Restaurant Revenue Prediction

December 11, 2024

1 Prediction of Resturant Revenue

1.1 Importing Libraries

[3]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
```

[4]:
```
import warnings
warnings.filterwarnings('ignore')
```

1.2 Importing Dataset

[6]:
```
revenue_data = pd.read_csv('Project-4- Predict Restaurant Revenue.csv')
revenue_data.head()
```

[6]:

	Name	Location	Cuisine	Rating	Seating Capacity	Average Meal Price	Marketing Budget	Social Media Followers	Chef Experience Years	Number of Reviews	Avg Review Length
0	Restaurant 0	Rural	Japanese	4.0	38	73.98	2224	23406	13	185	161.924906
1	Restaurant 1	Downtown	Mexican	3.2	76	28.11	4416	42741	8	533	148.759717
2	Restaurant 2	Rural	Italian	4.7	48	48.29	2796	37285	18	853	56.849189
3	Restaurant 3	Rural	Italian	4.4	34	51.55	1167	15214	13	82	205.433265
4	Restaurant 4	Downtown	Japanese	4.9	88	75.98	3639	40171	9	78	241.681584

```
   Ambience Score  Service Quality Score Parking Availability  \
0             1.3                    7.0                  Yes
1             2.6                    3.4                  Yes
2             5.3                    6.7                   No
3             4.6                    2.8                  Yes
4             8.6                    2.1                   No

   Weekend Reservations  Weekday Reservations     Revenue
0                    13                     4   638945.52
1                    48                     6   490207.83
2                    27                    14   541368.62
3                     9                    17   404556.80
4                    37                    26  1491046.35
```

1.3 Basic Data Analysis

1.3.1 First Five Rows

[9]:
```
revenue_data.head()
```

[9]:
```
           Name  Location   Cuisine  Rating  Seating Capacity  \
0  Restaurant 0     Rural  Japanese     4.0                38
1  Restaurant 1  Downtown   Mexican     3.2                76
2  Restaurant 2     Rural   Italian     4.7                48
3  Restaurant 3     Rural   Italian     4.4                34
4  Restaurant 4  Downtown  Japanese     4.9                88

   Average Meal Price  Marketing Budget  Social Media Followers  \
0               73.98              2224                   23406
1               28.11              4416                   42741
2               48.29              2796                   37285
3               51.55              1167                   15214
4               75.98              3639                   40171

   Chef Experience Years  Number of Reviews  Avg Review Length  \
0                     13                185         161.924906
1                      8                533         148.759717
2                     18                853          56.849189
3                     13                 82         205.433265
4                      9                 78         241.681584

   Ambience Score  Service Quality Score Parking Availability  \
0             1.3                    7.0                  Yes
1             2.6                    3.4                  Yes
2             5.3                    6.7                   No
3             4.6                    2.8                  Yes
4             8.6                    2.1                   No
```

	Weekend Reservations	Weekday Reservations	Revenue
0	13	4	638945.52
1	48	6	490207.83
2	27	14	541368.62
3	9	17	404556.80
4	37	26	1491046.35

1.3.2 Last Five Rows

[11]:
```
revenue_data.tail()
```

[11]:

	Name	Location	Cuisine	Rating	Seating Capacity	Average Meal Price	Marketing Budget	Social Media Followers	Chef Experience Years	Number of Reviews	Avg Review Length	Ambience Score	Service Quality Score	Parking Availability	Weekend Reservations	Weekday Reservations	Revenue
8363	Restaurant 8363	Suburban	Indian	3.4	54	34.85	1102	11298	11	380	253.919515	9.5	5.0	Yes	37	0	434653.45
8364	Restaurant 8364	Rural	Indian	3.7	49	36.88	1988	20432	9	713	175.590195	2.7	2.6	No	37	21	414977.92
8365	Restaurant 8365	Downtown	Italian	4.7	88	46.87	5949	63945	6	436	222.953647	4.8	1.7	Yes	83	21	930395.87
8366	Restaurant 8366	Rural	American	3.1	31	44.53	707	7170	1	729	178.482851	6.1	2.1	No	6	21	311493.48
8367	Restaurant 8367	Rural	Japanese	4.0	33	71.07	2003	24268	8	197	151.838065	5.9	7.5	Yes	5	12	534142.98

1.3.3 Total Number of Rows and Columns in dataset

[13]:
```
revenue_data.shape
```

[13]:
```
(8368, 17)
```

[14]:
```
print(f'Number of Rows in datasets = {revenue_data.shape[0]}')
print(f'Number of Columns in datasets = {revenue_data.shape[1]}')
```

```
Number of Rows in datasets = 8368
Number of Columns in datasets = 17
```

1.3.4 List of columns

[16]:
```
revenue_data.columns
```

[16]:
```
Index(['Name', 'Location', 'Cuisine', 'Rating', 'Seating Capacity',
       'Average Meal Price', 'Marketing Budget', 'Social Media Followers',
       'Chef Experience Years', 'Number of Reviews', 'Avg Review Length',
       'Ambience Score', 'Service Quality Score', 'Parking Availability',
       'Weekend Reservations', 'Weekday Reservations', 'Revenue'],
      dtype='object')
```

1.3.5 Information about the data

[18]:
```
revenue_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8368 entries, 0 to 8367
Data columns (total 17 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Name                    8368 non-null   object
 1   Location                8368 non-null   object
 2   Cuisine                 8368 non-null   object
 3   Rating                  8368 non-null   float64
 4   Seating Capacity        8368 non-null   int64
 5   Average Meal Price      8368 non-null   float64
 6   Marketing Budget        8368 non-null   int64
 7   Social Media Followers  8368 non-null   int64
 8   Chef Experience Years   8368 non-null   int64
 9   Number of Reviews       8368 non-null   int64
 10  Avg Review Length       8368 non-null   float64
 11  Ambience Score          8368 non-null   float64
 12  Service Quality Score   8368 non-null   float64
 13  Parking Availability    8368 non-null   object
 14  Weekend Reservations    8368 non-null   int64
 15  Weekday Reservations    8368 non-null   int64
 16  Revenue                 8368 non-null   float64
```

```
dtypes: float64(6), int64(7), object(4)
memory usage: 1.1+ MB
```

1.4 Data Cleaning and Transformation

1.4.1 Checking and removing duplicates

[21]:
```
revenue_data.duplicated().any()
```

[21]:
```
False
```

No duplicates in dataset

1.4.2 Checking Null Values

[24]:
```
revenue_data.isna().sum()
```

[24]:
```
Name                      0
Location                  0
Cuisine                   0
Rating                    0
Seating Capacity          0
Average Meal Price        0
Marketing Budget          0
Social Media Followers    0
Chef Experience Years     0
Number of Reviews         0
Avg Review Length         0
Ambience Score            0
Service Quality Score     0
Parking Availability      0
Weekend Reservations      0
Weekday Reservations      0
Revenue                   0
dtype: int64
```

No missing values in dataset

1.4.3 Deleting unwanted columns

[27]:
```
revenue_data1 = revenue_data.drop(columns='Name',axis=1)
revenue_data1.head(2)
```

[27]:
```
   Location   Cuisine  Rating  Seating Capacity  Average Meal Price  \
0     Rural  Japanese     4.0                38               73.98
1  Downtown   Mexican     3.2                76               28.11

   Marketing Budget  Social Media Followers  Chef Experience Years  \
0              2224                   23406                     13
```

```
1              4416                    42741                        8

   Number of Reviews  Avg Review Length  Ambience Score  \
0                185         161.924906             1.3
1                533         148.759717             2.6

   Service Quality Score Parking Availability  Weekend Reservations  \
0                    7.0                  Yes                    13
1                    3.4                  Yes                    48

   Weekday Reservations    Revenue
0                     4  638945.52
1                     6  490207.83
```

1.4.4 Renaming the columns

[29]:
```
revenue_data1 = revenue_data1.rename(columns={
    'Seating Capacity': 'Seating_Capacity',
    'Average Meal Price': 'Average_Meal_Price',
    'Marketing Budget': 'Marketing_Budget',
    'Social Media Followers': 'Social',
    'Chef Experience Years': 'Chef_exp',
    'Number of Reviews': 'Reviews',
        'Avg Review Length': 'Review_Length',
        'Ambience Score' : 'Ambience',
        'Service Quality Score': 'Service_Qual',
        'Weekend Reservations': 'Weekend_Reservations',
        'Weekday Reservations': 'Weekday_Reservations',
        'Parking Availability': 'Parking'})
```

[30]:
```
revenue_data1.head(2)
```

[30]:
```
   Location   Cuisine  Rating  Seating_Capacity  Average_Meal_Price  \
0     Rural  Japanese     4.0                38               73.98
1  Downtown   Mexican     3.2                76               28.11

   Marketing_Budget  Social  Chef_exp  Reviews  Review_Length  Ambience  \
0              2224   23406        13      185     161.924906       1.3
1              4416   42741         8      533     148.759717       2.6

