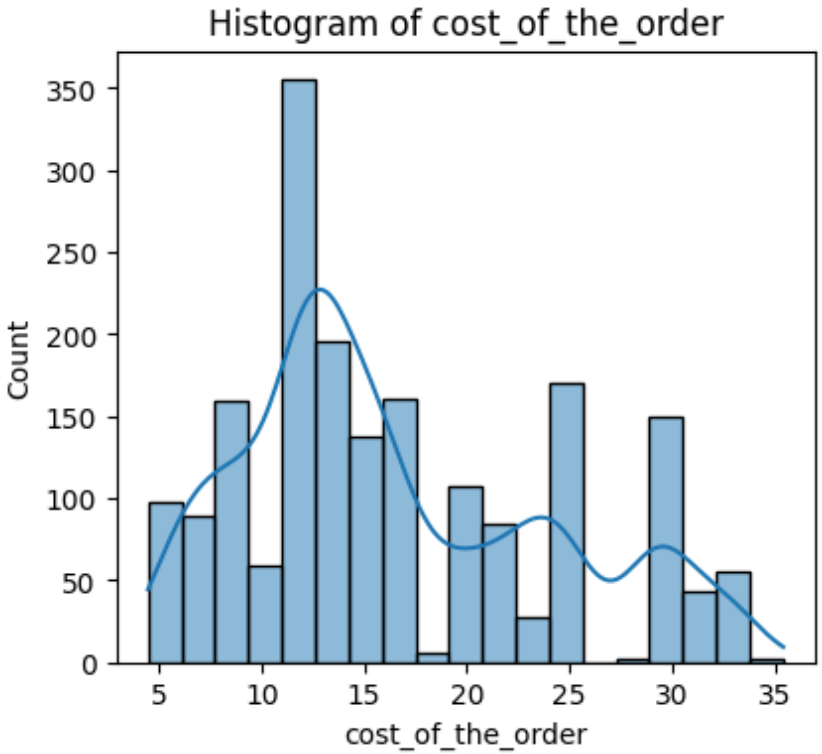
Exploratory Data Analysis (EDA)

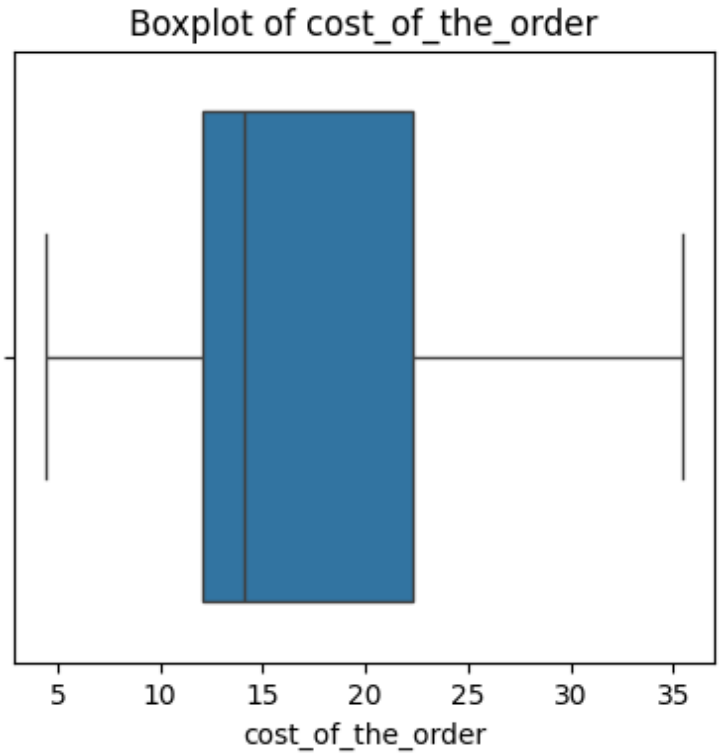
Univariate Analysis

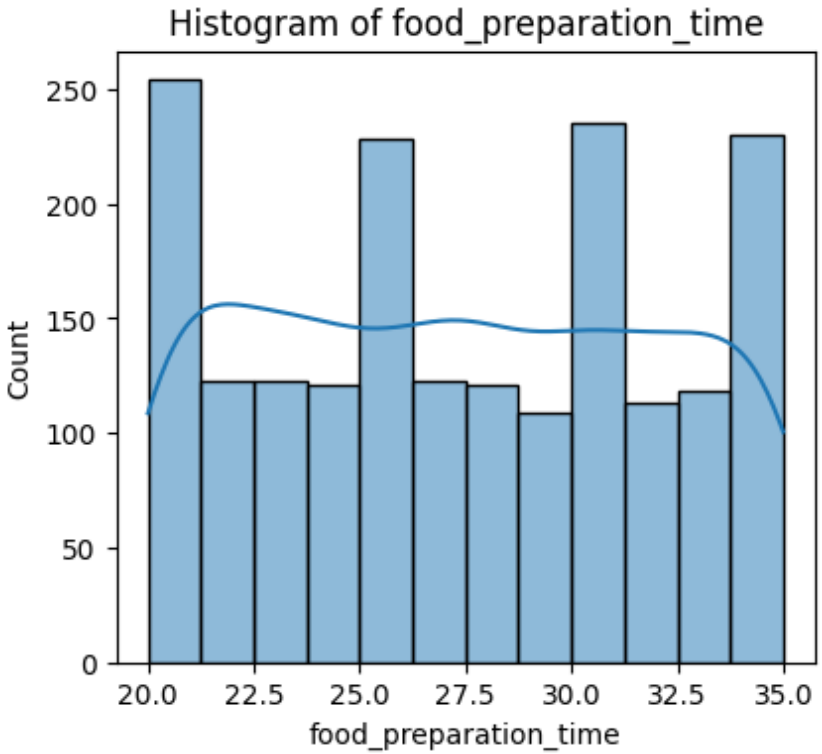
Question 6: Explore all the variables and provide observations on their distributions. (Generally, histograms, boxplots, countplots, etc. are used for univariate exploration.) [9 marks]

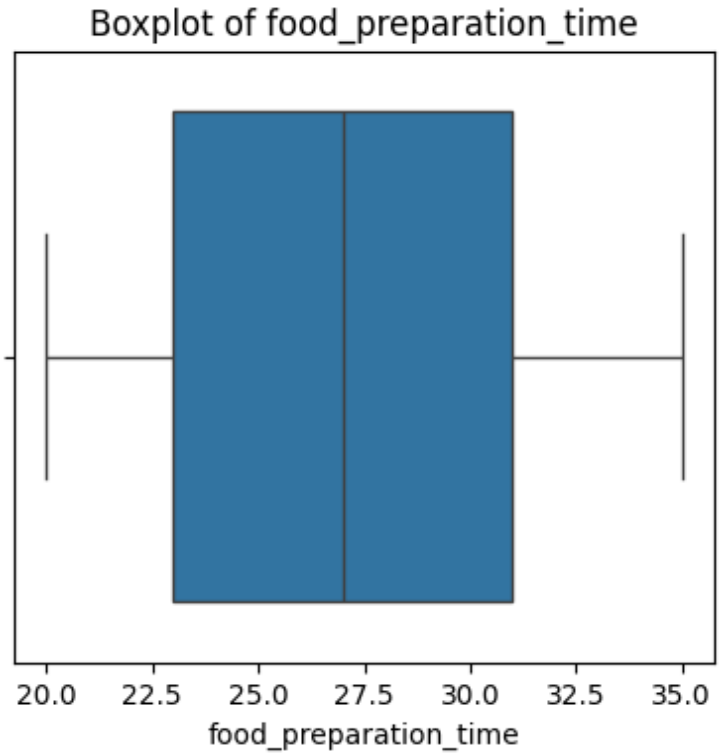
In [11]:
```
# Univariate analysis of numerical variables
numerical_columns = ['cost_of_the_order', 'food_preparation_time', 'delivery_time']

for col in numerical_columns:
        plt.figure(figsize=(10, 4))
```

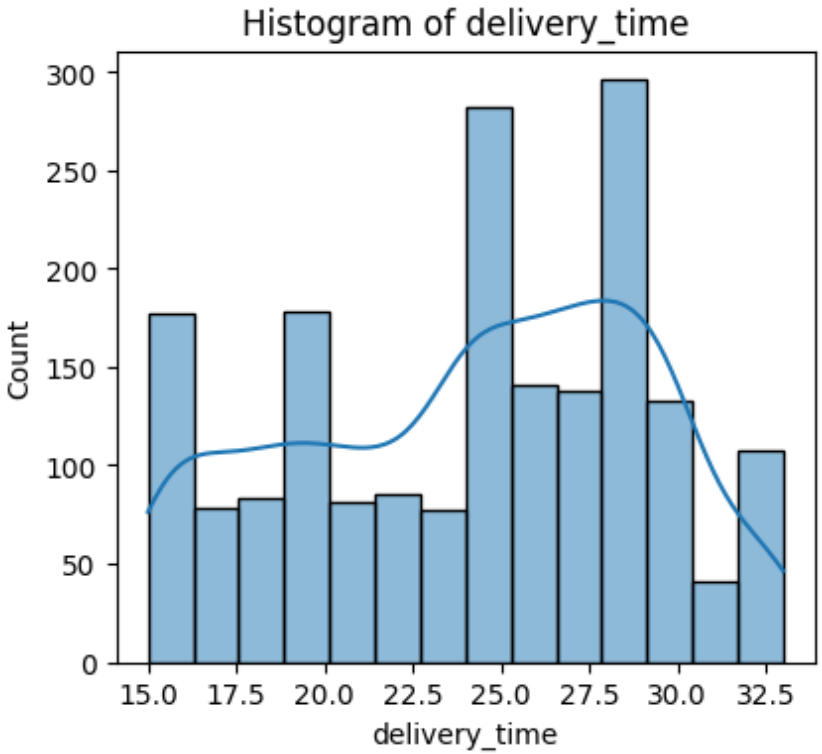
```
        plt.subplot(1, 2, 1)
        sns.histplot(data[col], kde=True)
        plt.title(f'Histogram of {col}')

        plt.subplot(1, 2, 2)
        sns.boxplot(x=data[col])
        plt.title(f'Boxplot of {col}')

        plt.show()

# Univariate analysis of categorical variables
categorical_columns = ['cuisine_type', 'day_of_the_week', 'rating']

for col in categorical_columns:
        plt.figure(figsize=(10, 4))
        sns.countplot(y=data[col], order=data[col].value_counts().index)
        plt.title(f'Countplot of {col}')
        plt.show()
```

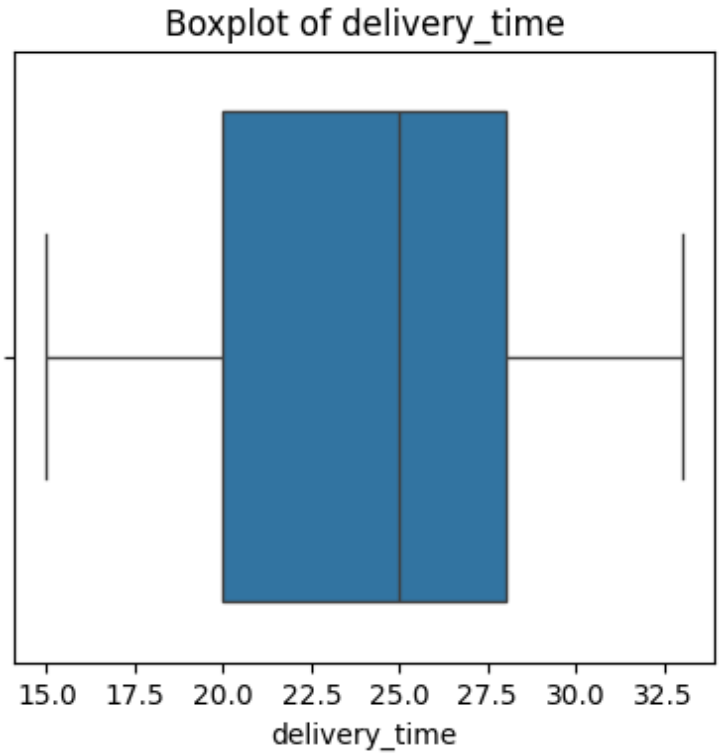

Histogram of cost_of_the_order
Count
cost_of_the_order
0
50
100
150
200
250
300
350
5
10
15
20
25
30
35

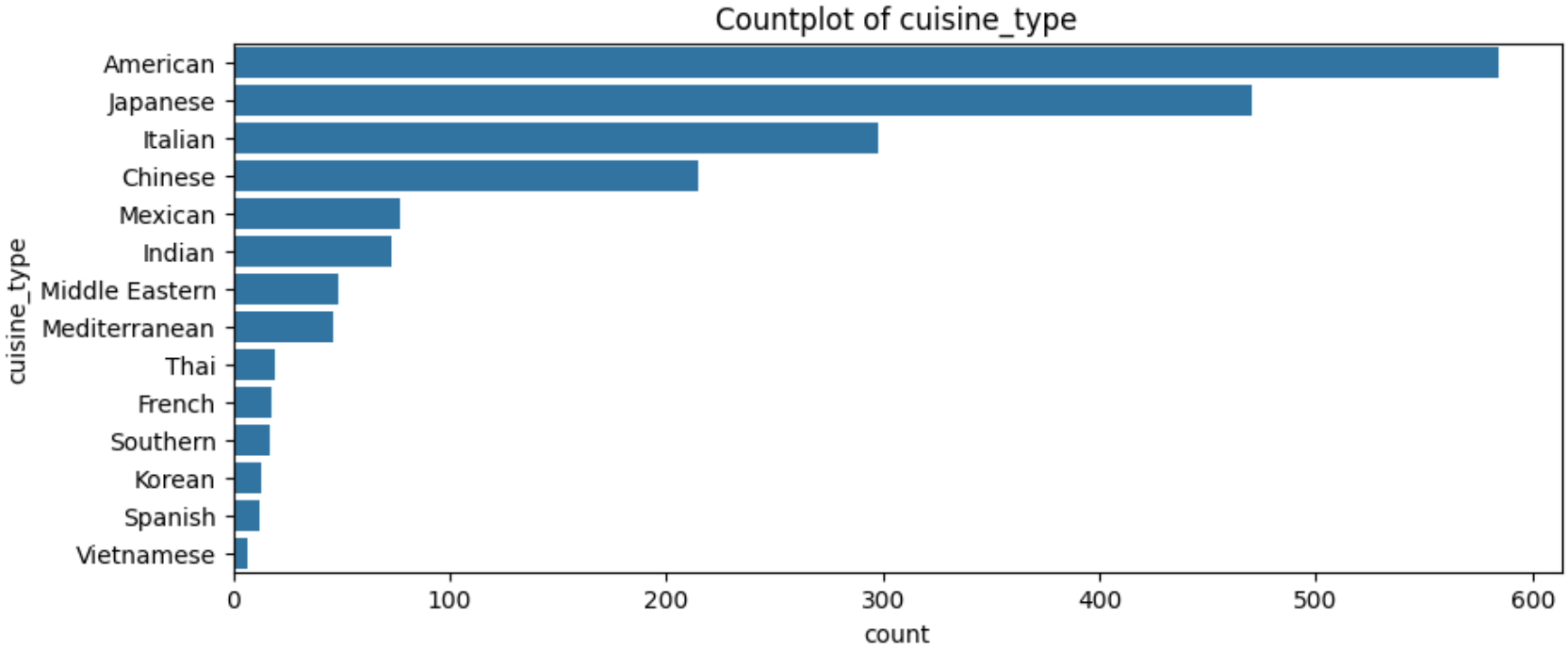

Boxplot of cost_of_the_order
cost_of_the_order
5
10
15
20
25
30
35

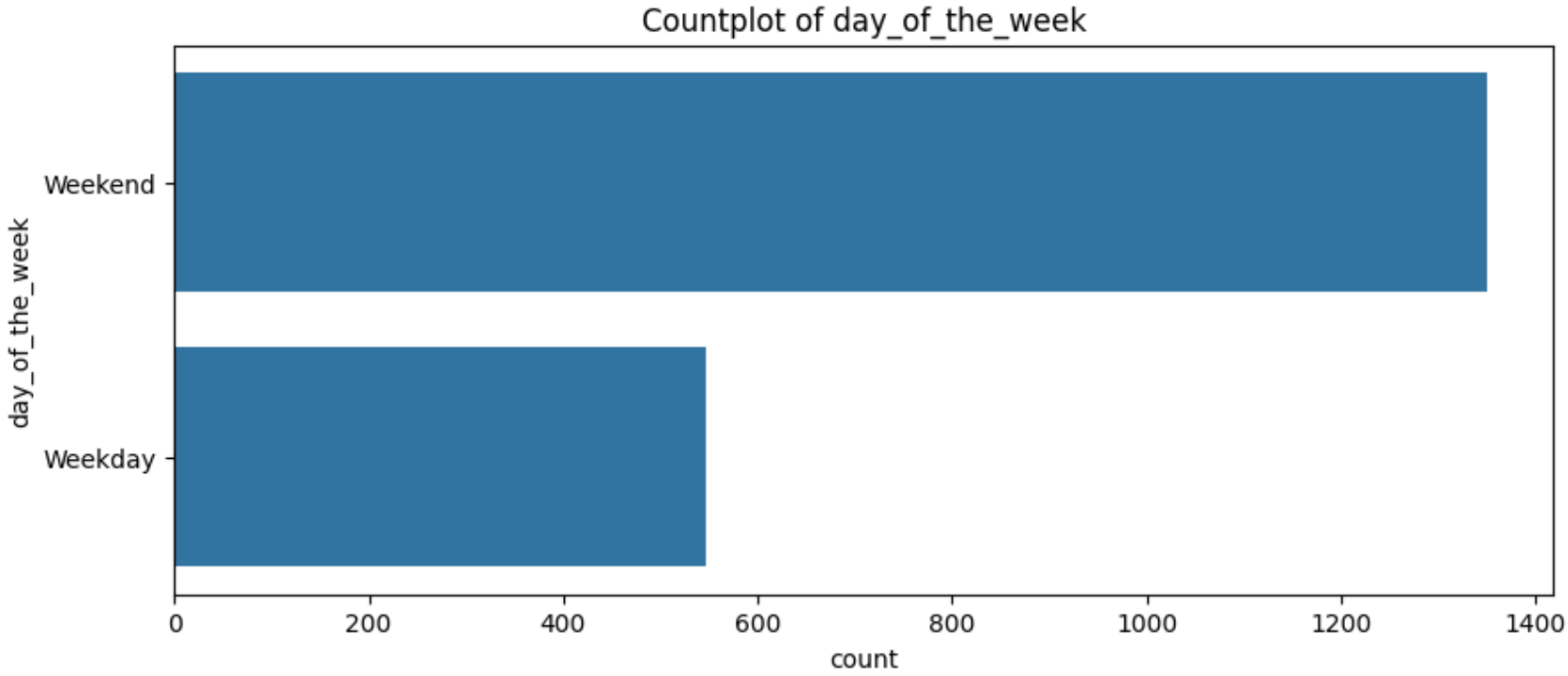

Histogram of food_preparation_time
Count
250
200
150
100
50
0
20.0
22.5
25.0
27.5
30.0
32.5
35.0
food_preparation_time