   Service_Qual Parking  Weekend_Reservations  Weekday_Reservations    Revenue
0           7.0     Yes                    13                     4  638945.52
1           3.4     Yes                    48                     6  490207.83
```

1.5 Exploratory Data Analysis (EDA)

1.5.1 Statistics

Basic Statistics of Numeric Features (mean, median, max, min, etc...)

[34]:
```
revenue_data1.describe()
```

[34]:

	Rating	Seating_Capacity	Average_Meal_Price	Marketing_Budget	\
count	8368.000000	8368.000000	8368.000000	8368.000000	
mean	4.008258	60.212835	47.896659	3218.254900	
std	0.581474	17.399488	14.336767	1824.896053	
min	3.000000	30.000000	25.000000	604.000000	
25%	3.500000	45.000000	35.490000	1889.000000	
50%	4.000000	60.000000	45.535000	2846.500000	
75%	4.500000	75.000000	60.300000	4008.500000	
max	5.000000	90.000000	76.000000	9978.000000	

	Social	Chef_exp	Reviews	Review_Length	Ambience	\
count	8368.000000	8368.000000	8368.000000	8368.000000	8368.000000	
mean	36190.621773	10.051984	523.010397	174.769974	5.521283	
std	18630.153330	5.516606	277.215127	71.998060	2.575442	
min	5277.000000	1.000000	50.000000	50.011717	1.000000	
25%	22592.500000	5.000000	277.000000	113.311102	3.300000	
50%	32518.500000	10.000000	528.000000	173.910079	5.500000	
75%	44566.250000	15.000000	764.250000	237.406885	7.800000	
max	103777.000000	19.000000	999.000000	299.984924	10.000000	

	Service_Qual	Weekend_Reservations	Weekday_Reservations	Revenue
count	8368.000000	8368.000000	8368.000000	8.368000e+03
mean	5.508772	29.491754	29.235301	6.560706e+05
std	2.586552	20.025415	20.004277	2.674137e+05
min	1.000000	0.000000	0.000000	1.847085e+05
25%	3.200000	13.000000	13.000000	4.546514e+05
50%	5.600000	27.000000	26.000000	6.042421e+05
75%	7.800000	43.000000	43.000000	8.130942e+05
max	10.000000	88.000000	88.000000	1.531868e+06

[35]:
```
revenue_data1.describe(include='object')
```

[35]:

	Location	Cuisine	Parking
count	8368	8368	8368
unique	3	6	2
top	Downtown	French	Yes
freq	2821	1433	4189

Unique values in categorical features

[37]:
```
location_ls = revenue_data1['Location'].unique().tolist()
location_ls
```

[37]:
```
['Rural', 'Downtown', 'Suburban']
```

[38]:
```
cuisine_ls = revenue_data1['Cuisine'].unique().tolist()
cuisine_ls
```

[38]:
```
['Japanese', 'Mexican', 'Italian', 'Indian', 'French', 'American']
```

[39]:
```
parking_ls = revenue_data1['Parking'].unique().tolist()
parking_ls
```

[39]:
```
['Yes', 'No']
```

Correlation matrix

[41]:
```
Numerical_columns = revenue_data1.select_dtypes(include="number").columns
print(f"Numerical columns in the data : {Numerical_columns}")
```

```
Numerical columns in the data : Index(['Rating', 'Seating_Capacity',
'Average_Meal_Price', 'Marketing_Budget',
       'Social', 'Chef_exp', 'Reviews', 'Review_Length', 'Ambience',
       'Service_Qual', 'Weekend_Reservations', 'Weekday_Reservations',
       'Revenue'],
      dtype='object')
```

[42]:
```
Categorical_columns = revenue_data1.select_dtypes(exclude="number").columns
print(f"Categorical columns in the data : {Categorical_columns}")
```