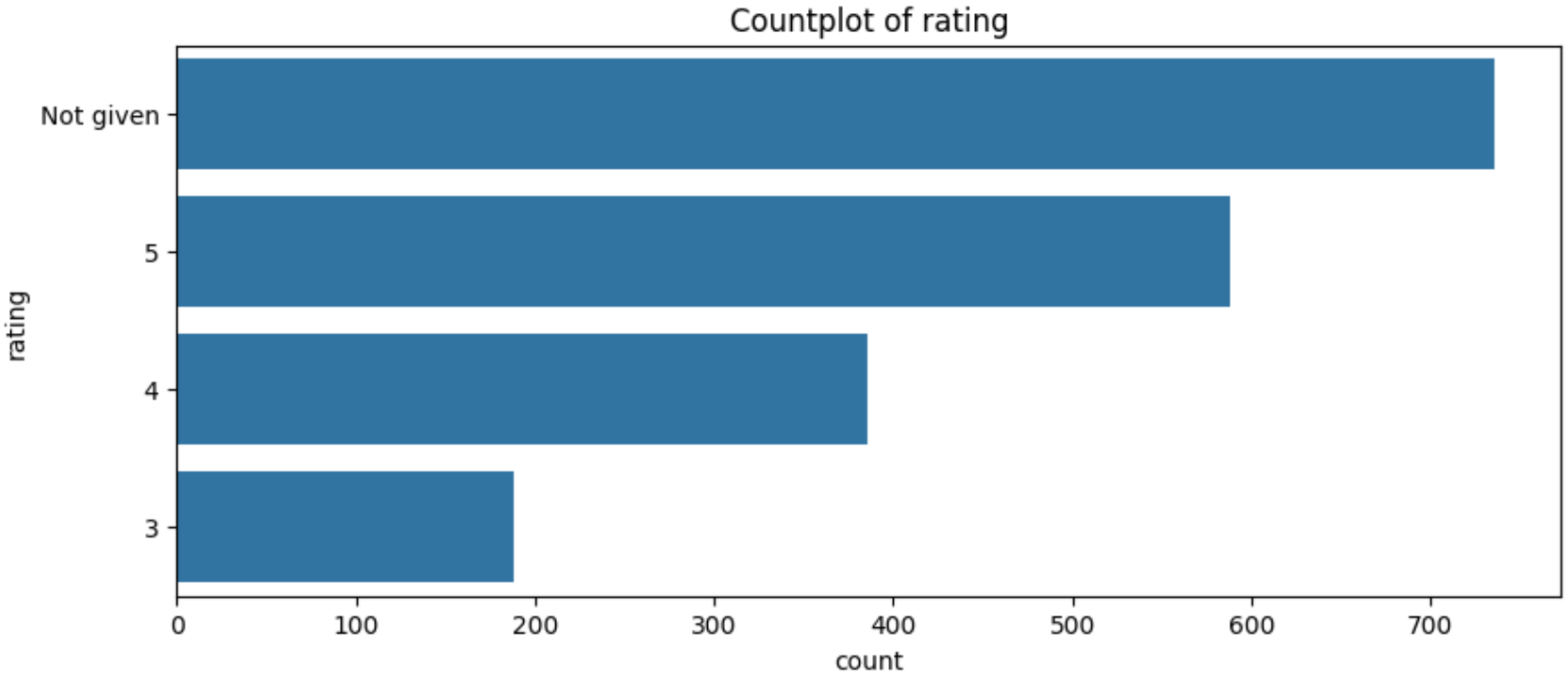

Boxplot of food_preparation_time
20.0
22.5
25.0
27.5
30.0
32.5
35.0
food_preparation_time


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


Boxplot of delivery_time
delivery_time
15.0
17.5
20.0
22.5
25.0
27.5
30.0
32.5


Countplot of cuisine_type
American
Japanese
Italian
Chinese
Mexican
Indian
Middle Eastern
Mediterranean
Thai
French
Southern
Korean
Spanish
Vietnamese
cuisine_type
0
100
200
300
400
500
600
count


Countplot of day_of_the_week
day_of_the_week
Weekend
Weekday
0
200
400
600
800
1000
1200
1400
count



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

In [12]:
```
# Find the top 5 restaurants in terms of the number of orders received
top_5_restaurants = data['restaurant_name'].value_counts().head(5)

# Print the result
print("Top 5 restaurants in terms of the number of orders received:")
print(top_5_restaurants)
```

```
Top 5 restaurants in terms of the number of orders received:
restaurant_name
Shake Shack                  219
The Meatball Shop            132
Blue Ribbon Sushi            119
Blue Ribbon Fried Chicken     96
Parm                          68
Name: count, dtype: int64
```

Observations:

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

In [13]:
```
# Filter the data for weekends
weekend_data = data[data['day_of_the_week'] == 'Weekend']

# Find the most popular cuisine on weekends
most_popular_cuisine_weekend = weekend_data['cuisine_type'].value_counts().idxmax()

# Calculate the percentage of orders for the most popular cuisine on weekends
most_popular_cuisine_count = weekend_data['cuisine_type'].value_counts().max()
total_weekend_orders = weekend_data.shape[0]
percentage_most_popular_cuisine = (most_popular_cuisine_count / total_weekend_orders) * 100

# Print the result
print(f'The most popular cuisine on weekends is: {most_popular_cuisine_weekend}')
print(f'Percentage of orders for the {most_popular_cuisine_weekend} cusine on weekends: {percentage
```

```
The most popular cuisine on weekends is: American
Percentage of orders for the American cusine on weekends: 30.72%
```

Observations:

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

In [14]:
```
# Calculate the number of orders that cost more than 20 dollars
orders_cost_more_than_20 = data[data['cost_of_the_order'] > 20].shape[0]
```

```
# Calculate the percentage of such orders
percentage_orders_cost_more_than_20 = (orders_cost_more_than_20 / num_rows) * 100

# Print the result
print(f'The number of orders that cost more than 20 dollars: {orders_cost_more_than_20}')
print(f'The percentage of orders that cost more than 20 dollars: {percentage_orders_cost_more_than_
```

```
The number of orders that cost more than 20 dollars: 555
The percentage of orders that cost more than 20 dollars: 29.24%
```

Observations:

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

In [15]:
```
# Calculate the mean order delivery time
mean_delivery_time = data['delivery_time'].mean()

# Print the result
print(f'The mean order delivery time is: {mean_delivery_time:.2f} minutes')
```

```
The mean order delivery time is: 24.16 minutes
```

Observations:

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 [16]:
```
# Find the top 3 most frequent customers
top_3_customers = data['customer_id'].value_counts().head(3)

# Print the result
print("Top 3 most frequent customers and the number of orders they placed:")
print(top_3_customers)
```

```
Top 3 most frequent customers and the number of orders they placed:
customer_id
52832    13
47440    10
83287     9
Name: count, dtype: int64
```

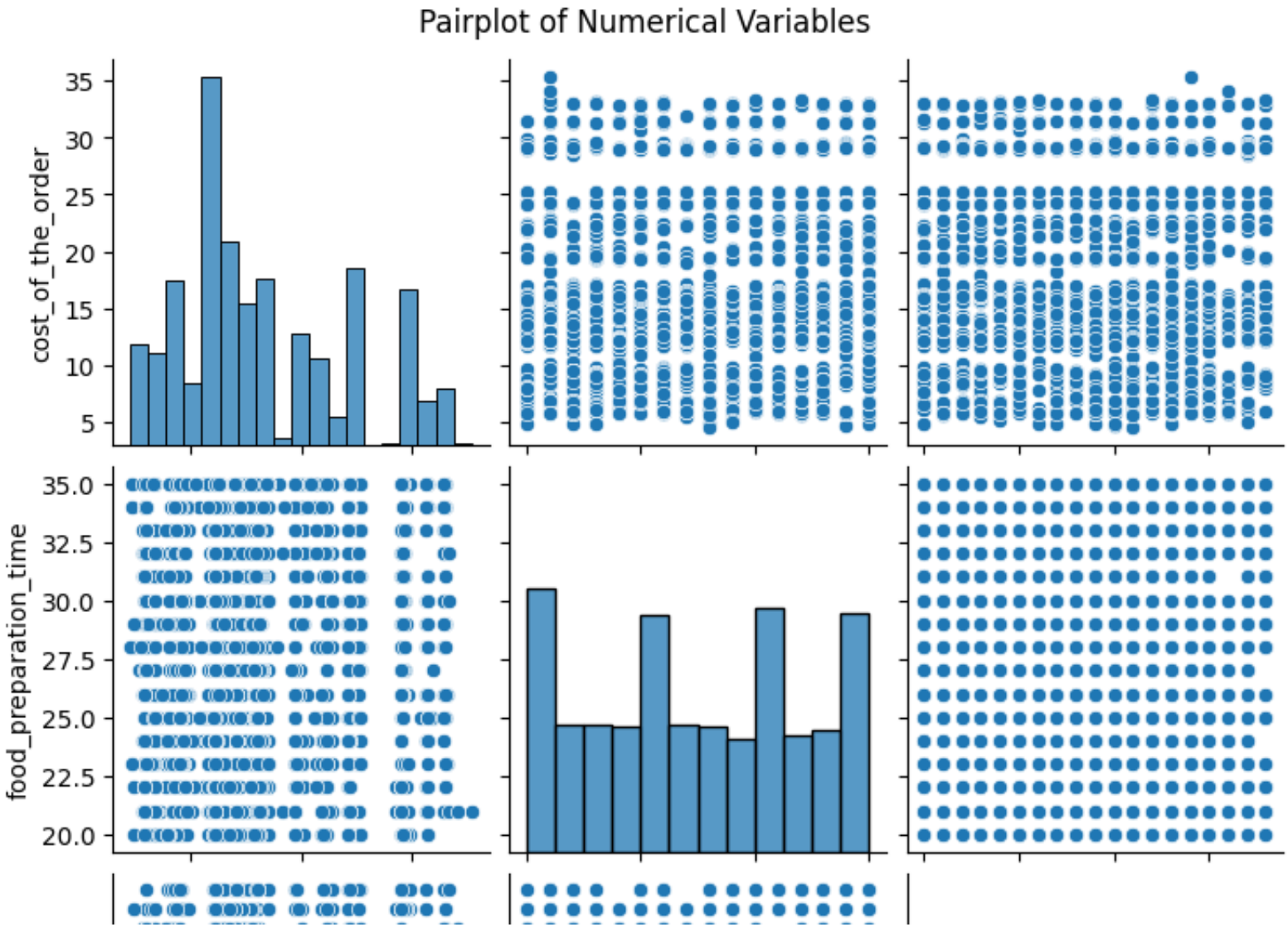
Observations:

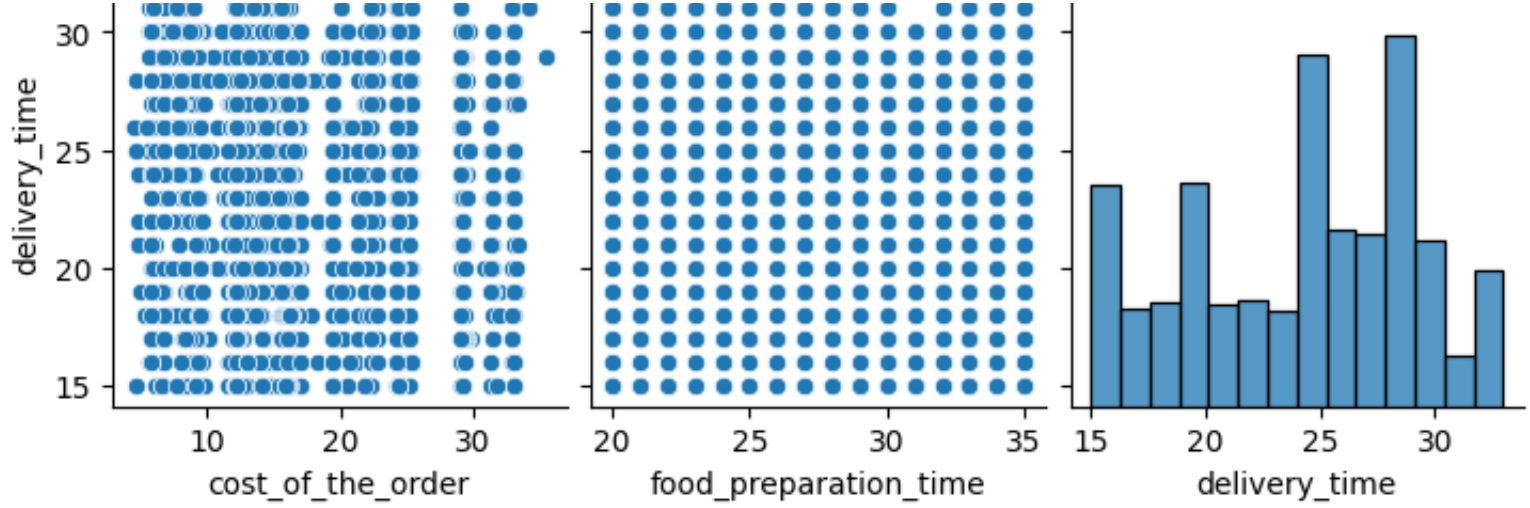
Multivariate Analysis

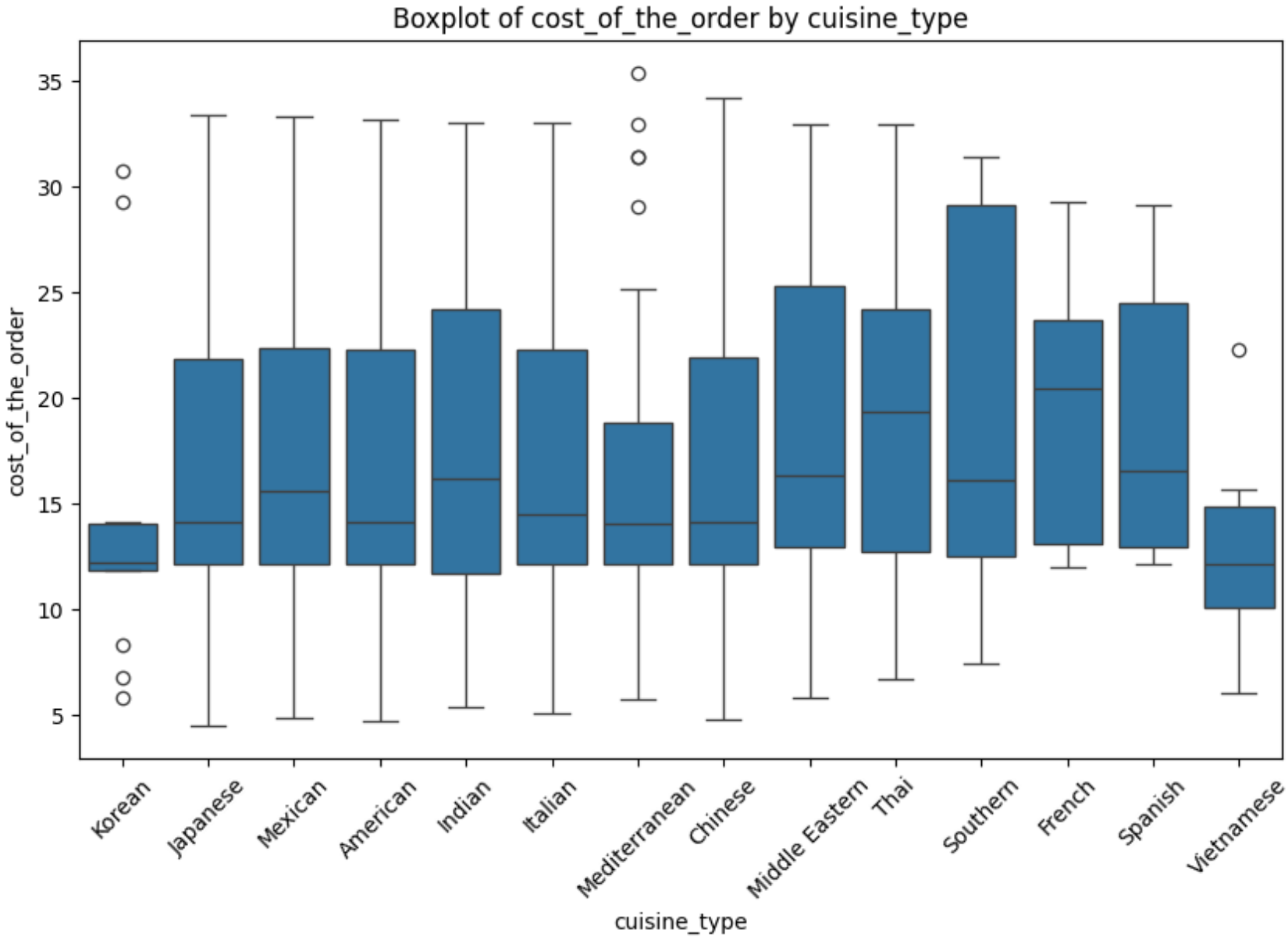
Question 12: Perform a multivariate analysis to explore relationships between the important variables in the dataset. (It is a good idea to explore relations between numerical variables as well as relations between numerical and categorical variables) [10 marks]

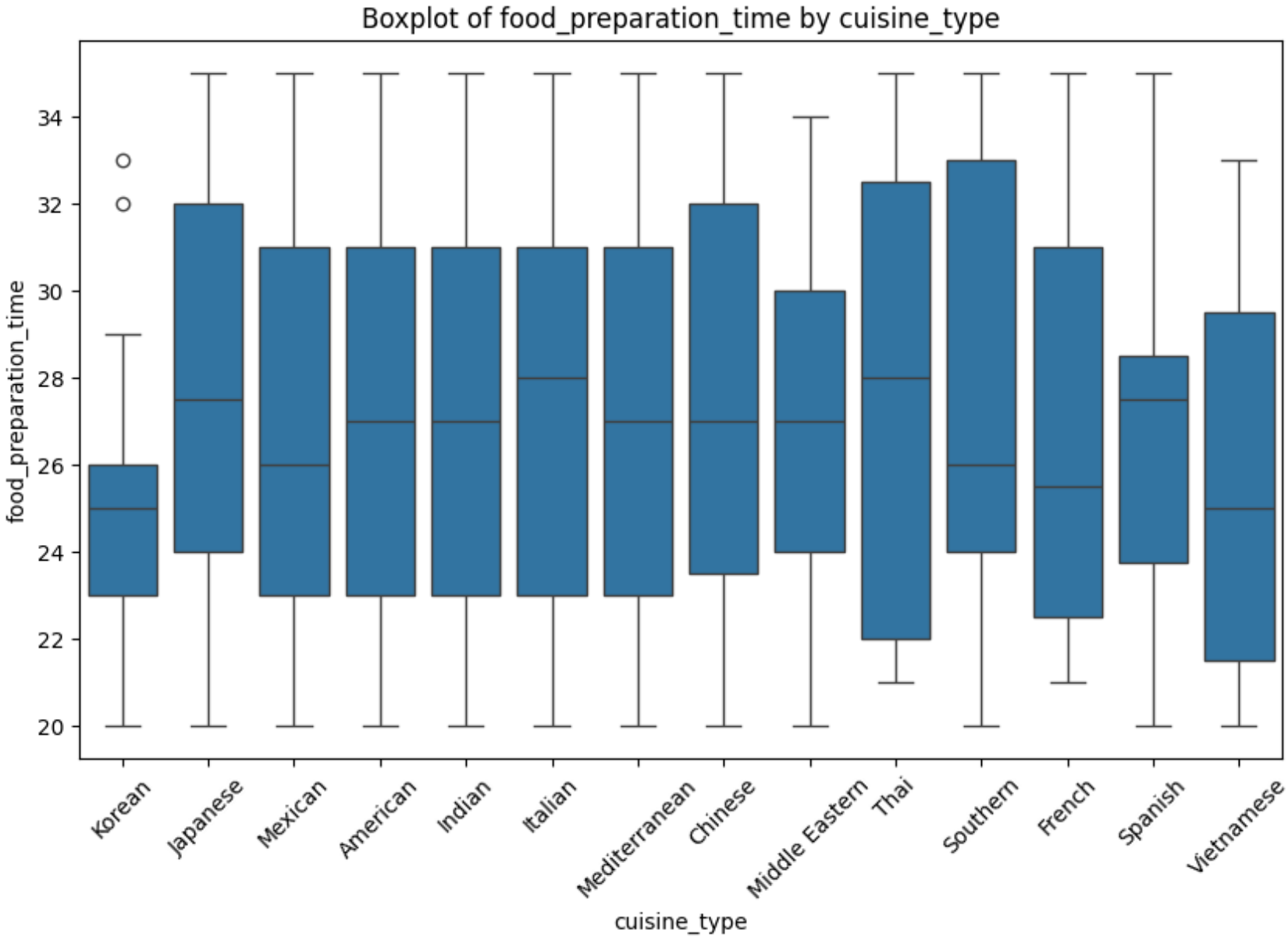
In [17]:
```
# Scatter plot to explore relationships between numerical variables
sns.pairplot(data[numerical_columns])
plt.suptitle('Pairplot of Numerical Variables', y=1.02)
plt.show()
# Calculate the total time required to deliver the food (food preparation time + delivery time)
data['total_delivery_time'] = data['food_preparation_time'] + data['delivery_time']

numerical_columns.append("total_delivery_time")



# Box plots to explore relationships between numerical and categorical variables
for cat_col in categorical_columns:
        for num_col in numerical_columns:
                plt.figure(figsize=(10, 6))
                sns.boxplot(x=cat_col, y=num_col, data=data)
                plt.title(f'Boxplot of {num_col} by {cat_col}')
                plt.xticks(rotation=45)
                plt.show()
```

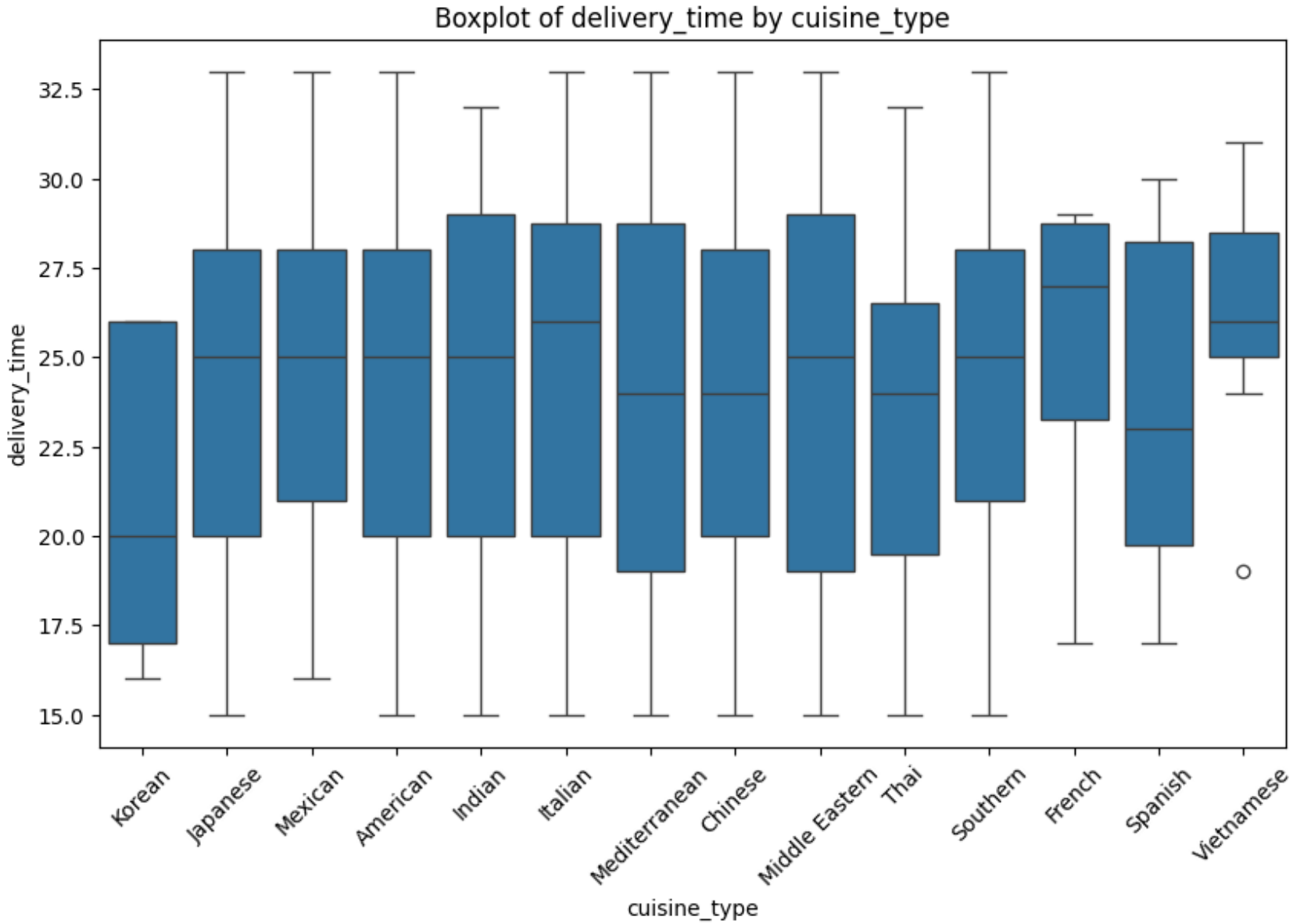
```
# Point plots to explore relationships between numerical and categorical variables
for cat_col in categorical_columns:
        for num_col in numerical_columns:
                plt.figure(figsize=(10, 6))
                sns.pointplot(x=cat_col, y=num_col, data=data)
                plt.title(f'Pointplot of {num_col} by {cat_col}')
                plt.xticks(rotation=45)
                plt.show()
```

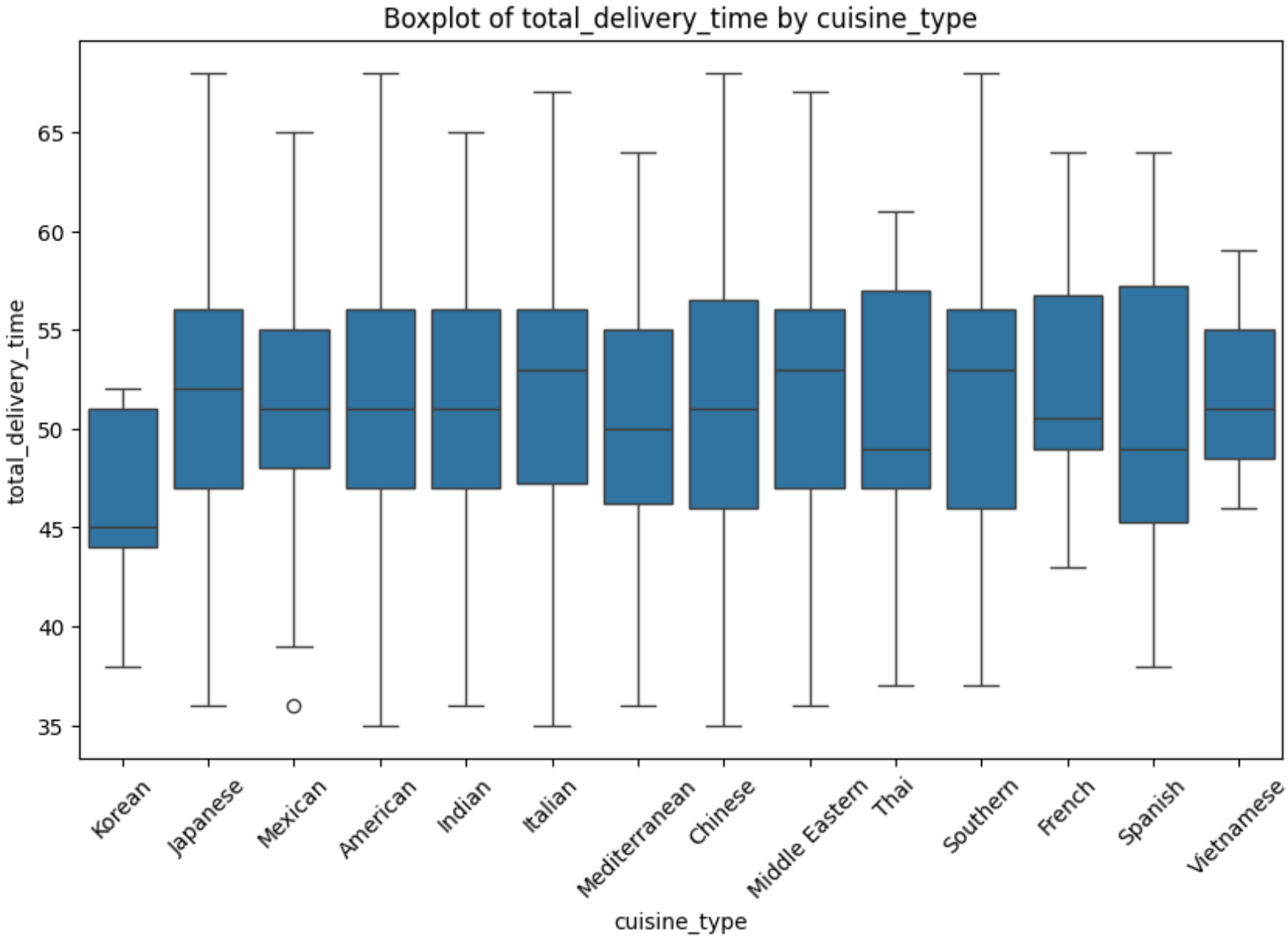


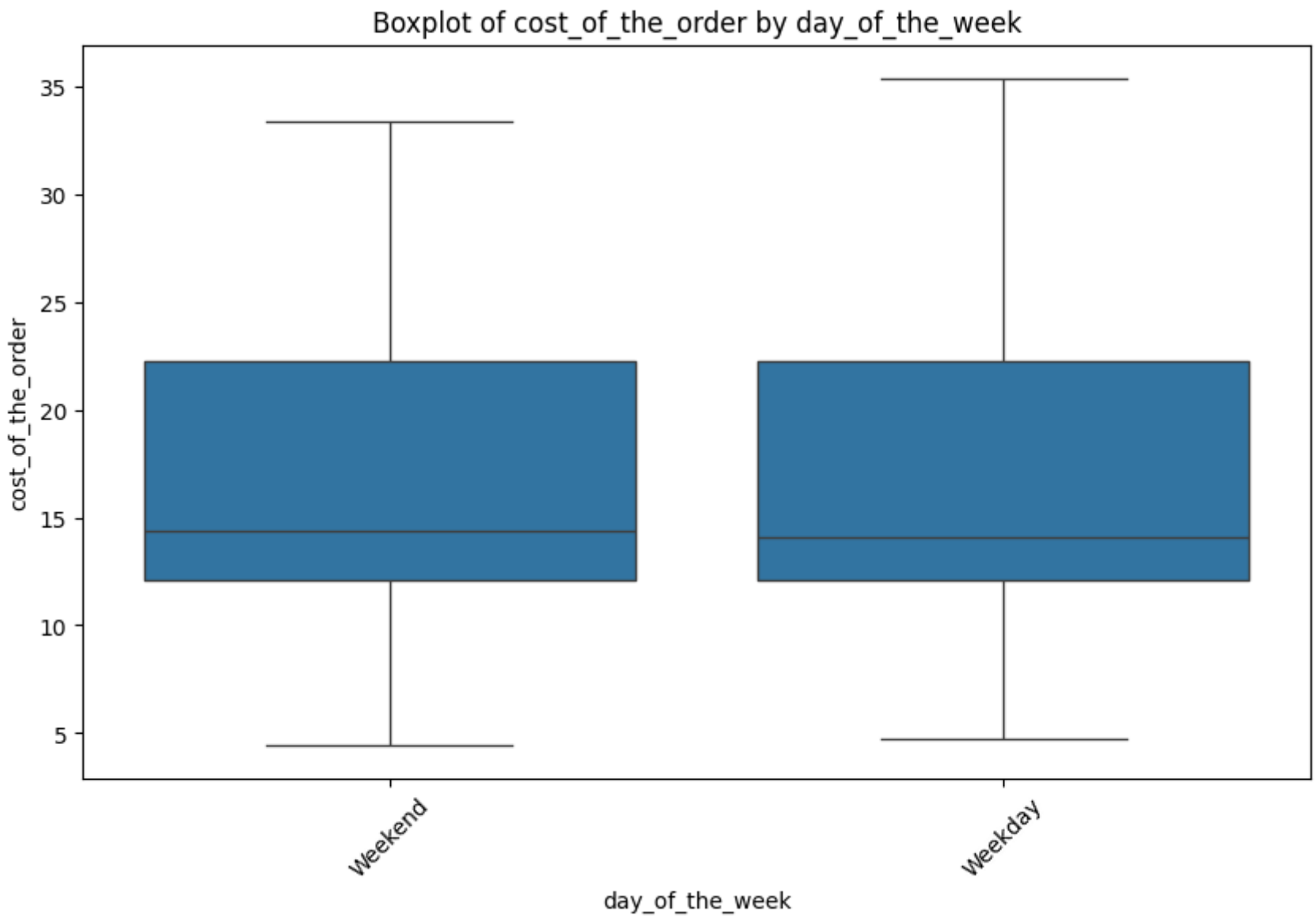

delivery_time
30
25
20
15
10
20
30
cost_of_the_order
20
25
30
35
food_preparation_time
15
20
25
30
delivery_time