```
Categorical columns in the data : Index(['Location', 'Cuisine', 'Parking'],
dtype='object')
```

Correlation matrix

[44]:
```
corr_data = revenue_data1[Numerical_columns].corr()
corr_data
```

[44]:

	Rating	Seating_Capacity	Average_Meal_Price	\
Rating	1.000000	0.004862	-0.002265	
Seating_Capacity	0.004862	1.000000	-0.028809	
Average_Meal_Price	-0.002265	-0.028809	1.000000	
Marketing_Budget	0.263448	0.509560	-0.002054	
Social	0.289559	0.496917	-0.005753	
Chef_exp	0.022192	0.010324	0.000401	
Reviews	-0.004238	-0.016100	0.001802	
Review_Length	-0.001272	-0.032670	0.016013	
Ambience	0.007099	0.002842	-0.001094	
Service_Qual	0.000887	-0.006400	0.011714	
Weekend_Reservations	-0.000274	0.437416	-0.018072	
Weekday_Reservations	0.001706	0.418636	-0.027837	
Revenue	0.009899	0.677317	0.686365	

	Marketing_Budget	Social	Chef_exp	Reviews
Rating	0.263448	0.289559	0.022192	-0.004238
Seating_Capacity	0.509560	0.496917	0.010324	-0.016100
Average_Meal_Price	-0.002054	-0.005753	0.000401	0.001802
Marketing_Budget	1.000000	0.987511	0.021612	-0.008985
Social	0.987511	1.000000	0.022943	-0.009181
Chef_exp	0.021612	0.022943	1.000000	-0.007710
Reviews	-0.008985	-0.009181	-0.007710	1.000000
Review_Length	-0.030064	-0.025464	0.001865	0.005842
Ambience	0.008120	0.007428	-0.001288	0.010186
Service_Qual	-0.000693	0.000210	-0.008508	0.005432
Weekend_Reservations	0.225355	0.218451	-0.000372	0.000780
Weekday_Reservations	0.221088	0.214612	-0.008823	-0.006126
Revenue	0.365322	0.354466	0.026899	-0.008233

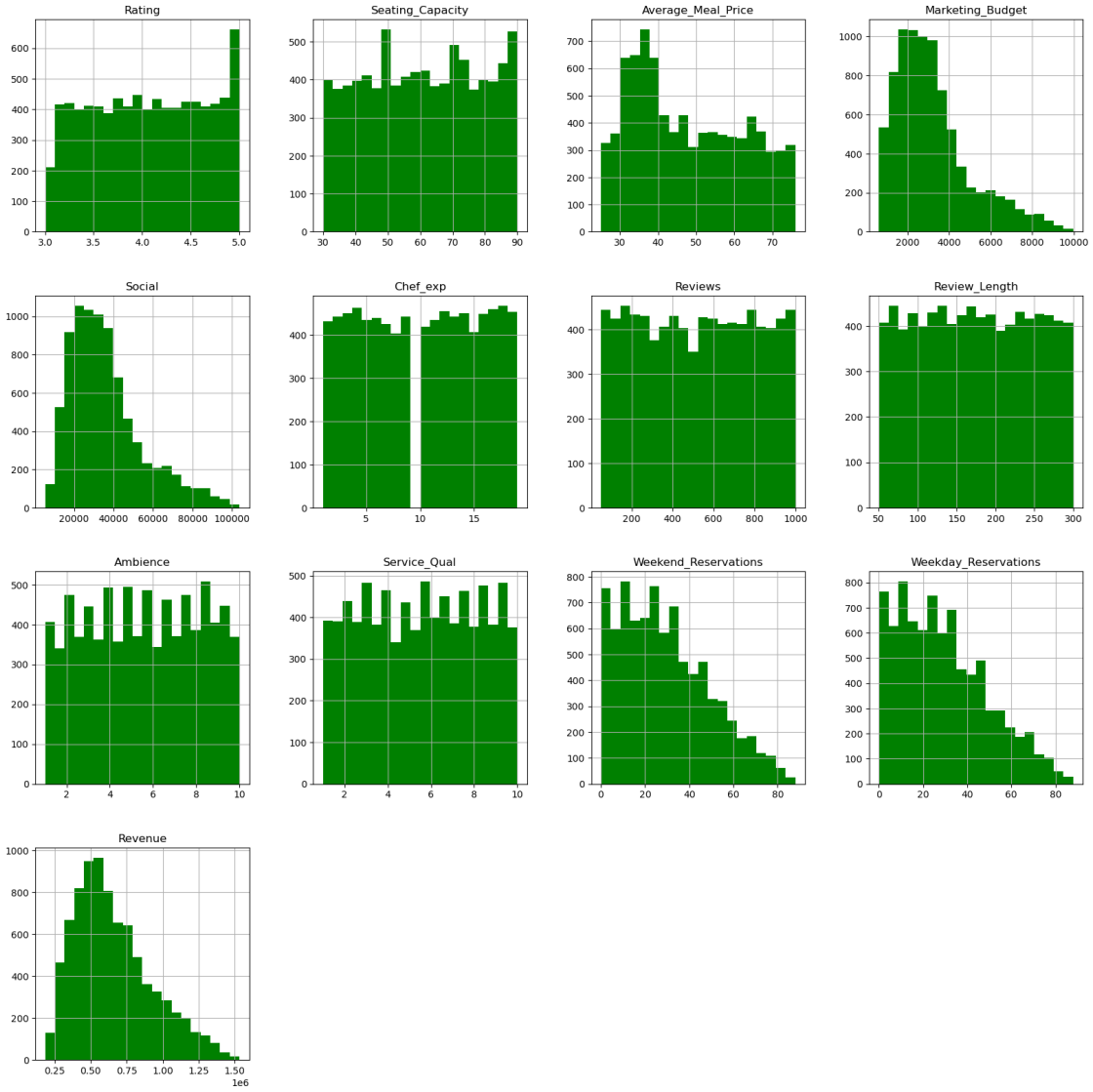
	Review_Length	Ambience	Service_Qual
Rating	-0.001272	0.007099	0.000887
Seating_Capacity	-0.032670	0.002842	-0.006400
Average_Meal_Price	0.016013	-0.001094	0.011714
Marketing_Budget	-0.030064	0.008120	-0.000693