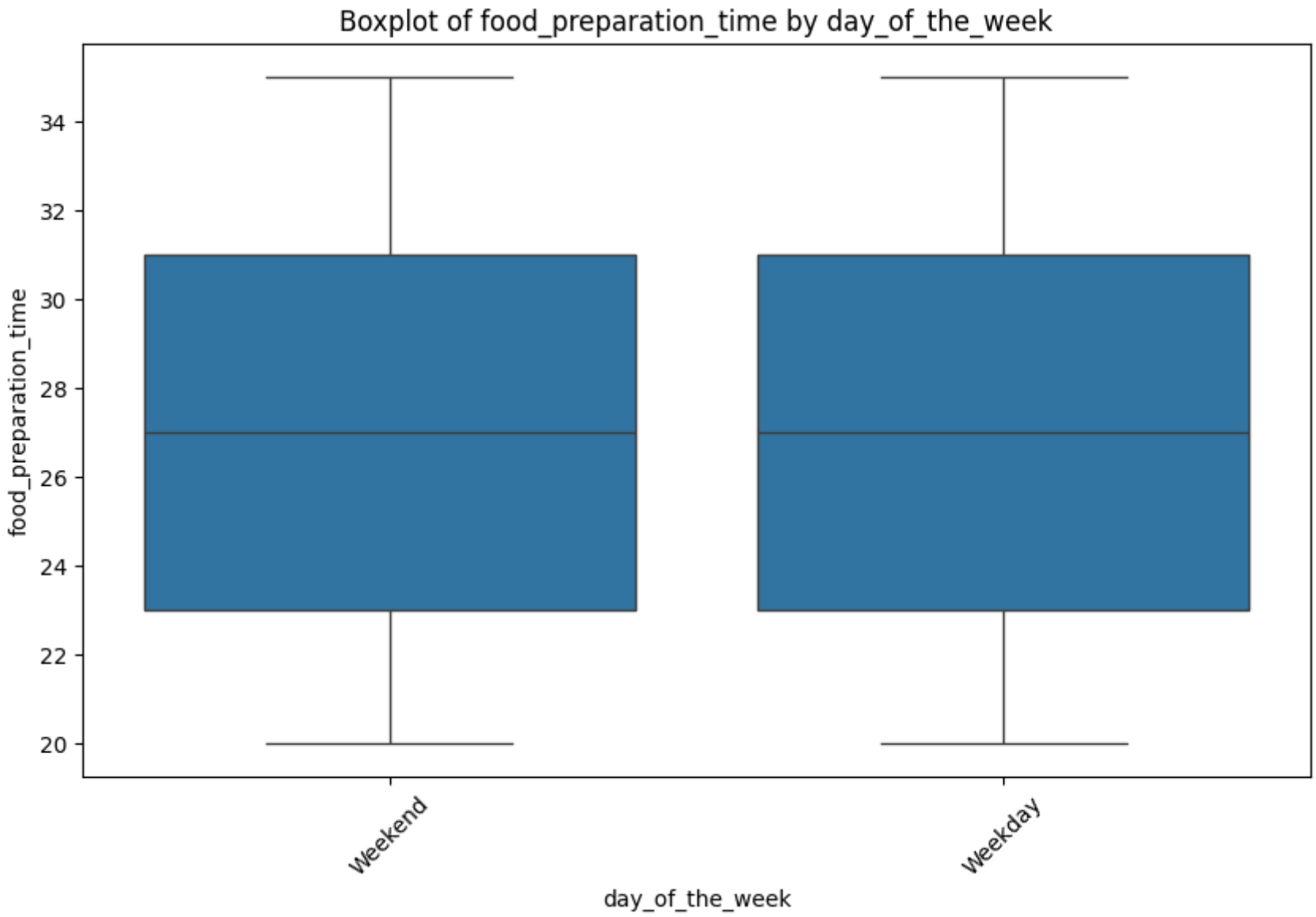

Boxplot of cost_of_the_order by cuisine_type
cost_of_the_order
35
30
25
20
15
10
5
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

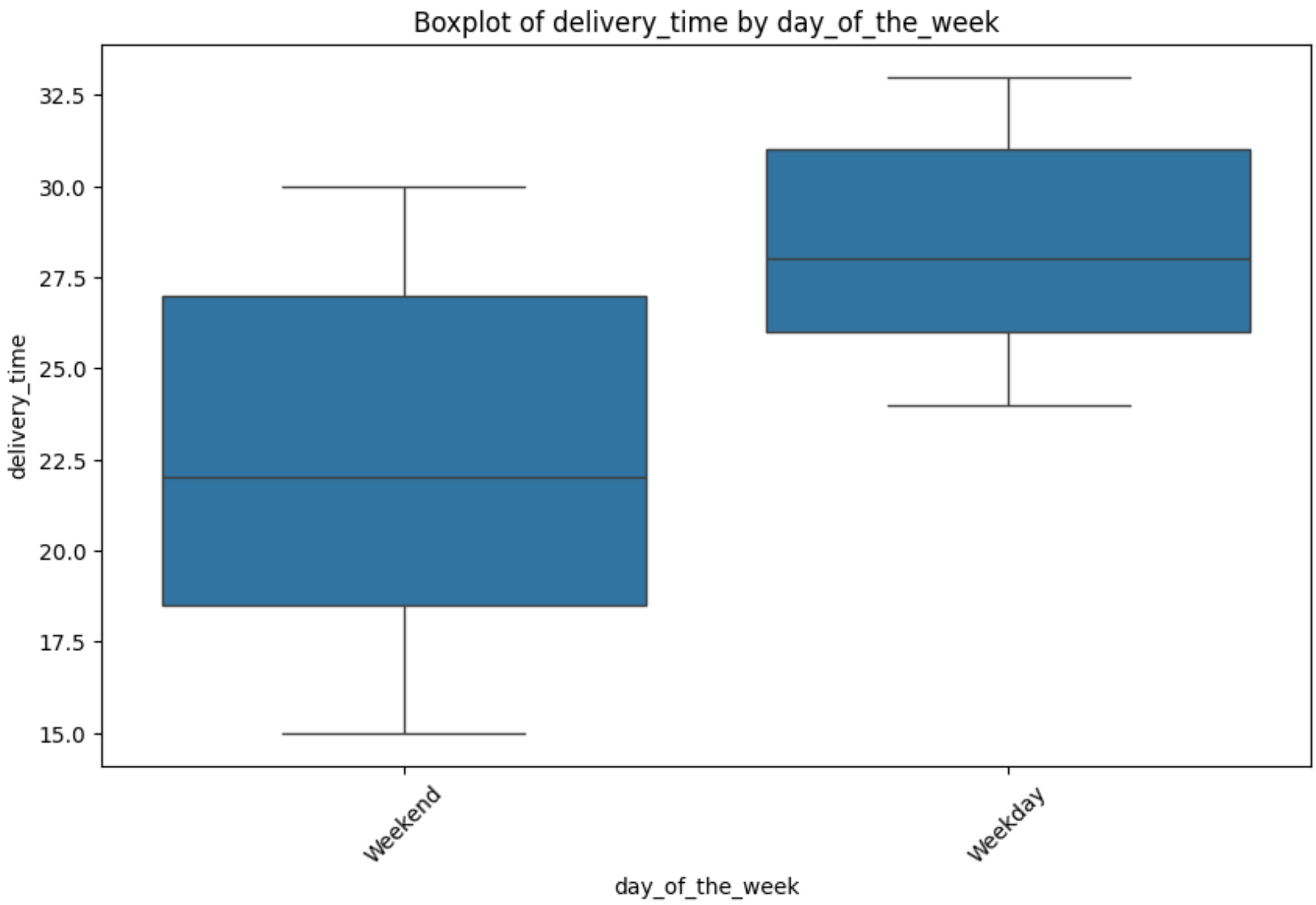

Boxplot of food_preparation_time by cuisine_type
food_preparation_time
34
32
30
28
26
24
22
20
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

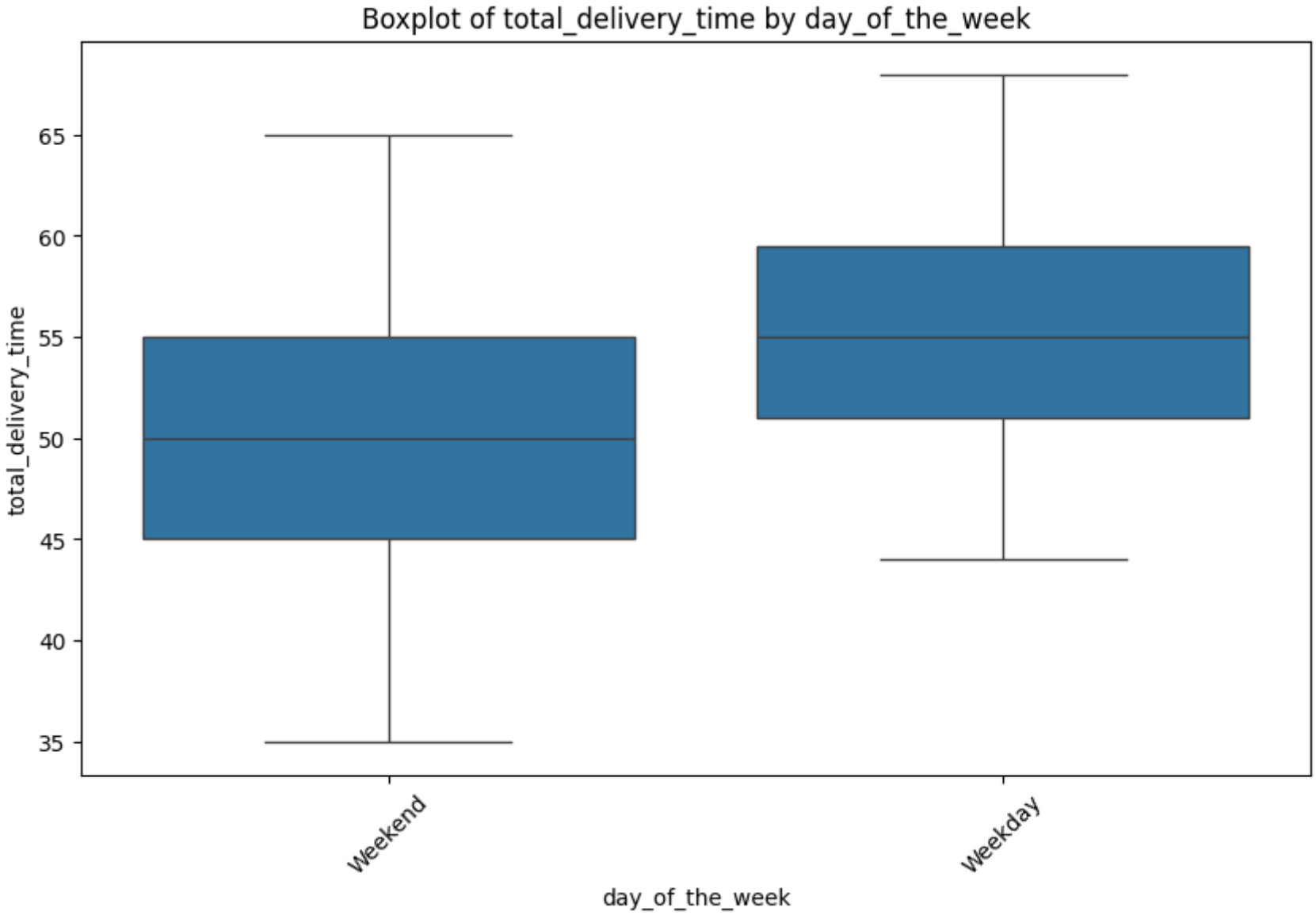

Boxplot of delivery_time by cuisine_type
delivery_time
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

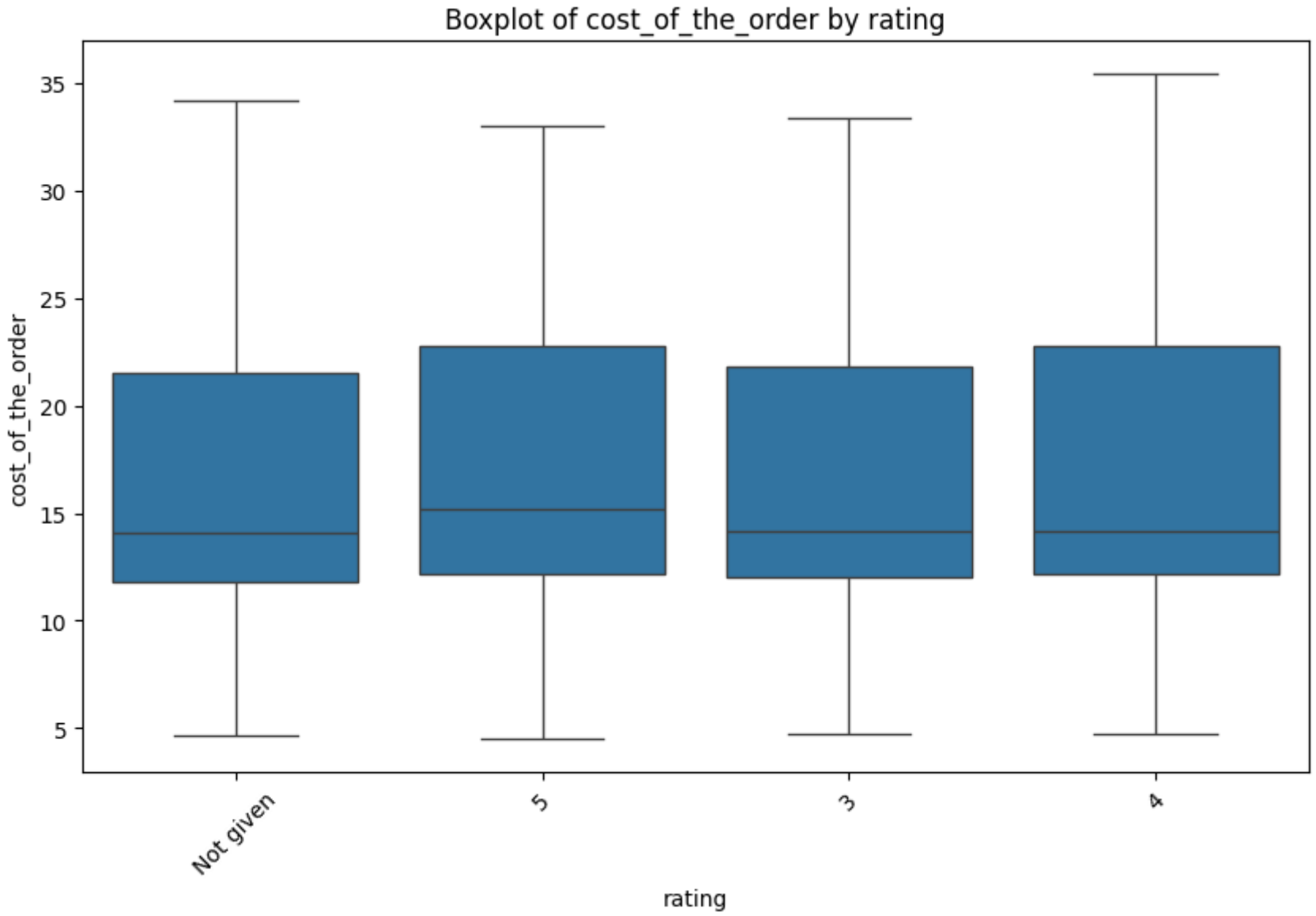

Boxplot of total_delivery_time by cuisine_type
total_delivery_time
65
60
55
50
45
40
35
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