Social	-0.025464	0.007428	0.000210
Chef_exp	0.001865	-0.001288	-0.008508
Reviews	0.005842	0.010186	0.005432
Review_Length	1.000000	0.004807	0.008836
Ambience	0.004807	1.000000	0.000612
Service_Qual	0.008836	0.000612	1.000000
Weekend_Reservations	-0.014241	-0.012207	-0.002689
Weekday_Reservations	-0.011020	-0.016652	0.004204
Revenue	-0.011278	0.003388	0.005375

	Weekend_Reservations	Weekday_Reservations	Revenue
Rating	-0.000274	0.001706	0.009899
Seating_Capacity	0.437416	0.418636	0.677317
Average_Meal_Price	-0.018072	-0.027837	0.686365
Marketing_Budget	0.225355	0.221088	0.365322
Social	0.218451	0.214612	0.354466
Chef_exp	-0.000372	-0.008823	0.026899
Reviews	0.000780	-0.006126	-0.008233
Review_Length	-0.014241	-0.011020	-0.011278
Ambience	-0.012207	-0.016652	0.003388
Service_Qual	-0.002689	0.004204	0.005375
Weekend_Reservations	1.000000	0.179730	0.292400
Weekday_Reservations	0.179730	1.000000	0.273284
Revenue	0.292400	0.273284	1.000000

1.5.2 Data Visualization

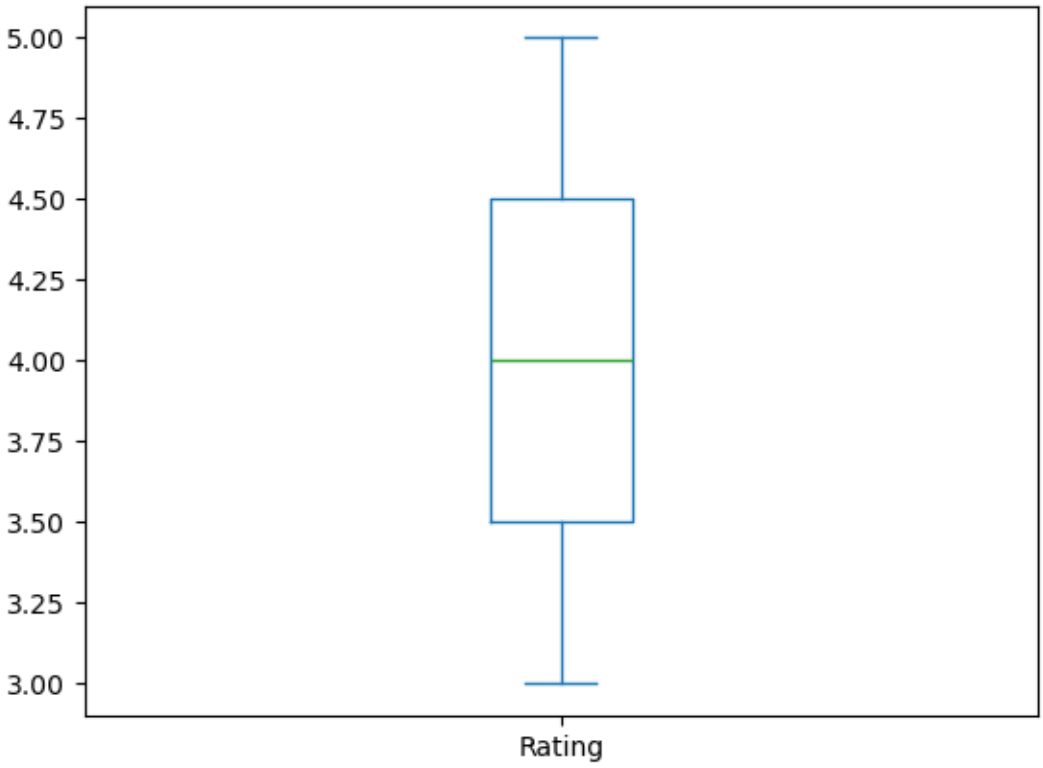
Histogram

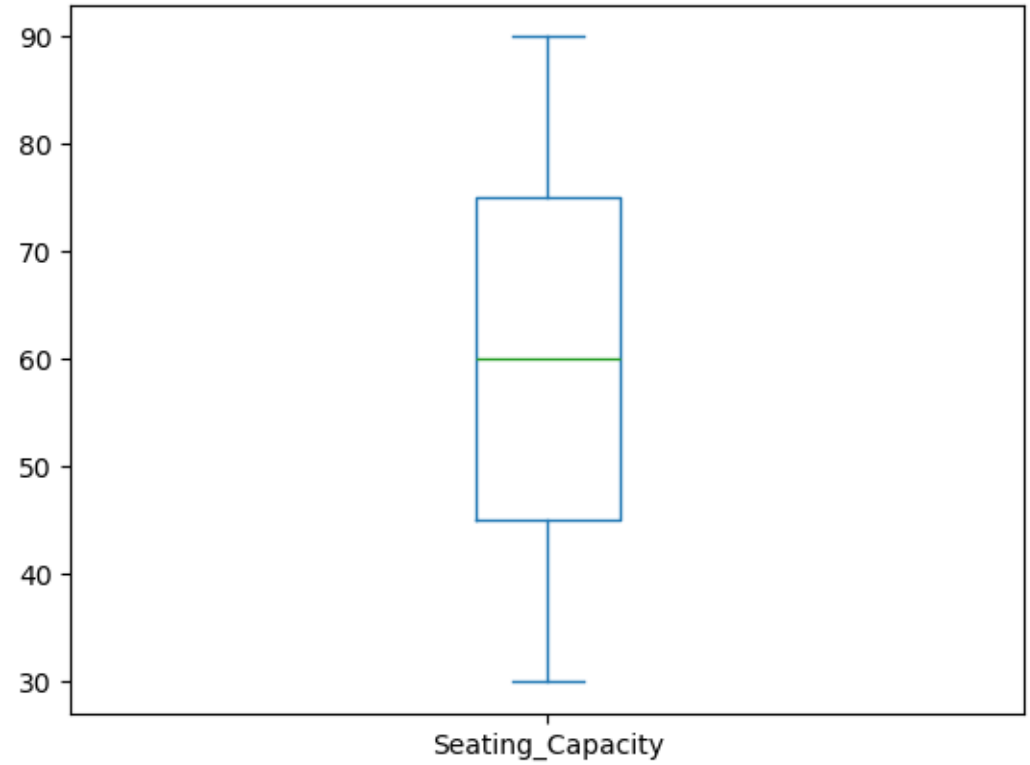
[47]:
```
revenue_data1.hist(figsize=(20,20), color = 'green',bins=20)
plt.show()
```

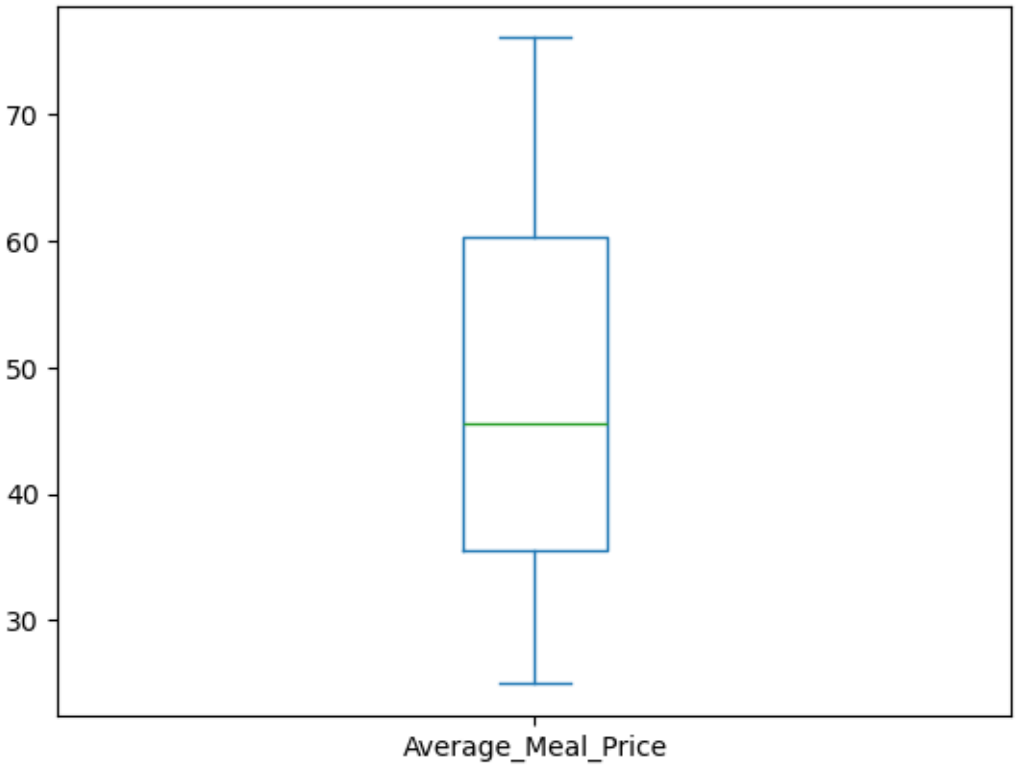


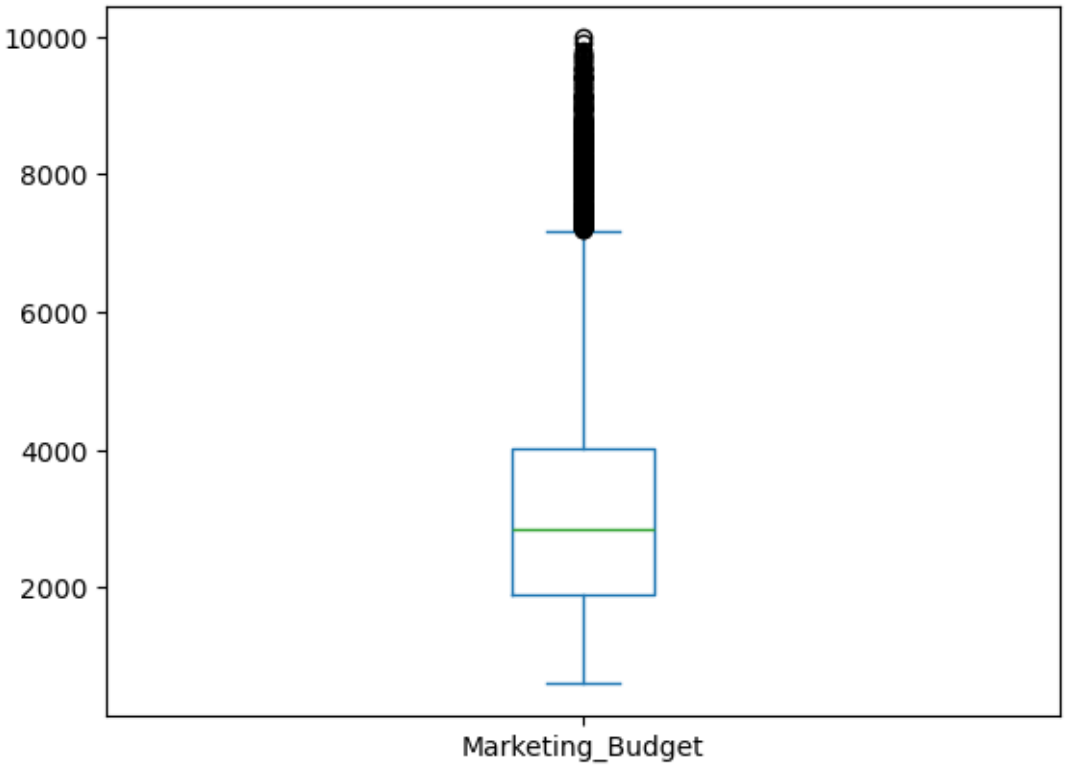
Box Plot

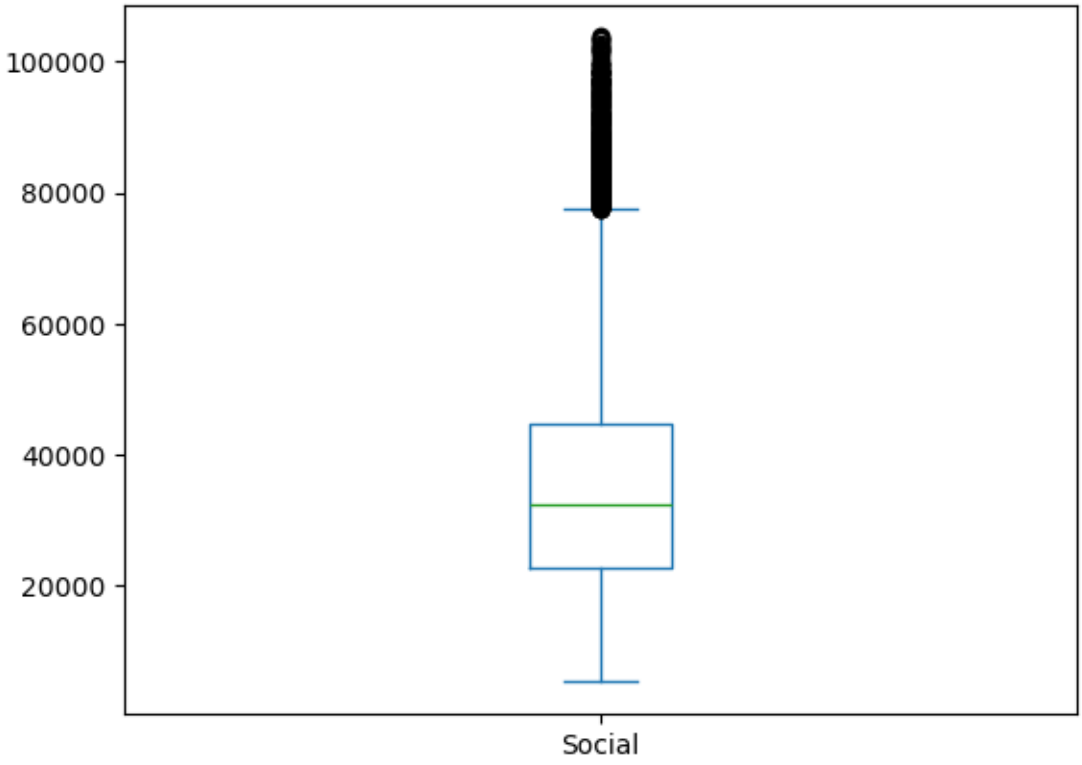
[49]:
```
for col in Numerical_columns:
    revenue_data1[col].plot(kind = 'box')
    plt.show()
```

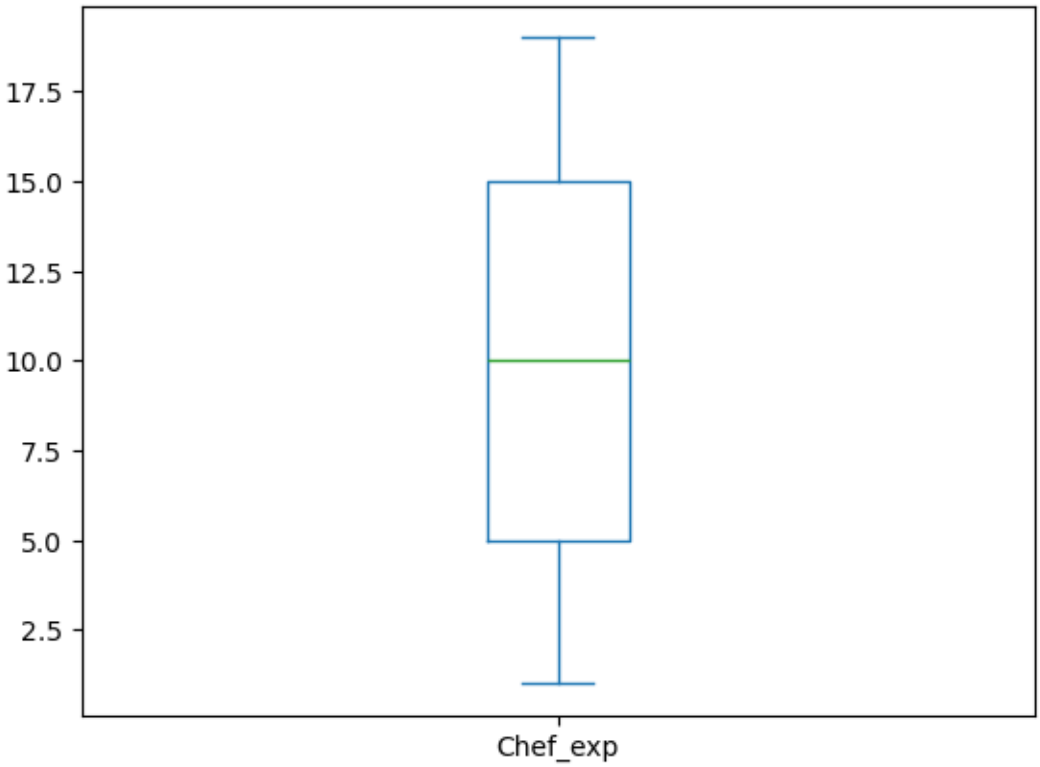

5.00
4.75
4.50
4.25
4.00
3.75
3.50
3.25
3.00
Rating

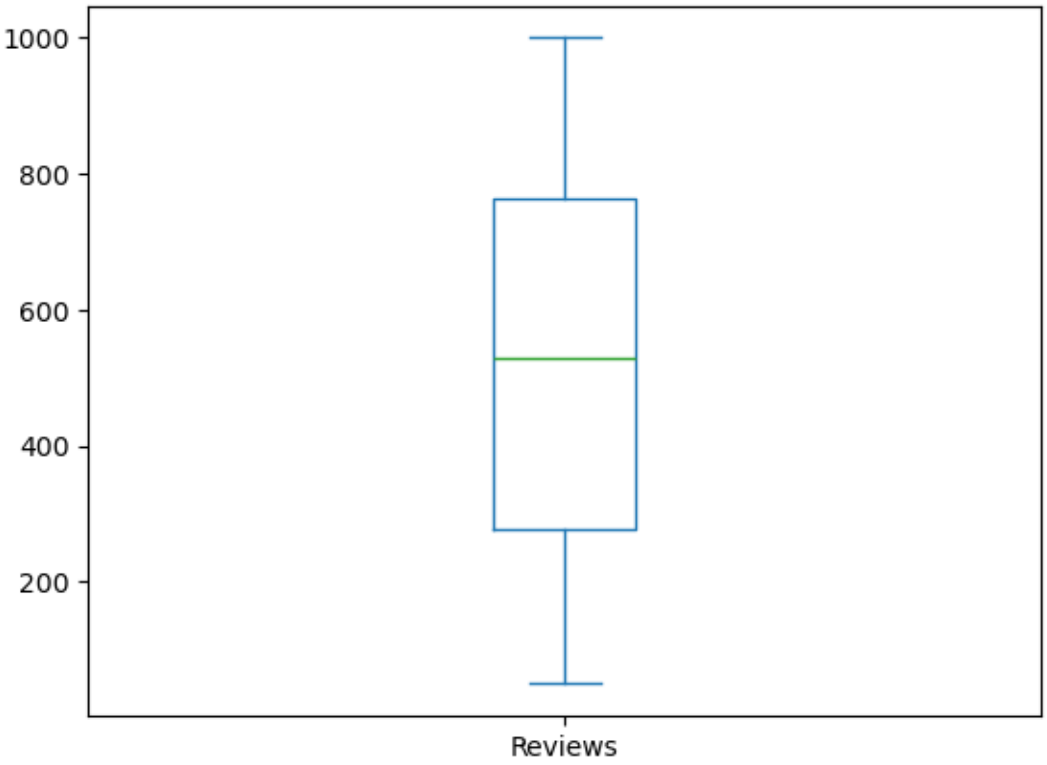

90
80
70
60
50
40
30
Seating_Capacity

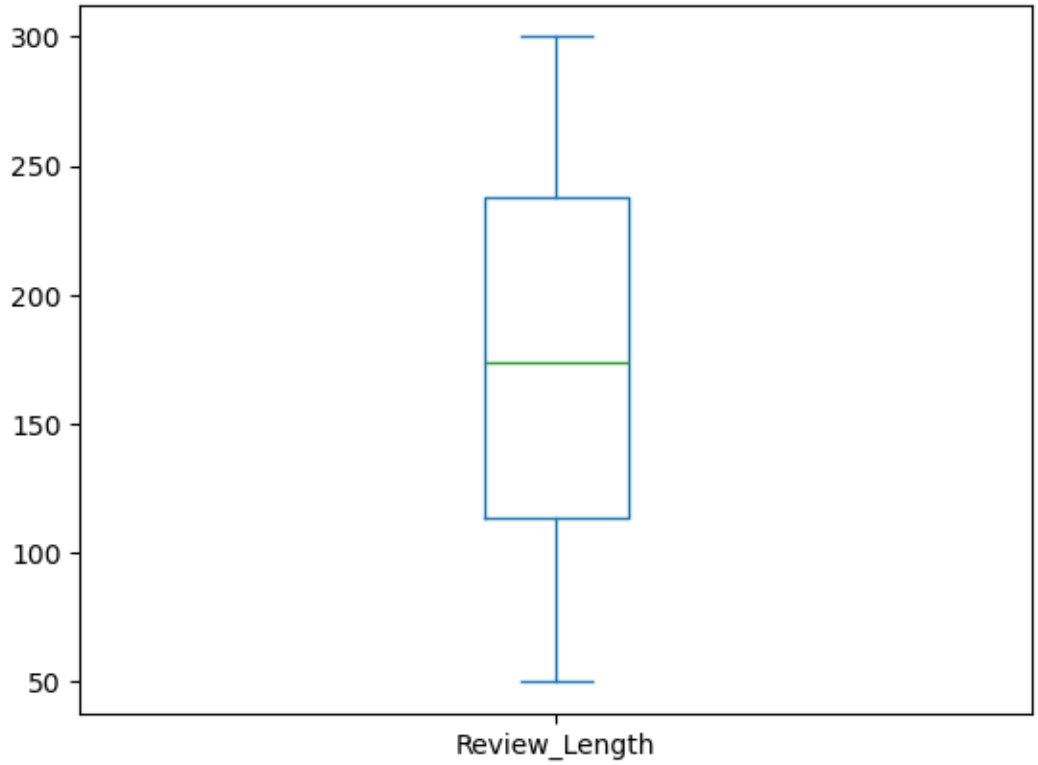

70
60
50
40
30
Average_Meal_Price

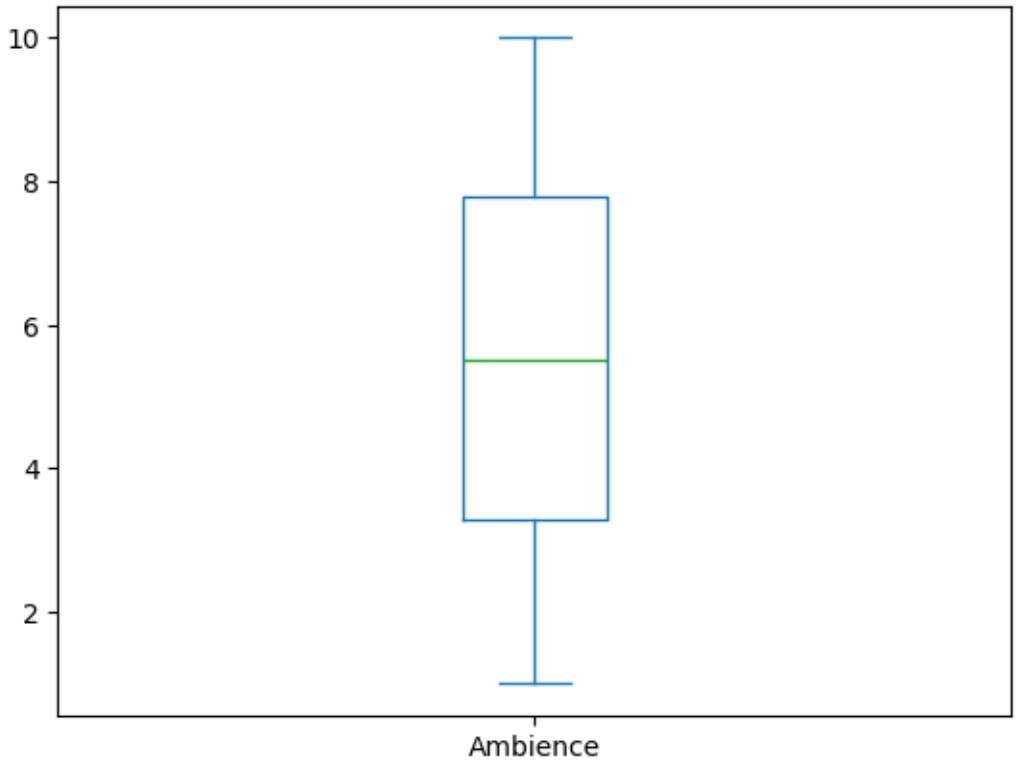

10000
8000
6000
4000
2000
Marketing_Budget

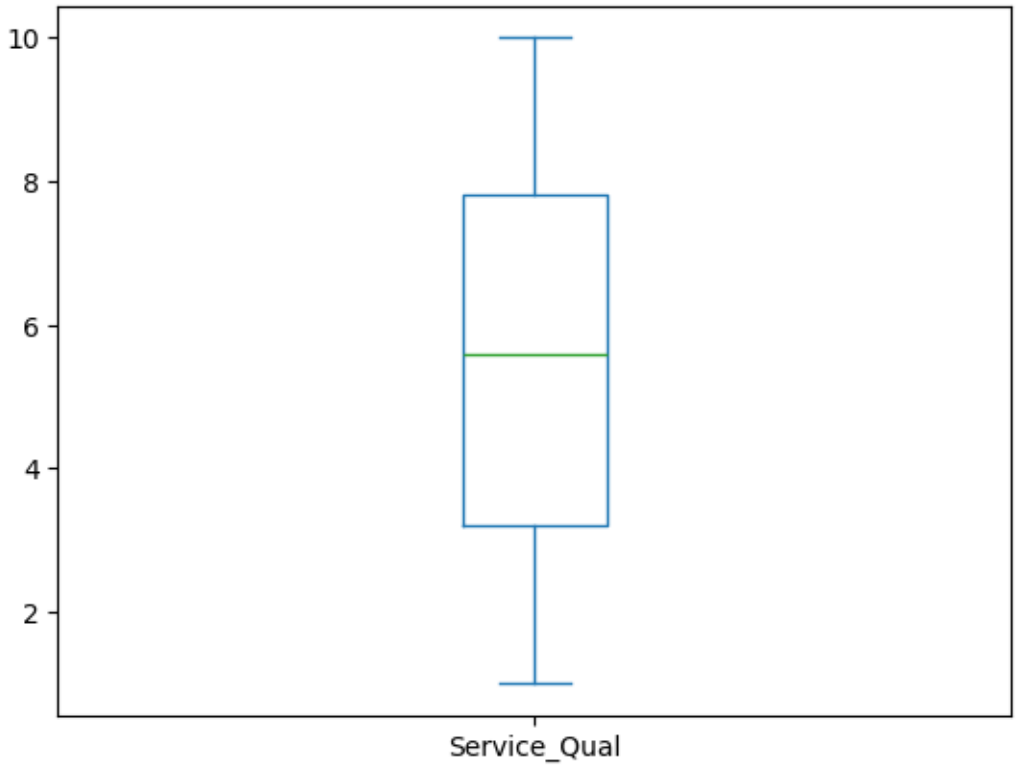

100000
80000
60000
40000
20000
Social

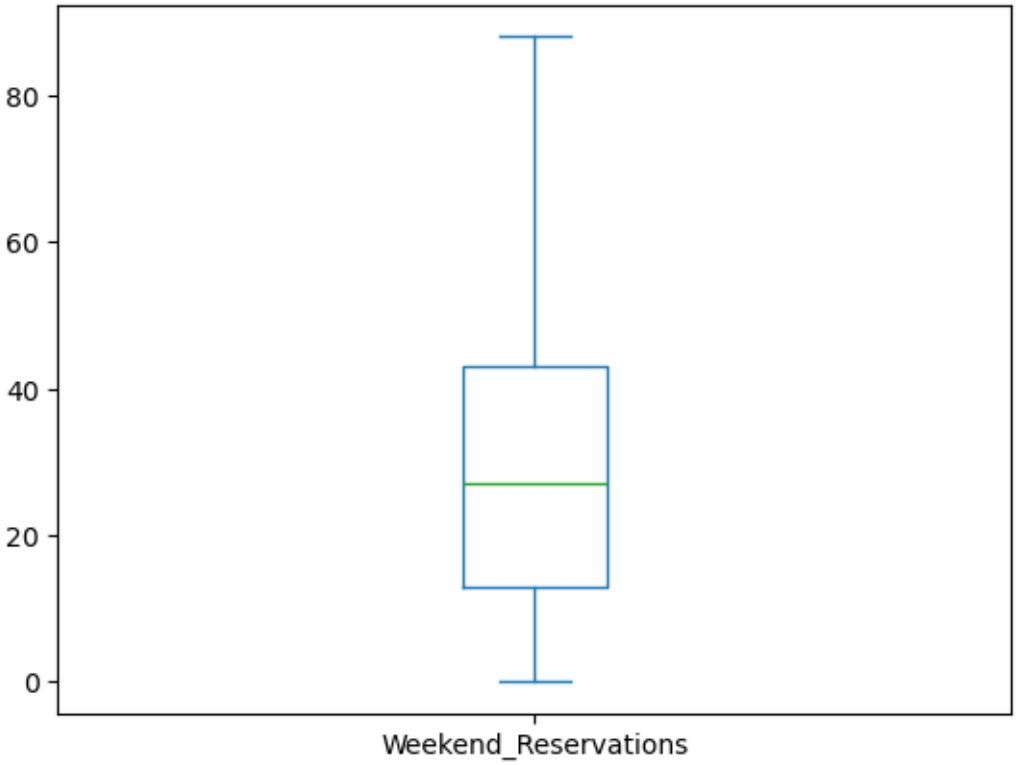

17.5
15.0
12.5
10.0
7.5
5.0
2.5
Chef_exp

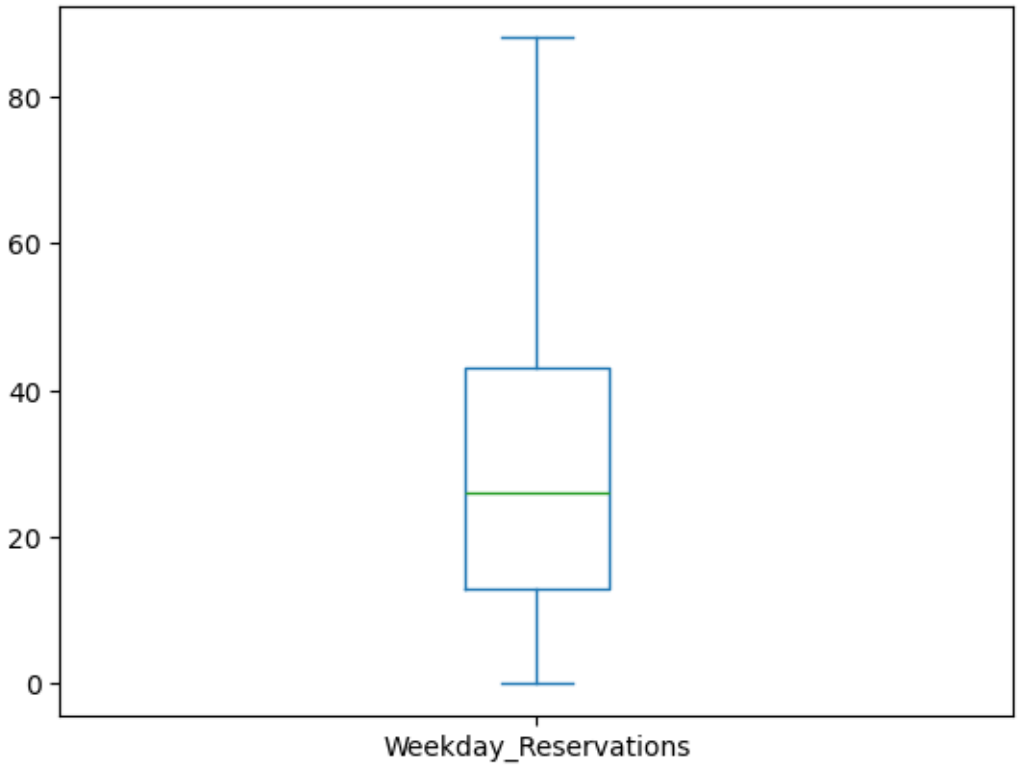

1000
800
600
400
200
Reviews

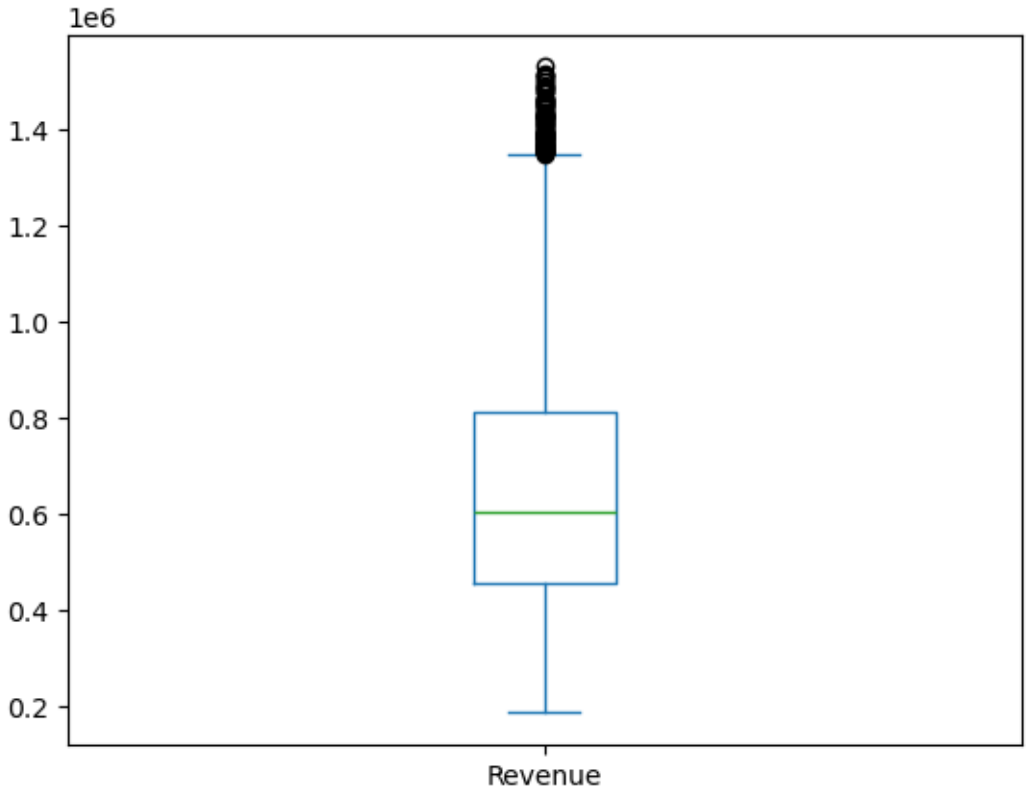

300
250
200
150
100
50
Review_Length


10
8
6
4
2
Ambience


10
8
6
4
2
Service_Qual

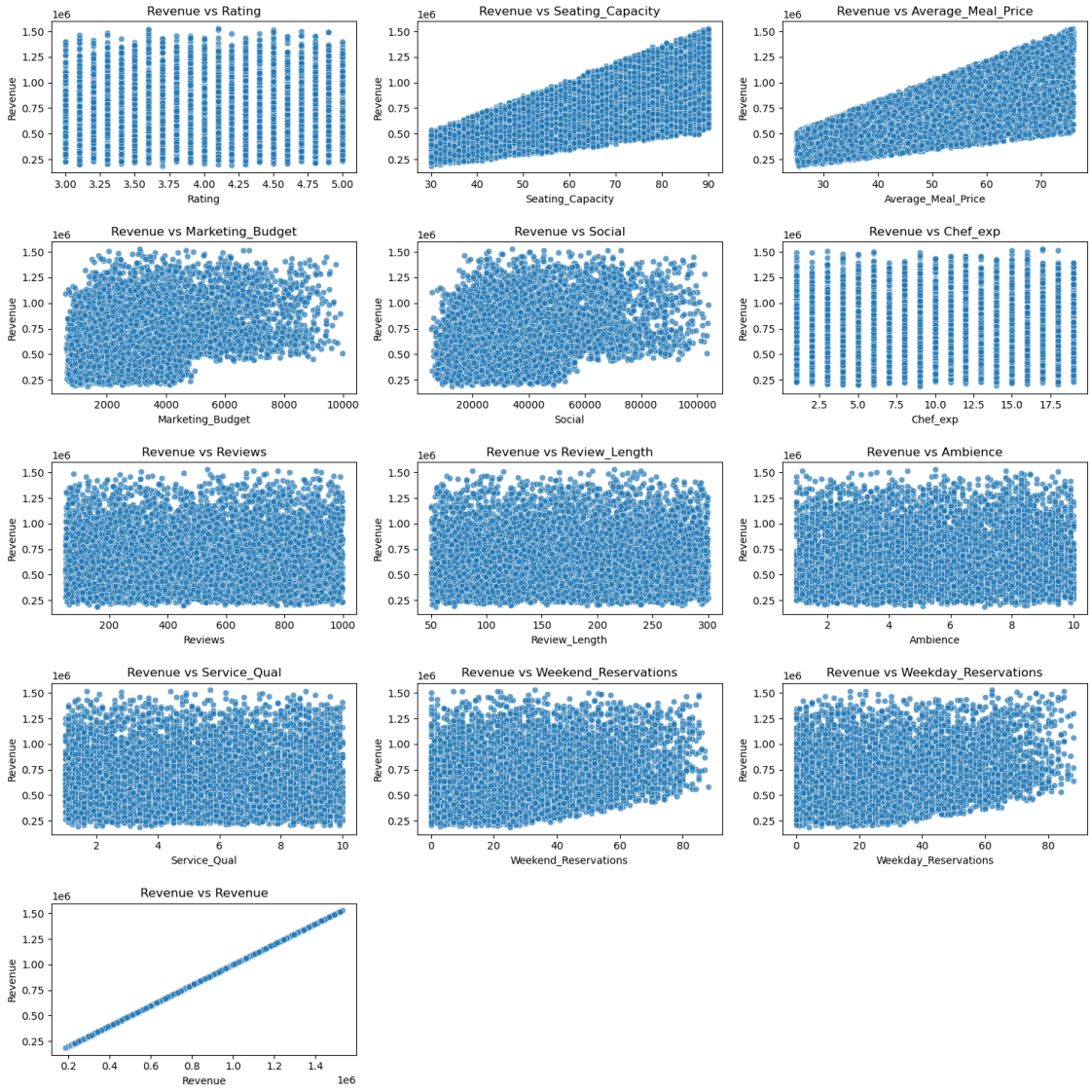

80
60
40
20
0
Weekend_Reservations


80
60
40
20
0
Weekday_Reservations

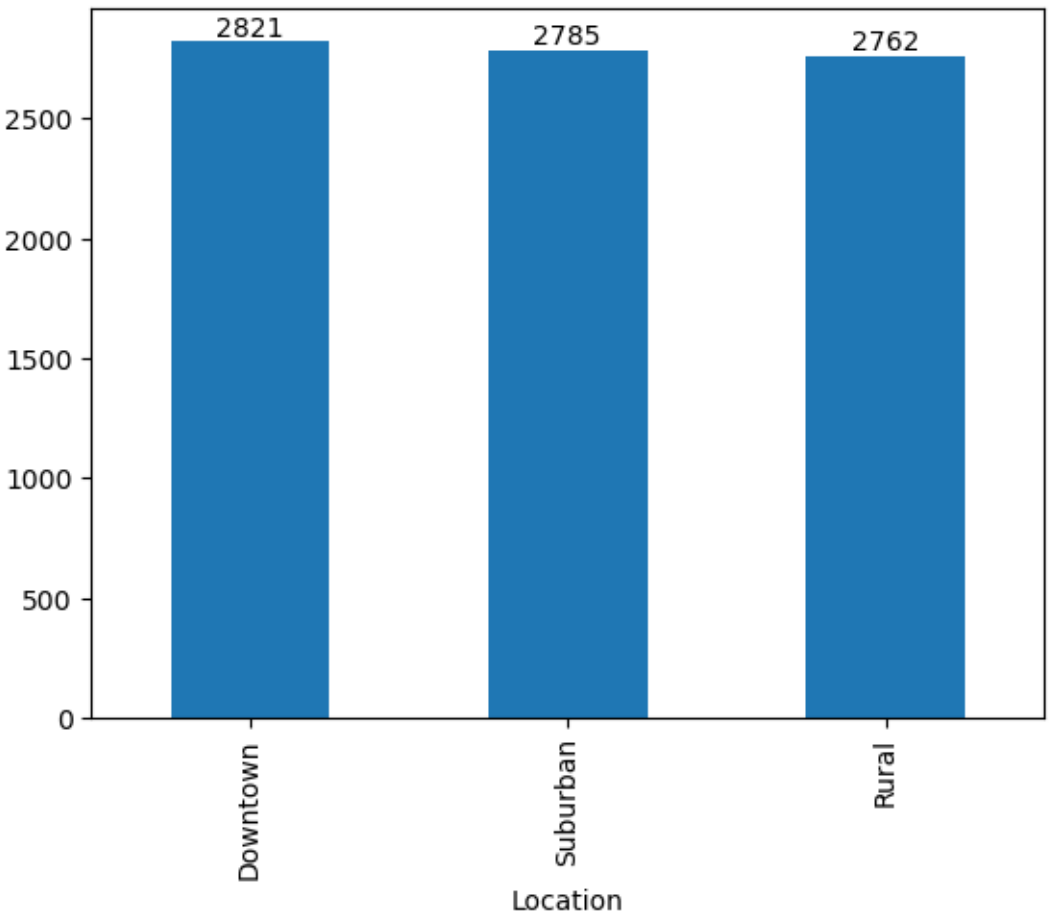


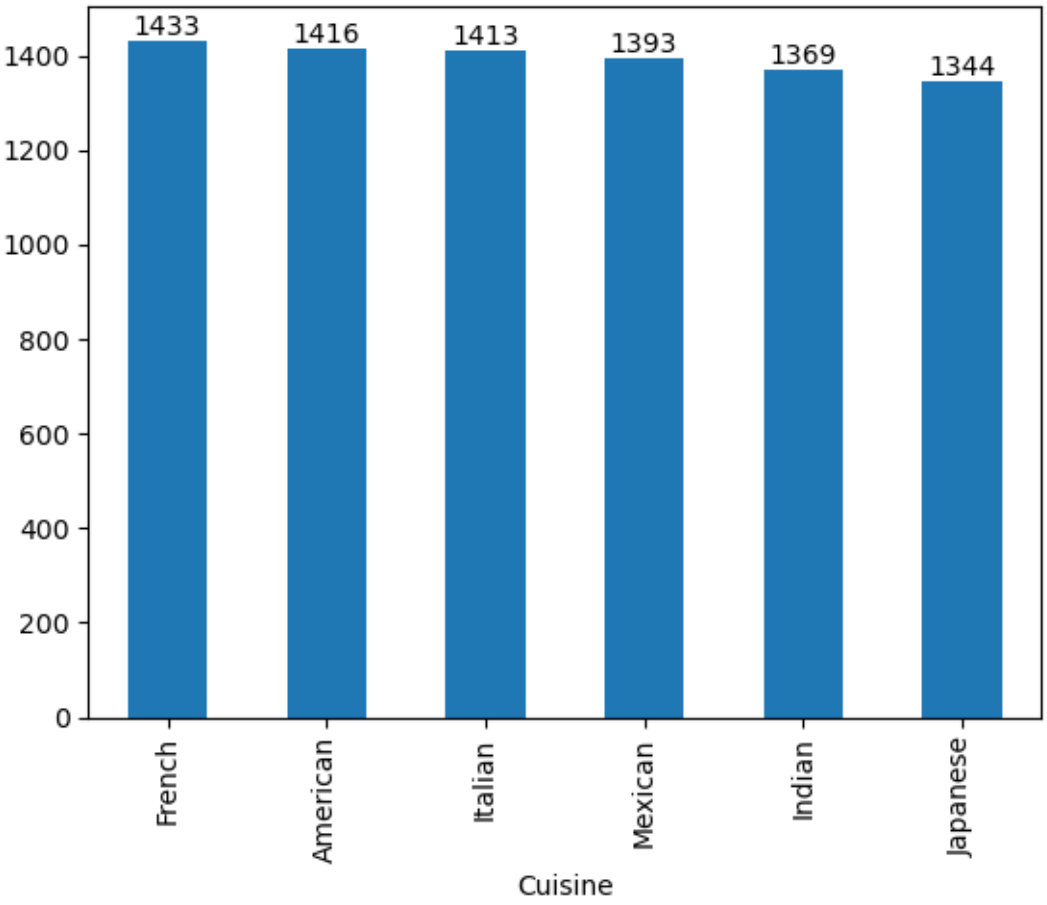
Scatter Plot

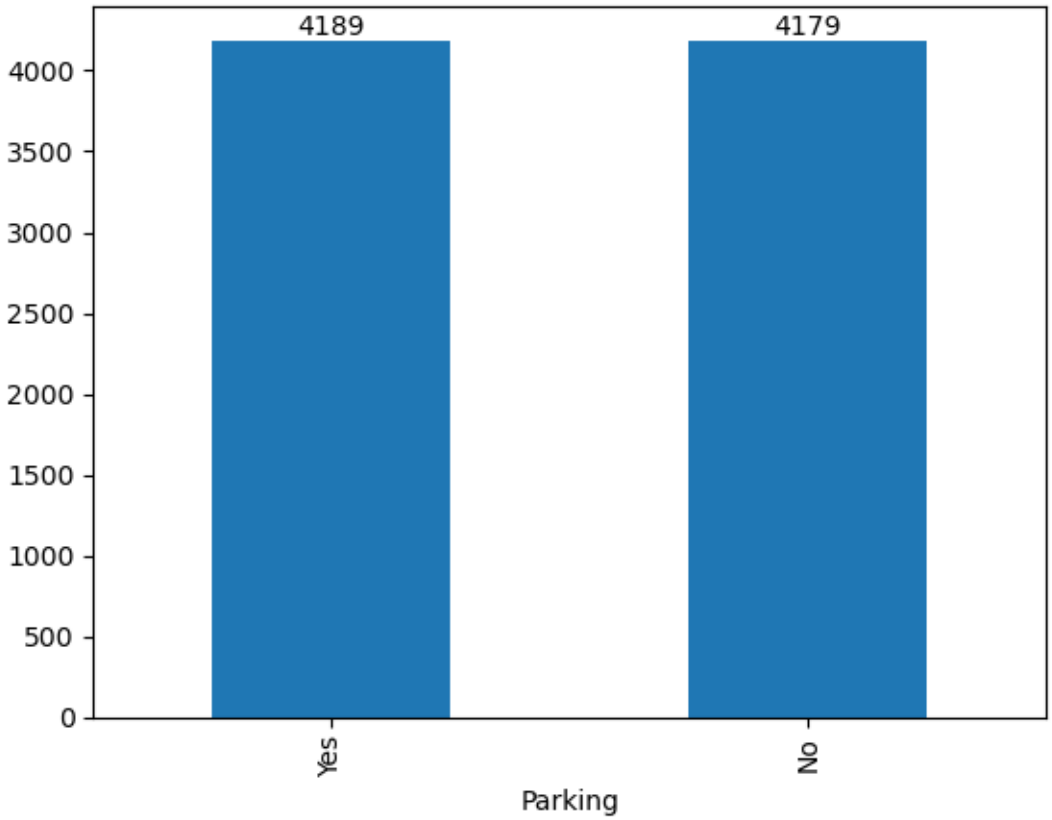
[51]:
```
plt.figure(figsize=(15,15))
for i, feature in enumerate(Numerical_columns, 1):
    plt.subplot(5, 3, i)
    sns.scatterplot(data=revenue_data1, x=feature, y='Revenue', alpha=0.7)
    plt.title(f'Revenue vs {feature}')
    plt.xlabel(feature)
    plt.ylabel('Revenue')