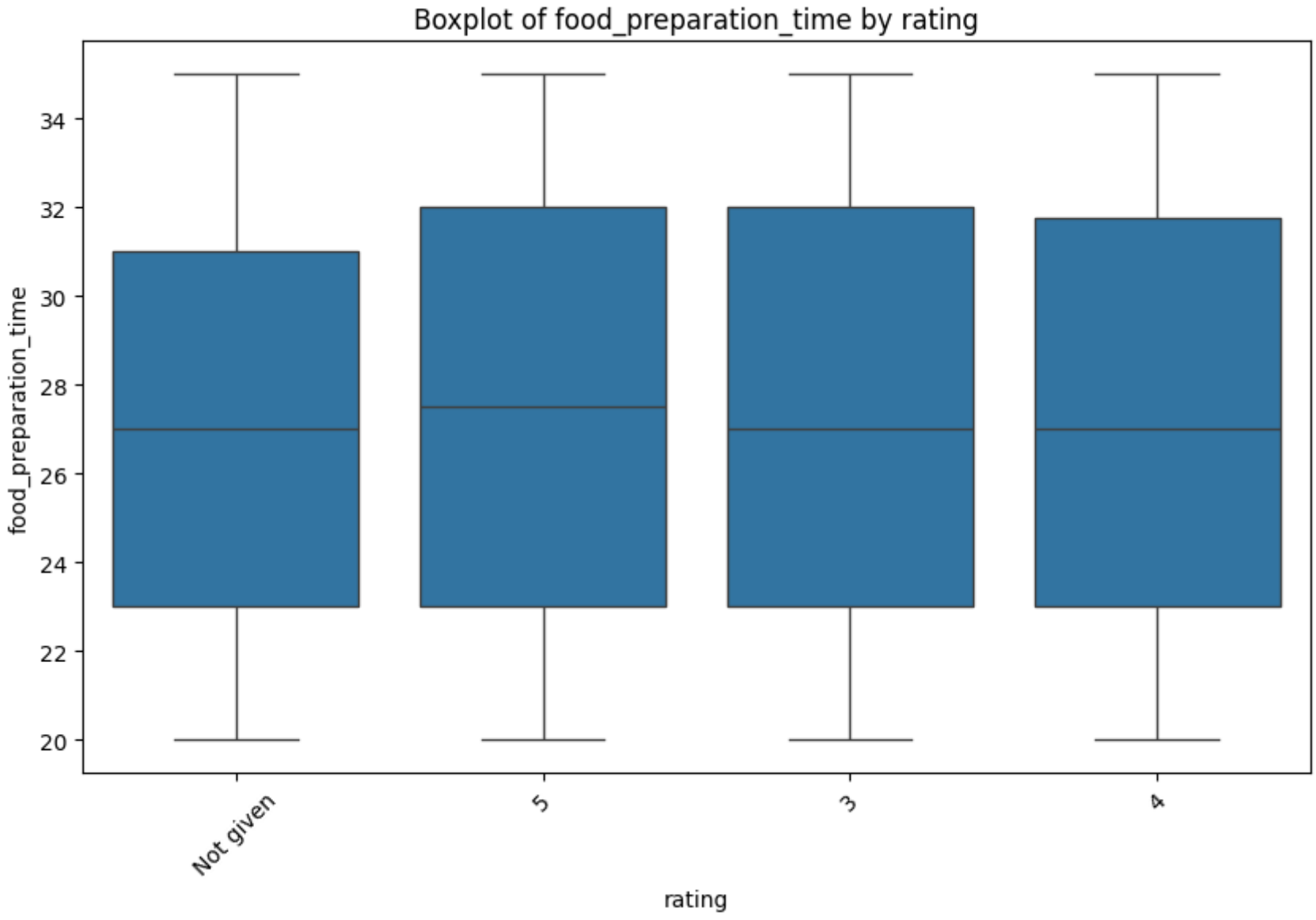

Boxplot of cost_of_the_order by day_of_the_week
cost_of_the_order
35
30
25
20
15
10
5
Weekend
Weekday
day_of_the_week

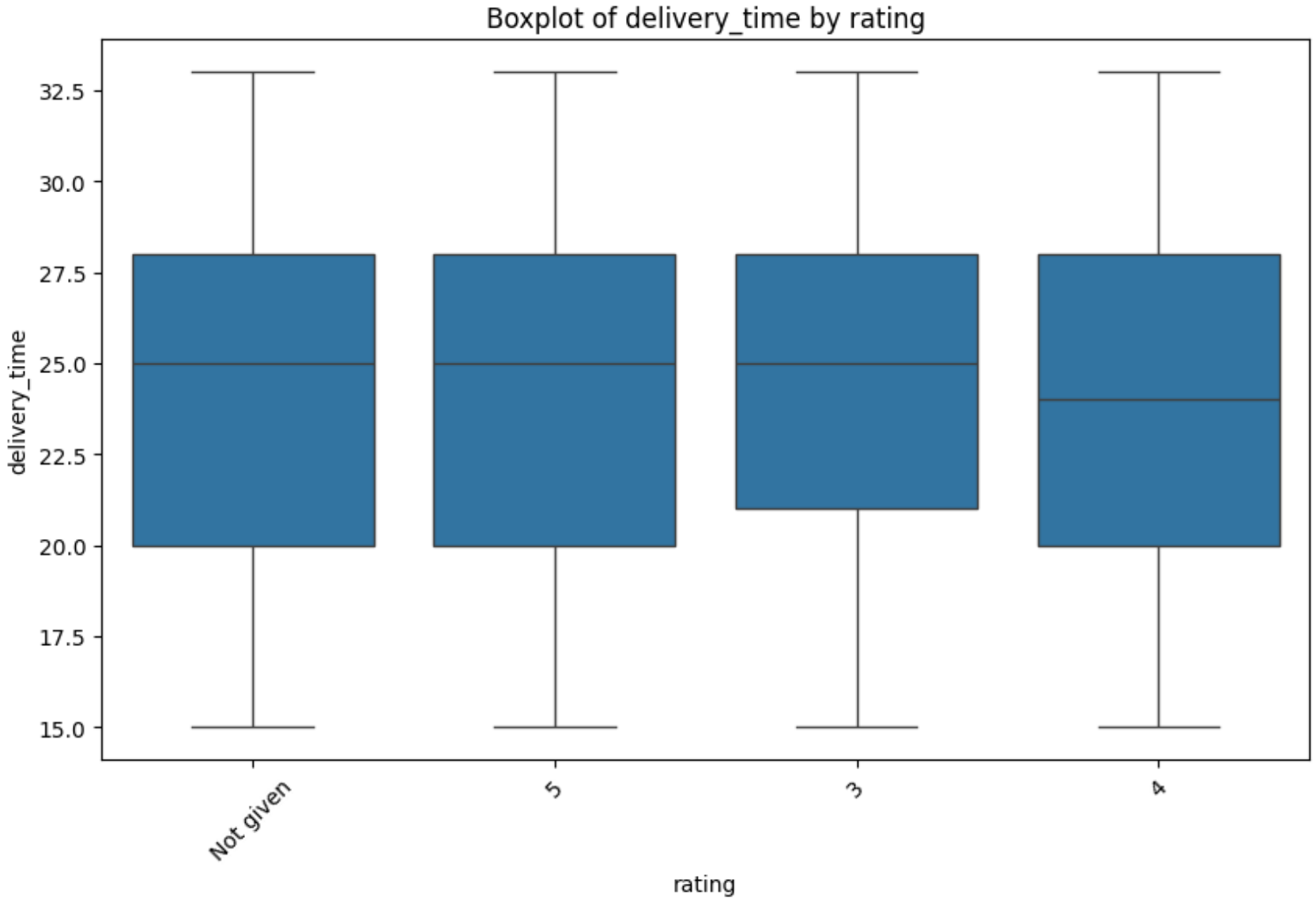

Boxplot of food_preparation_time by day_of_the_week
food_preparation_time
34
32
30
28
26
24
22
20
Weekend
Weekday
day_of_the_week

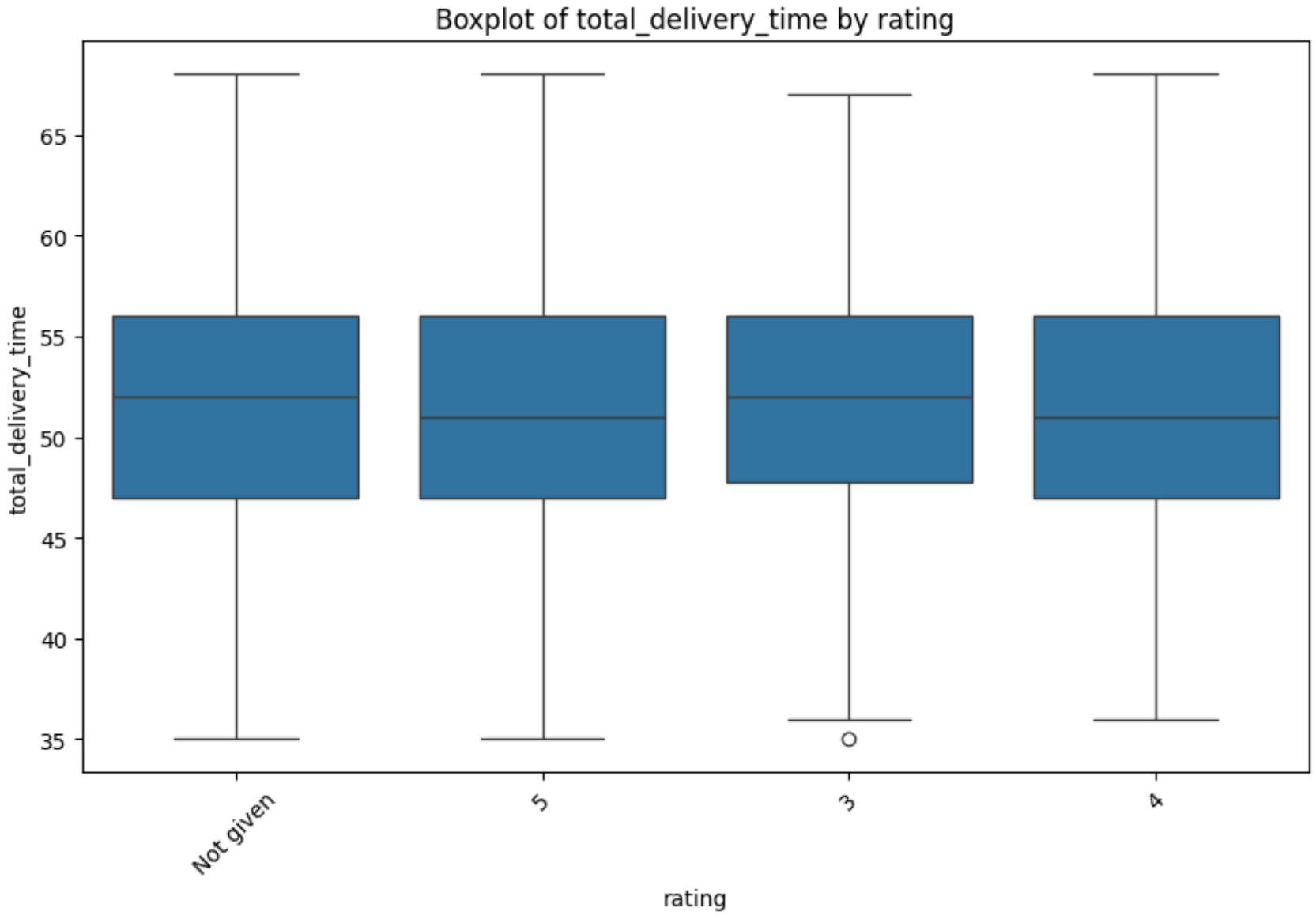

Boxplot of delivery_time by day_of_the_week
delivery_time
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
Weekend
Weekday
day_of_the_week

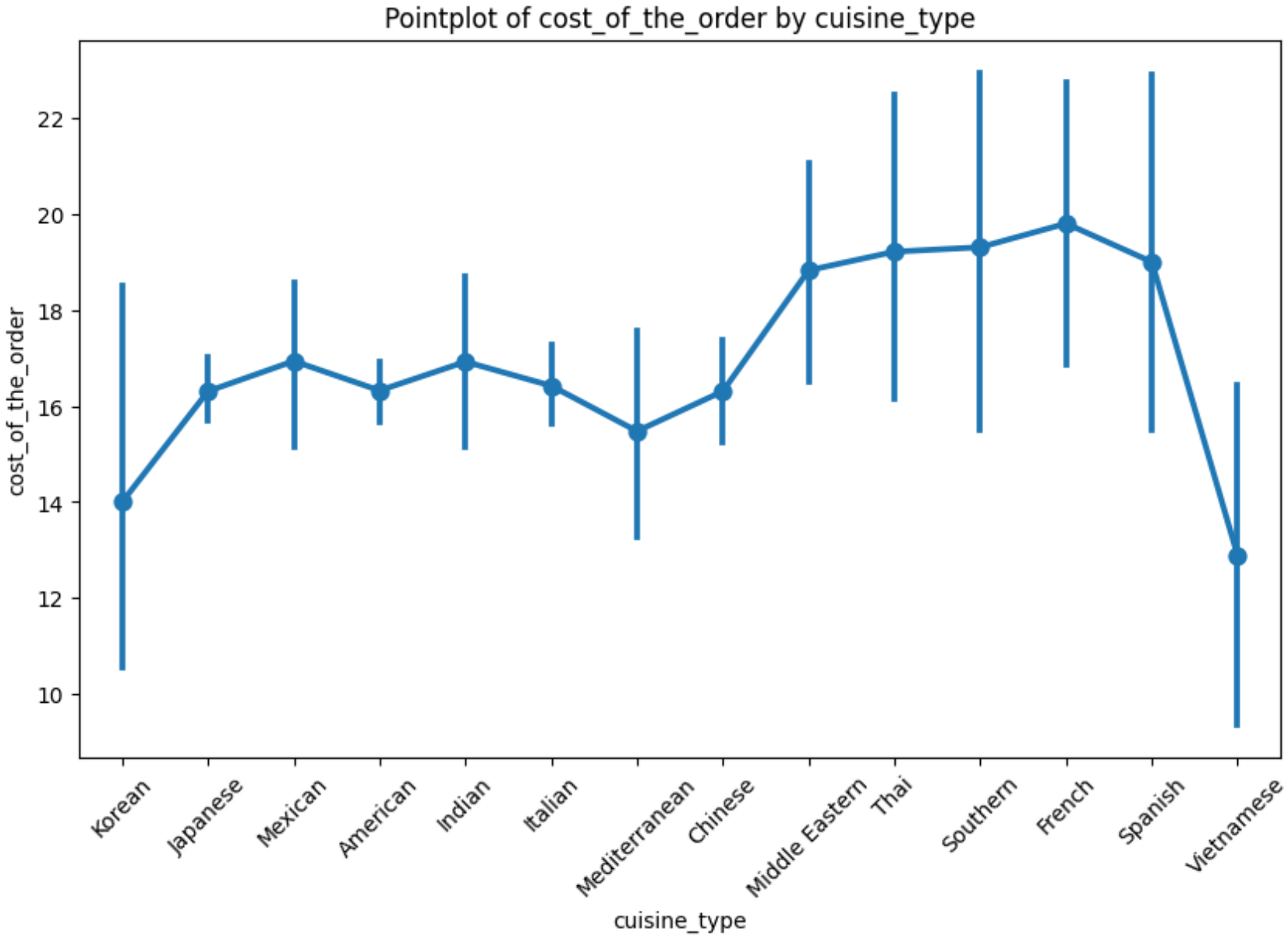

Boxplot of total_delivery_time by day_of_the_week
total_delivery_time
65
60
55
50
45
40
35
Weekend
Weekday
day_of_the_week

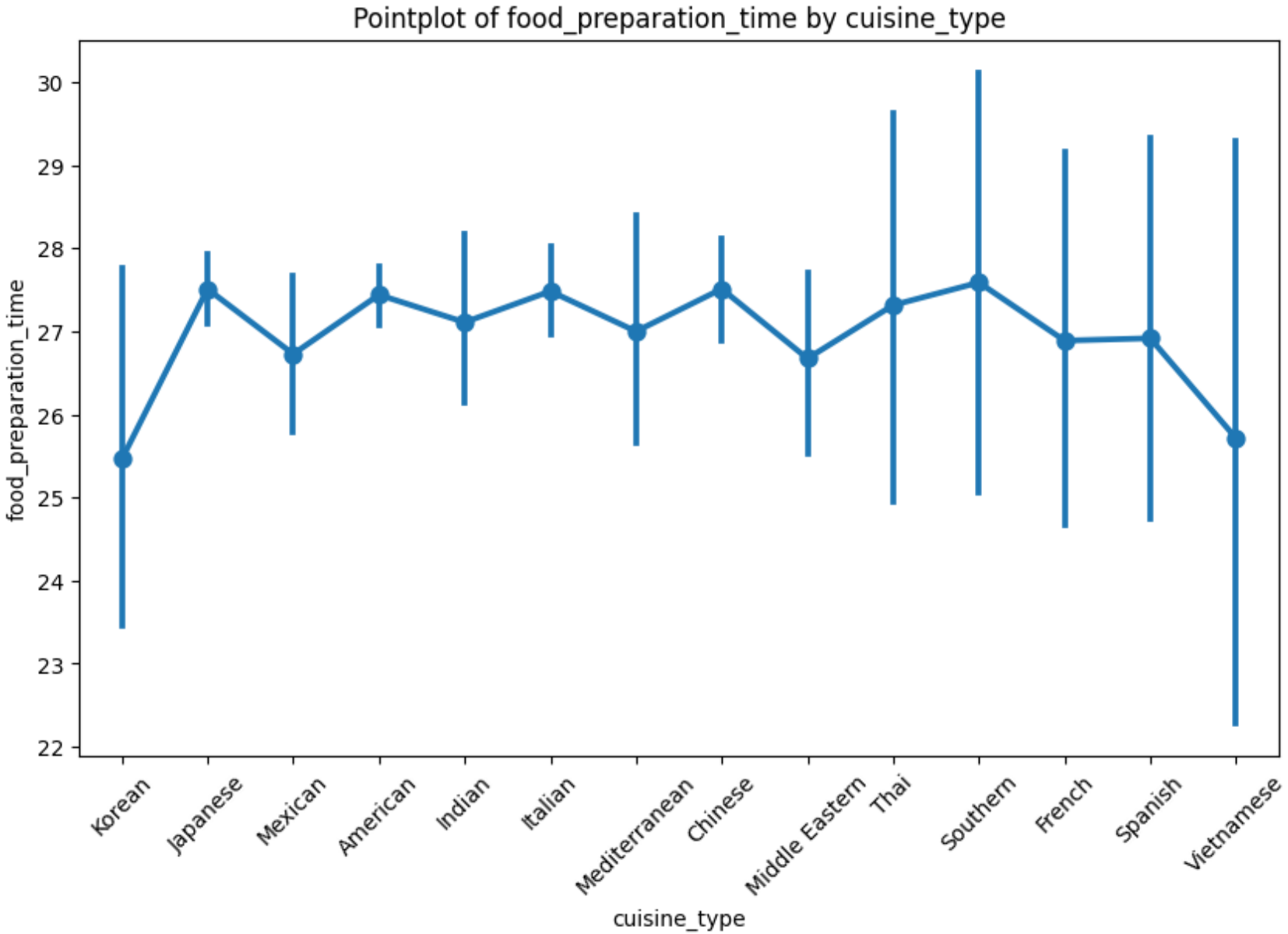

Boxplot of cost_of_the_order by rating
cost_of_the_order
35
30
25
20
15
10
5
Not given
5
3
4
rating

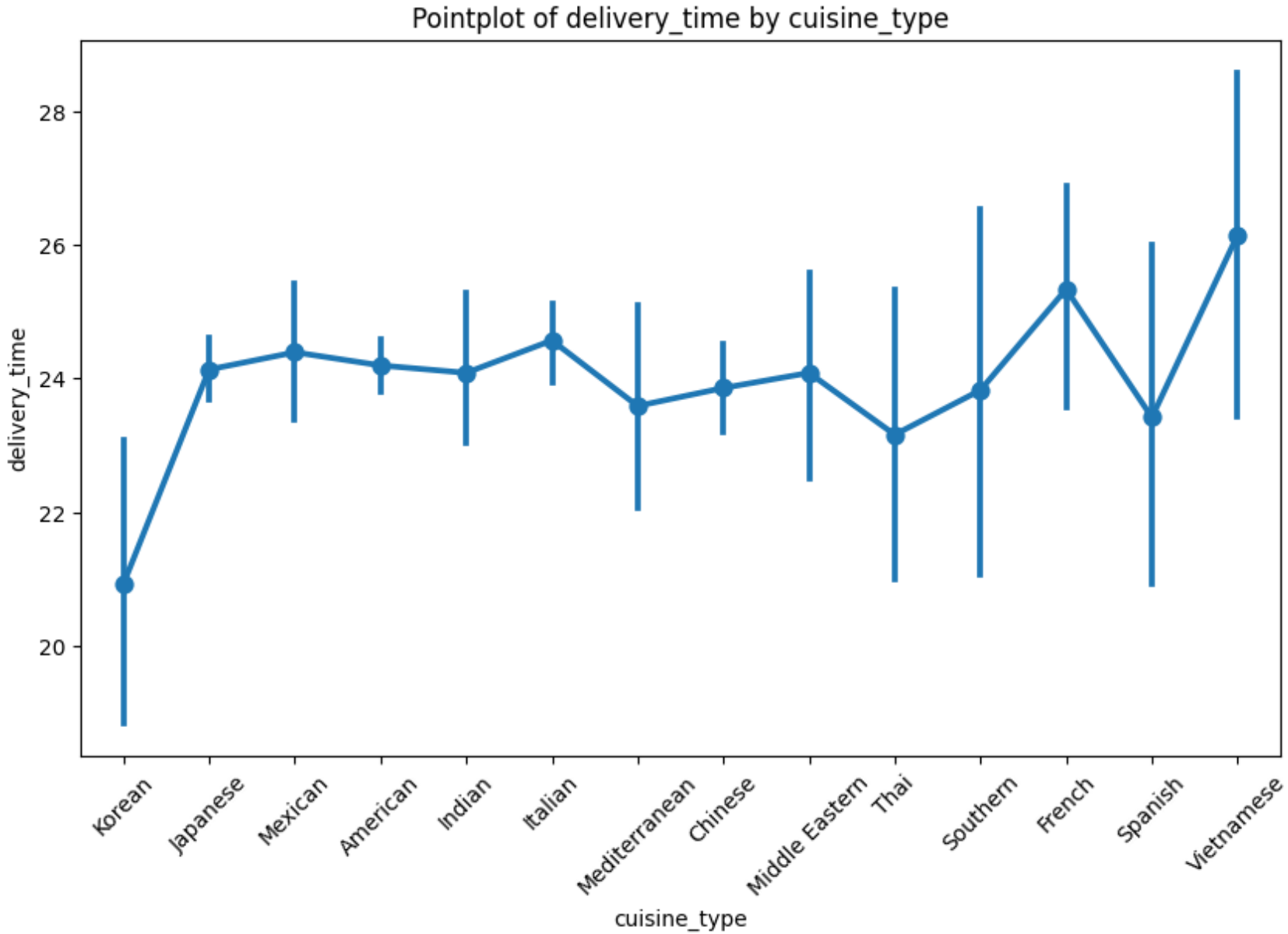

Boxplot of food_preparation_time by rating
food_preparation_time
34
32
30
28
26
24
22
20
Not given
5
3
4
rating

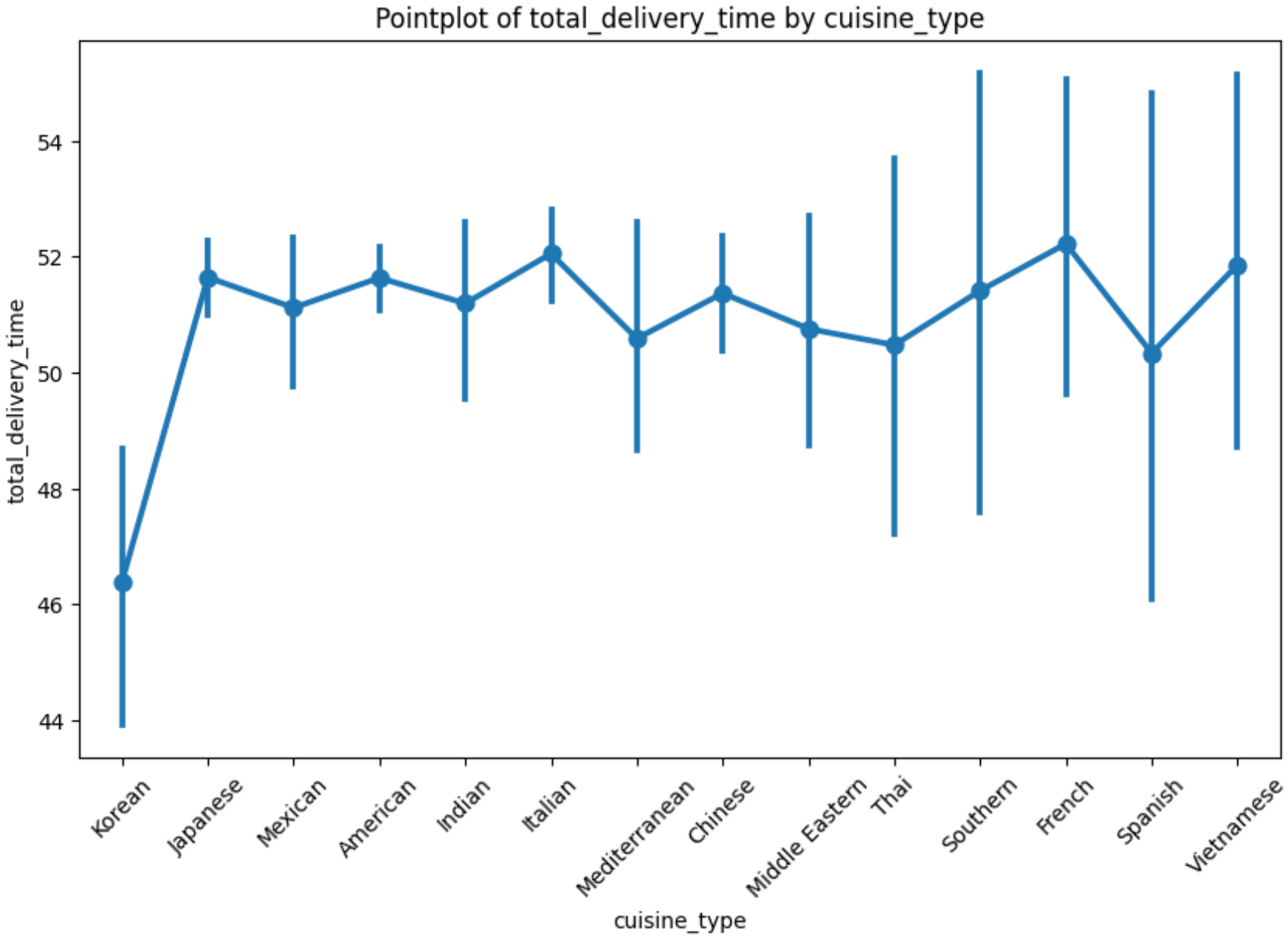

Boxplot of delivery_time by rating
delivery_time
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
Not given
5
3
4
rating

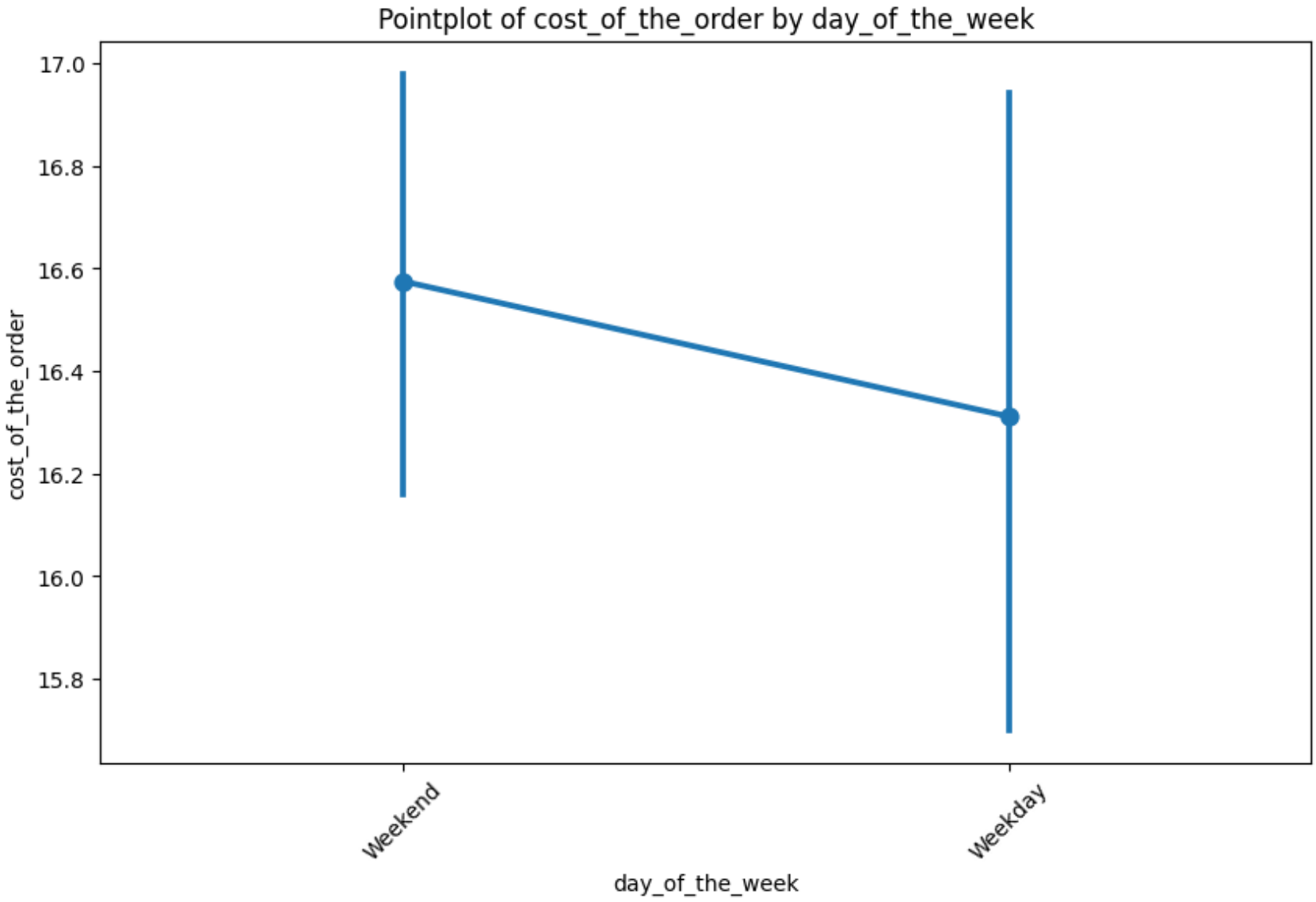

Boxplot of total_delivery_time by rating
total_delivery_time
65
60
55
50
45
40
35
Not given
5
3
4
rating

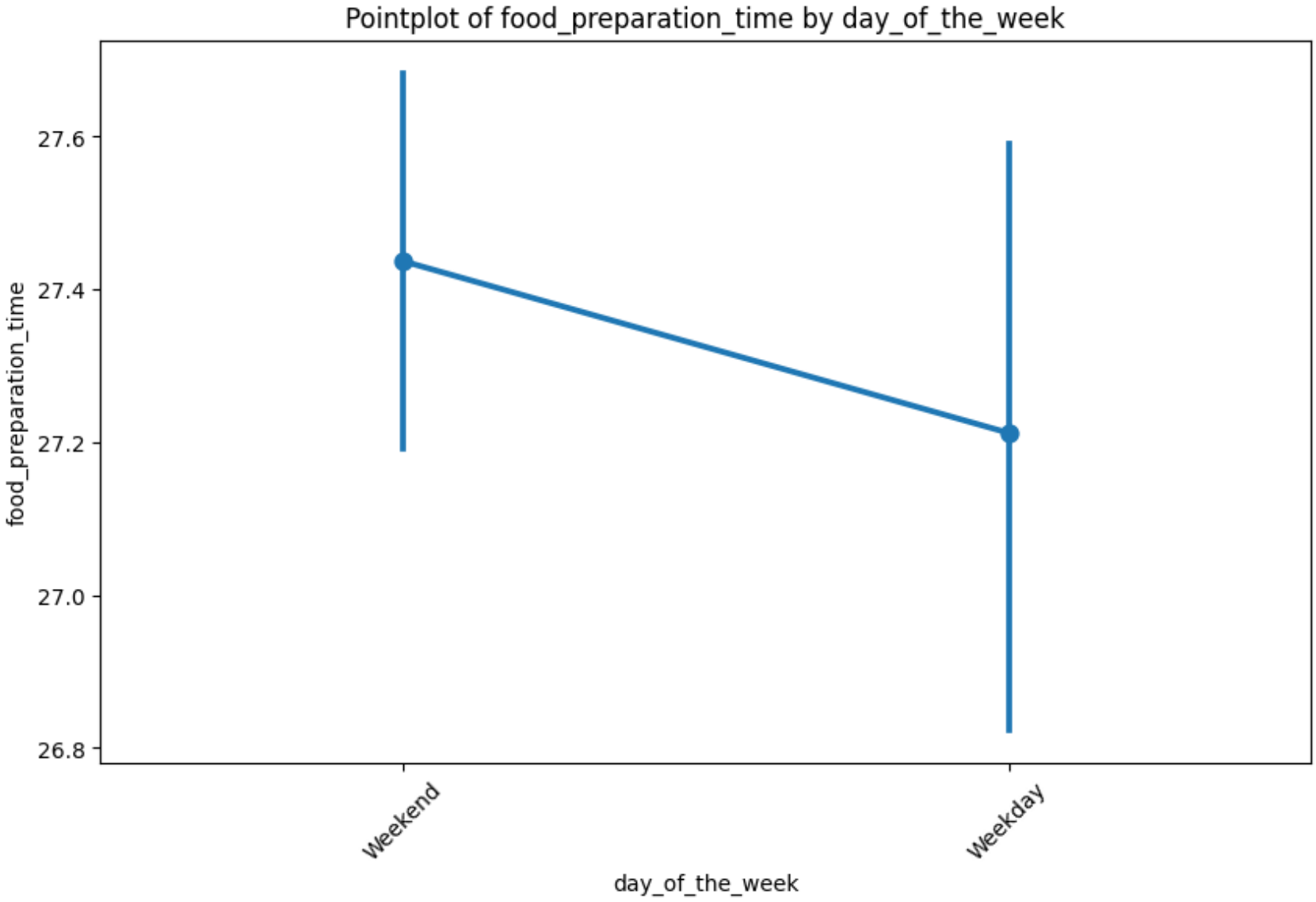

Pointplot of cost_of_the_order by cuisine_type
cost_of_the_order
22
20
18
16
14
12
10
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

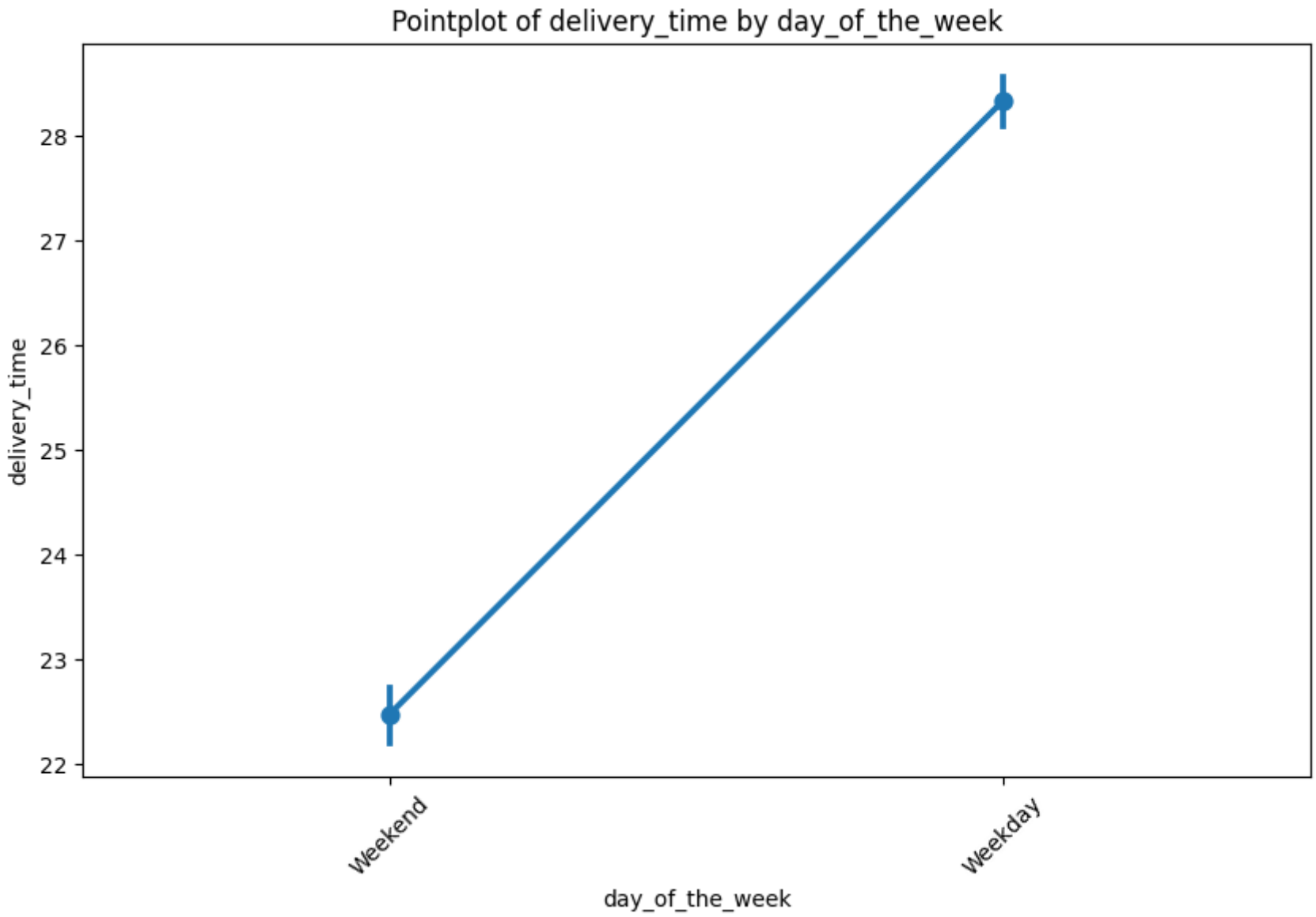

Pointplot of food_preparation_time by cuisine_type
food_preparation_time
30
29
28
27
26
25
24
23
22
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

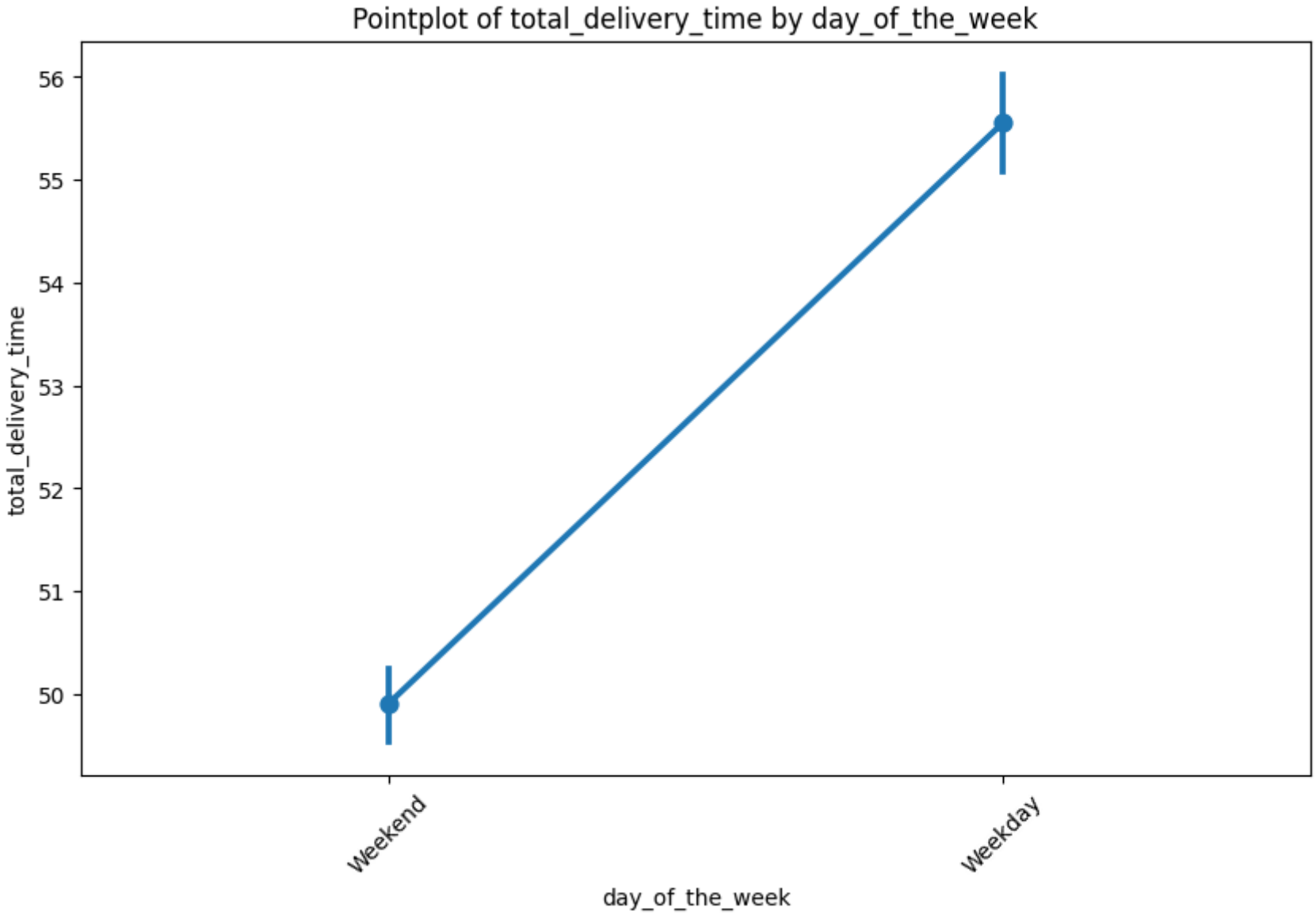

Pointplot of delivery_time by cuisine_type
delivery_time
28
26
24
22
20
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

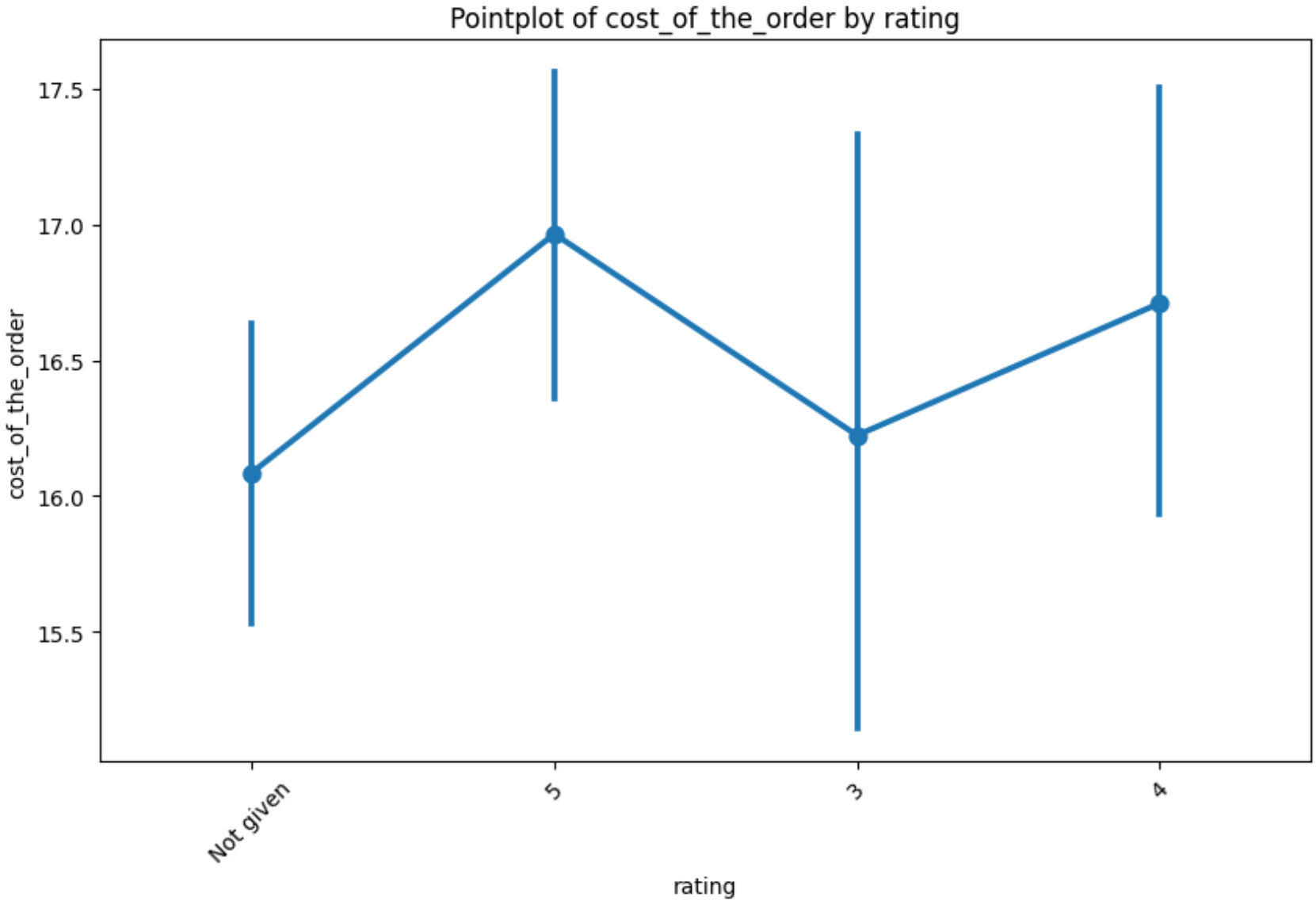

Pointplot of total_delivery_time by cuisine_type
total_delivery_time
54
52
50
48
46
44
Korean
Japanese
Mexican
American
Indian
Italian
Mediterranean
Chinese
Middle Eastern
Thai
Southern
French
Spanish
Vietnamese
cuisine_type

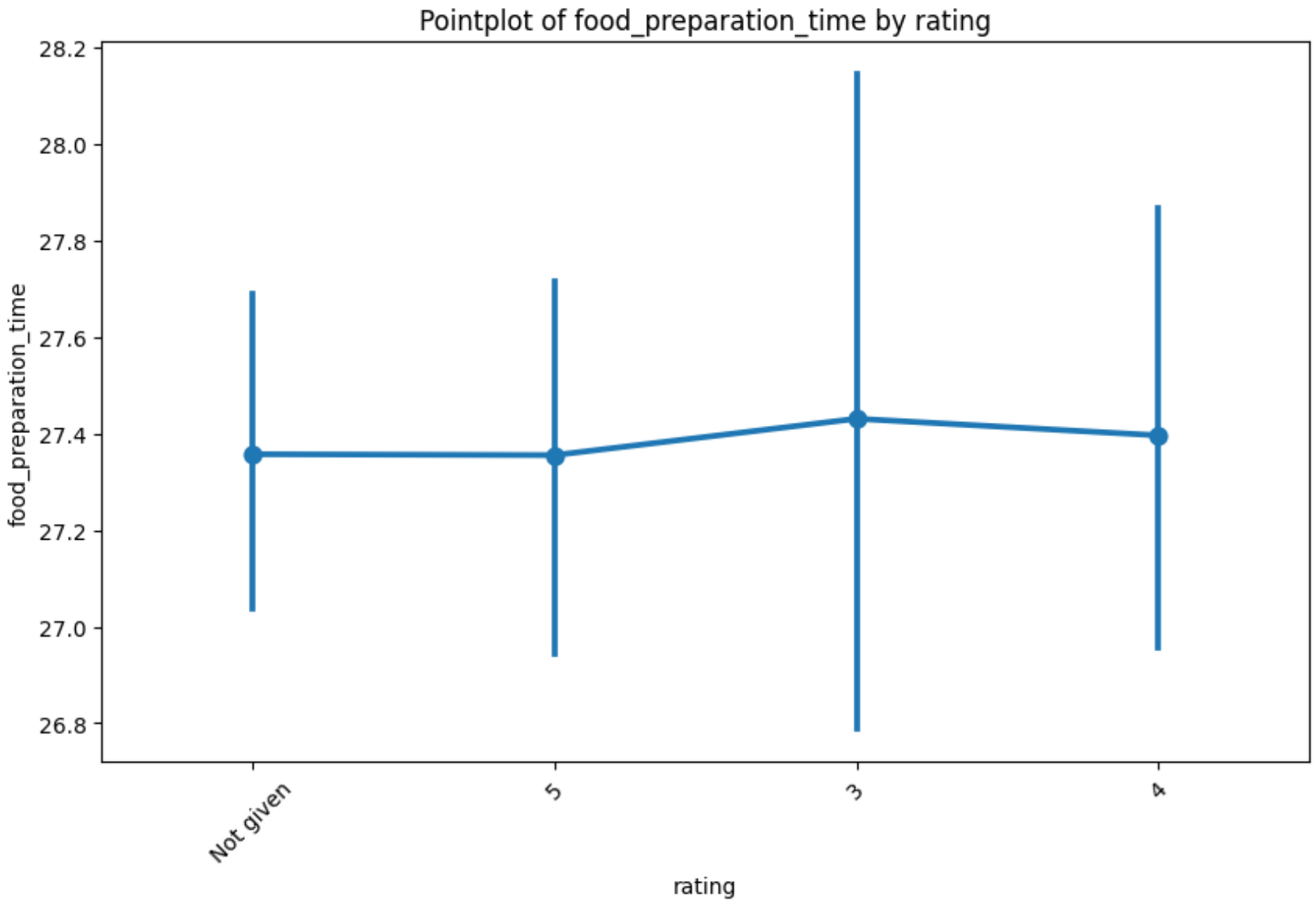

Pointplot of cost_of_the_order by day_of_the_week
17.0
16.8
16.6
16.4
16.2
16.0
15.8
cost_of_the_order
Weekend
Weekday
day_of_the_week


Pointplot of food_preparation_time by day_of_the_week
food_preparation_time
27.6
27.4
27.2
27.0
26.8
Weekend
Weekday
day_of_the_week

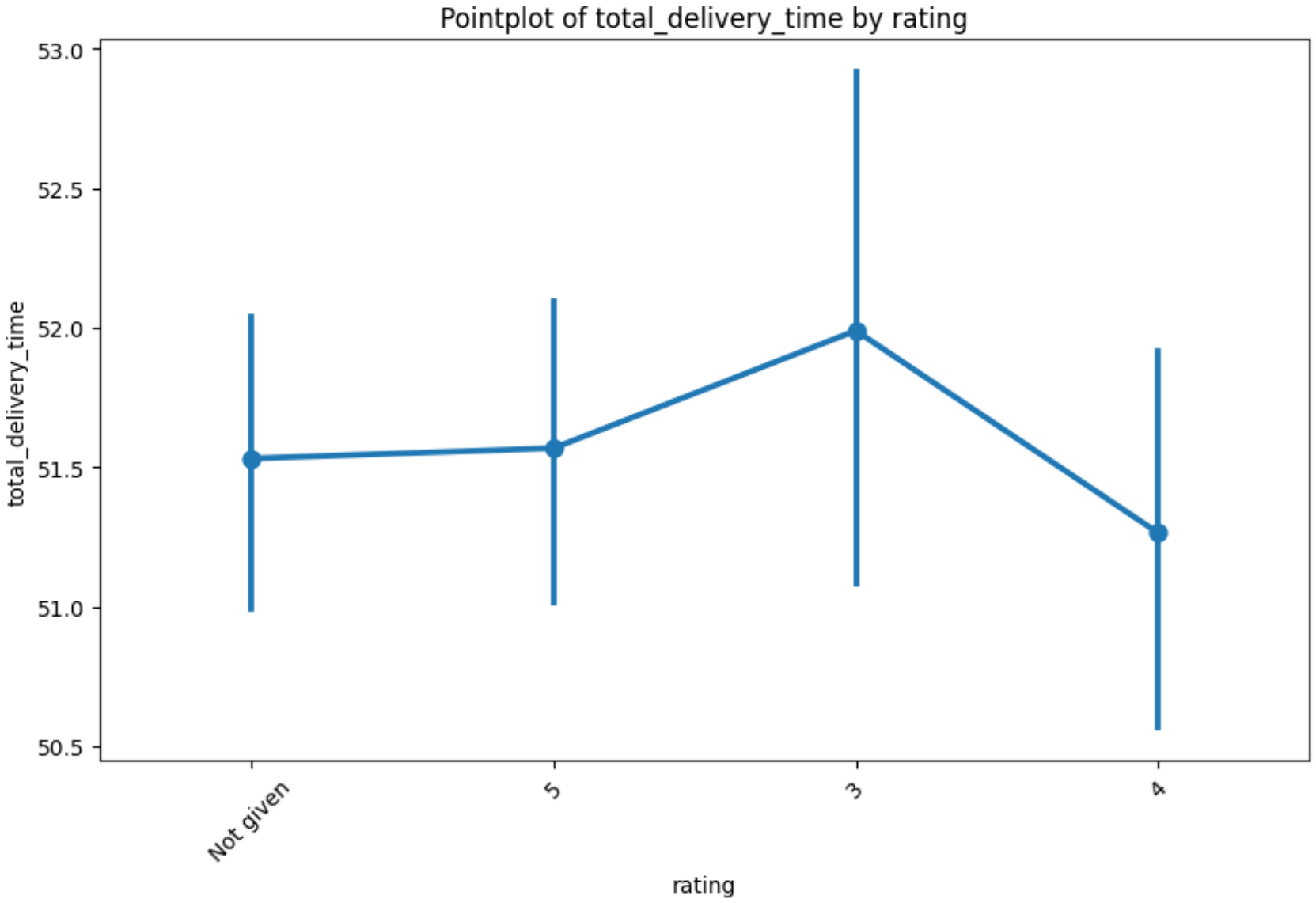

Pointplot of delivery_time by day_of_the_week
delivery_time
28
27
26
25
24
23
22
Weekend
Weekday
day_of_the_week


Pointplot of total_delivery_time by day_of_the_week
total_delivery_time
56
55
54
53
52
51
50
Weekend
Weekday
day_of_the_week

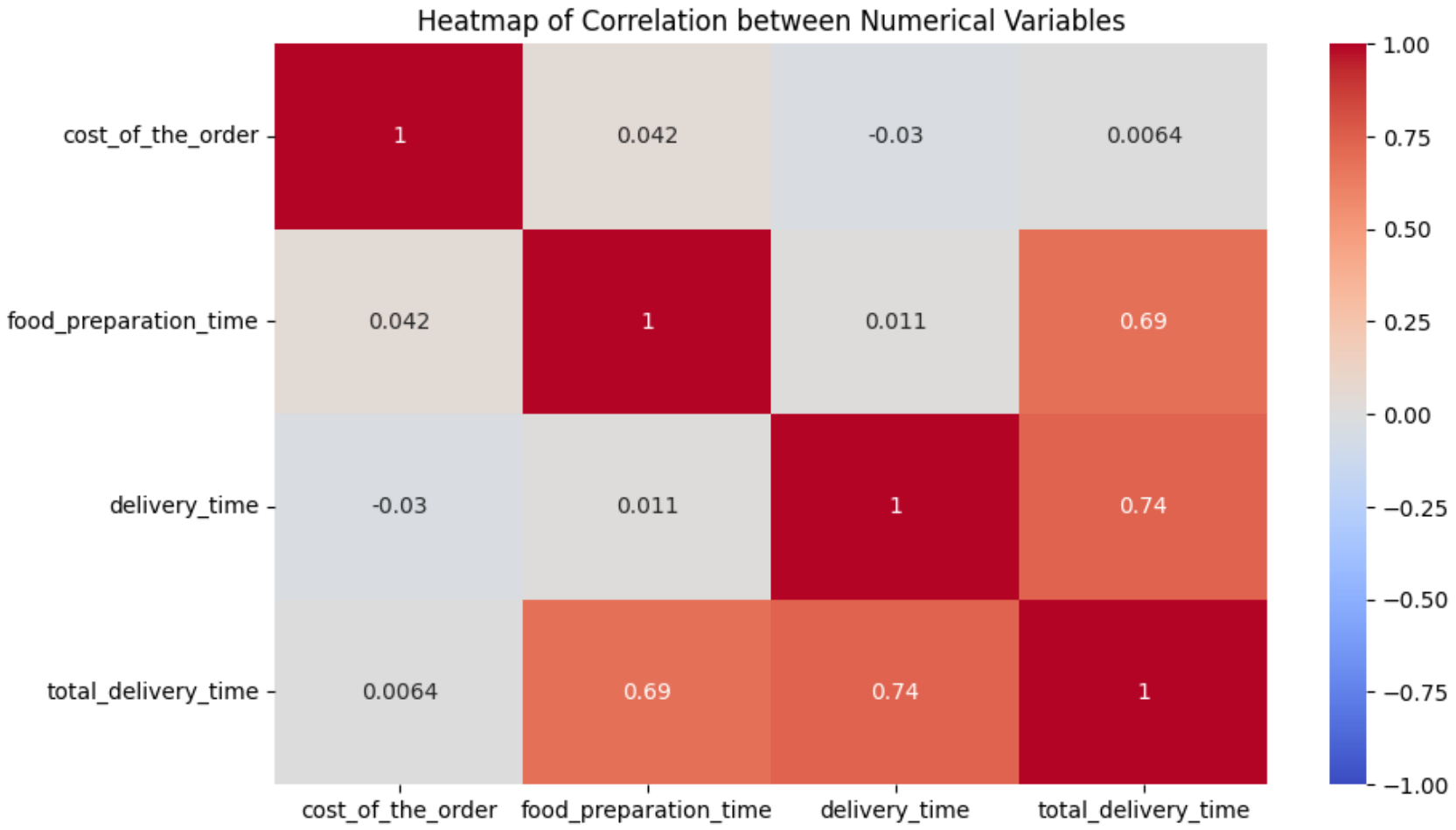

Pointplot of cost_of_the_order by rating
cost_of_the_order
17.5
17.0
16.5
16.0
15.5
Not given
5
3
4
rating


Pointplot of food_preparation_time by rating
food_preparation_time
28.2
28.0
27.8
27.6
27.4
27.2
27.0
26.8
Not given
5
3
4
rating


Pointplot of delivery_time by rating
25.25
25.00
24.75
24.50
24.25
24.00
23.75
23.50
delivery_time
Not given
5
3
4
rating



In [18]:
```
# Heatmap to explore correlation between numerical variables
plt.figure(figsize=(10, 6))
sns.heatmap(data[numerical_columns].corr(), annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Heatmap of Correlation between Numerical Variables')
plt.show()
```



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 [19]:
```
# Convert the 'rating' column to numeric, replacing 'Not given' with NaN, and store it in a new col
data['rating_numeric'] = pd.to_numeric(data['rating'], errors='coerce')

# Find the restaurants with a rating count of more than 50 and an average rating greater than 4
```

```
restaurant_ratings = data.groupby('restaurant_name')['rating_numeric'].agg(['count', 'mean'])
eligible_restaurants = restaurant_ratings[(restaurant_ratings['count'] > 50) & (restaurant_ratings[

# Print the result
print("Restaurants eligible for the promotional offer:")
print(eligible_restaurants)
```

```
Restaurants eligible for the promotional offer:
                          count      mean
restaurant_name
Blue Ribbon Fried Chicken     64  4.328125
Blue Ribbon Sushi             73  4.219178
Shake Shack                  133  4.278195
The Meatball Shop             84  4.511905
```

Observations:

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 [20]:
```
# Calculate the revenue for each order based on the given conditions
data['revenue'] = data['cost_of_the_order'].apply(lambda x: 0.25 * x if x > 20 else (0.15 * x if x

# Calculate the total revenue
total_revenue = data['revenue'].sum()

# Print the result
print(f'The net revenue generated by the company across all orders is: ${total_revenue:.2f}')
```

```
The net revenue generated by the company across all orders is: $6166.30
```

Observations:

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 [21]:
```
# Calculate the number of orders that take more than 60 minutes to get delivered
orders_more_than_60_min = data[data['total_delivery_time'] > 60].shape[0]

# Calculate the percentage of such orders
percentage_orders_more_than_60_min = (orders_more_than_60_min / num_rows) * 100

# Print the result
print(f'The percentage of orders that take more than 60 minutes to get delivered: {percentage_order
```

```
The percentage of orders that take more than 60 minutes to get delivered: 10.54%
```

Observations:

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 [22]:
```
# Calculate the mean delivery time for weekdays and weekends
mean_delivery_time_weekday = data[data['day_of_the_week'] == 'Weekday']['delivery_time'].mean()
mean_delivery_time_weekend = data[data['day_of_the_week'] == 'Weekend']['delivery_time'].mean()

# Print the results
print(f'The mean delivery time on weekdays is: {mean_delivery_time_weekday:.2f} minutes')
print(f'The mean delivery time on weekends is: {mean_delivery_time_weekend:.2f} minutes')
```

```
The mean delivery time on weekdays is: 28.34 minutes
The mean delivery time on weekends is: 22.47 minutes
```

Observations:

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:

1. **Order Distribution**: The data consists of 1898 orders with various cuisine types and ratings. The most popular cuisine on weekends is American, accounting for approximately 30.72% of the orders.
2. **Order Cost**: About 29.24% of the orders cost more than 20 dollars.
3. **Preparation and Delivery Time**: The average food preparation time is approximately 27.37 minutes, with a minimum of 20 minutes and a maximum of 35 minutes. The mean delivery time is around 24.16 minutes, with weekdays having a higher mean delivery time (28.34 minutes) compared to weekends (22.47 minutes).
4. **Customer Ratings**: There are 736 orders that are not rated. Among the rated orders, the average rating varies across different restaurants.
5. **Top Restaurants**: The top 5 restaurants in terms of the number of orders received are Shake Shack, The Meatball Shop, Blue Ribbon Sushi, Blue Ribbon Fried Chicken, and Parm.
6. **Revenue**: The net revenue generated by the company across all orders is approximately $6166.30.
7. **Delivery Time Analysis**: About 10.54% of the orders take more than 60 minutes to get delivered from the time the order is placed.

In [23]:
```
# Calculate the total number of restaurants
total_restaurants = data['restaurant_name'].nunique()

# Calculate the cumulative revenue percentage for each restaurant
restaurant_revenue = data.groupby('restaurant_name')['revenue'].sum().sort_values(ascending=False)
cumulative_revenue_percentage = restaurant_revenue.cumsum() / restaurant_revenue.sum() * 100

# Select the restaurants that provide 80% of the revenue
top_restaurants_80_percent = cumulative_revenue_percentage[cumulative_revenue_percentage <= 80]

# Calculate the number of restaurants that provide 80% of the revenue
num_top_restaurants_80_percent = top_restaurants_80_percent.shape[0]

# Calculate the percentage of restaurants that provide 80% of the revenue
```

```
percentage_top_restaurants_80_percent = (num_top_restaurants_80_percent / total_restaurants) * 100

# Print the results
print(f'Total number of restaurants: {total_restaurants}')
print(f'Number of restaurants that provide 80% of the revenue: {num_top_restaurants_80_percent}')
print(f'Percentage of restaurants that provide 80% of the revenue: {percentage_top_restaurants_80_p
```

```
Total number of restaurants: 178
Number of restaurants that provide 80% of the revenue: 42
Percentage of restaurants that provide 80% of the revenue: 23.60%
```

Recommendations:

1. **Focus on Top Revenue-Generating Restaurants**: Approximately 80% of the revenue comes from 42 restaurants, which is about 23.60% of the total restaurants. Focus promotional offers and marketing efforts on these top revenue-generating restaurants to maximize the impact. These restaurants include Shake Shack, The Meatball Shop, Blue Ribbon Sushi, Blue Ribbon Fried Chicken, and Parm, among others. Providing additional support and visibility to these restaurants can help drive more orders and increase overall revenue.

2. **Promote Popular Cuisines**: Since American cuisine is the most popular on weekends, accounting for approximately 30.72% of the orders, consider promoting American restaurants more aggressively during weekends through targeted advertisements and special offers.

3. **Encourage Customer Feedback**: There are 736 orders that are not rated. Implement incentives for customers to provide ratings and feedback on their orders. This will help gather more data on customer preferences and improve service quality.

4. **Support High-Performing Restaurants**: Provide additional support and promotional offers to restaurants with high ratings and a significant number of orders, such as Shake Shack and The Meatball Shop. Shake Shack received 219 orders and has an average rating of 4.28, while The Meatball Shop received 132 orders and has an average rating of 4.51.

5. **Analyze High-Cost Orders**: About 29.24% of the orders cost more than 20 dollars. Investigate the reasons behind the high percentage of orders costing more than 20 dollars. Consider offering discounts or loyalty

programs to encourage repeat orders from customers who place high-cost orders.

6. **Optimize Food Preparation Time**: The average food preparation time is approximately 27.37 minutes, with a minimum of 20 minutes and a maximum of 35 minutes. Work with restaurants to streamline their food preparation processes and reduce the average preparation time, which will contribute to faster overall delivery times.

7. **Improve Delivery Efficiency**: Focus on reducing the delivery time, especially on weekdays, to enhance customer satisfaction. The mean delivery time on weekdays is 28.34 minutes, which is higher than the 22.47 minutes on weekends. Implement strategies such as optimizing delivery routes and increasing the number of delivery personnel during peak hours.