plt.tight_layout()
plt.show()
```



[52]:
```
for col in Categorical_columns:
    ax = revenue_data1[col].value_counts().plot(kind = 'bar')
    for i in ax.containers:
        ax.bar_label(i)
    plt.show()
```

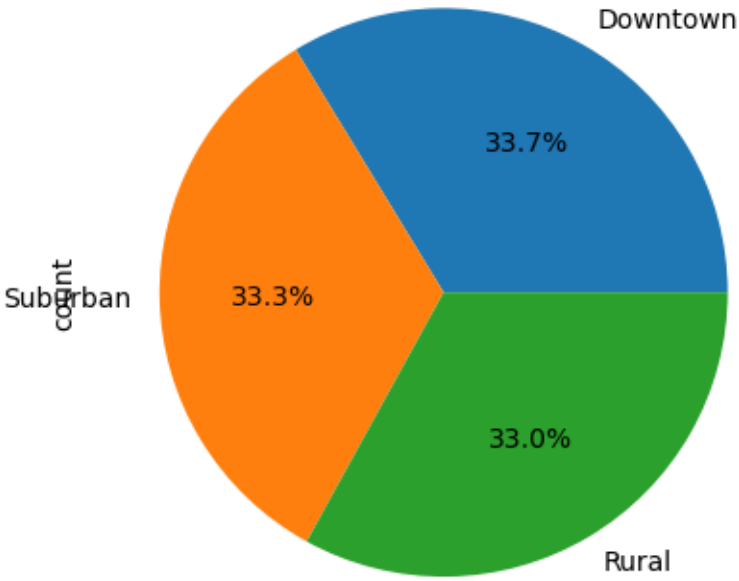

2821
2785
2762
2500
2000
1500
1000
500
0
Downtown
Suburban
Rural
Location


1433
1416
1413
1393
1369
1344
1400
1200
1000
800
600
400
200
0
French
American
Italian
Mexican
Indian
Japanese
Cuisine

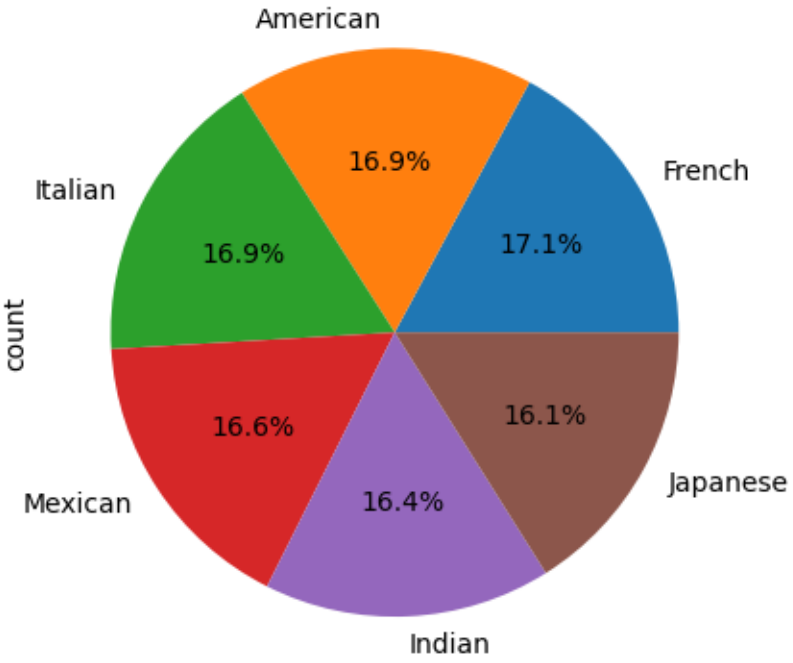


Pie chart

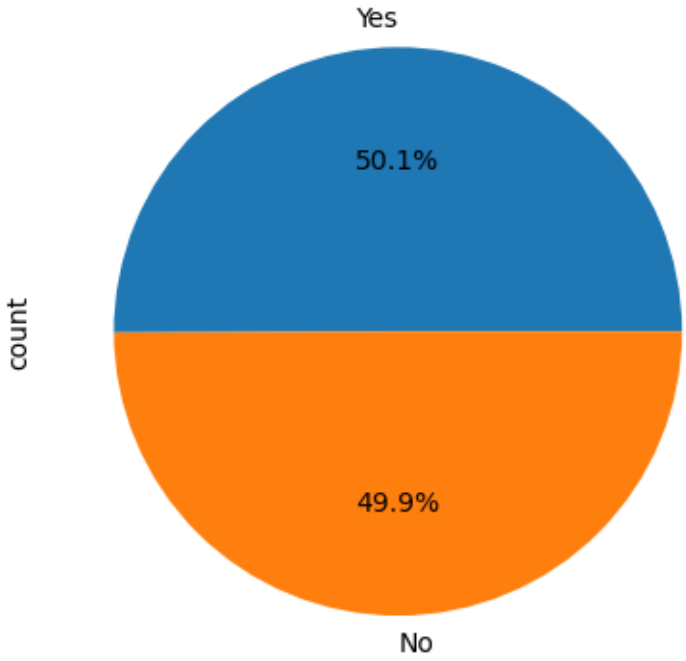
[54]:
```
revenue_data1['Location'].value_counts().plot(kind = 'pie', autopct = '%1.1f%%')
plt.show()
```



[55]:
```
revenue_data1['Cuisine'].value_counts().plot(kind = 'pie', autopct = '%1.1f%%')
plt.show()
```

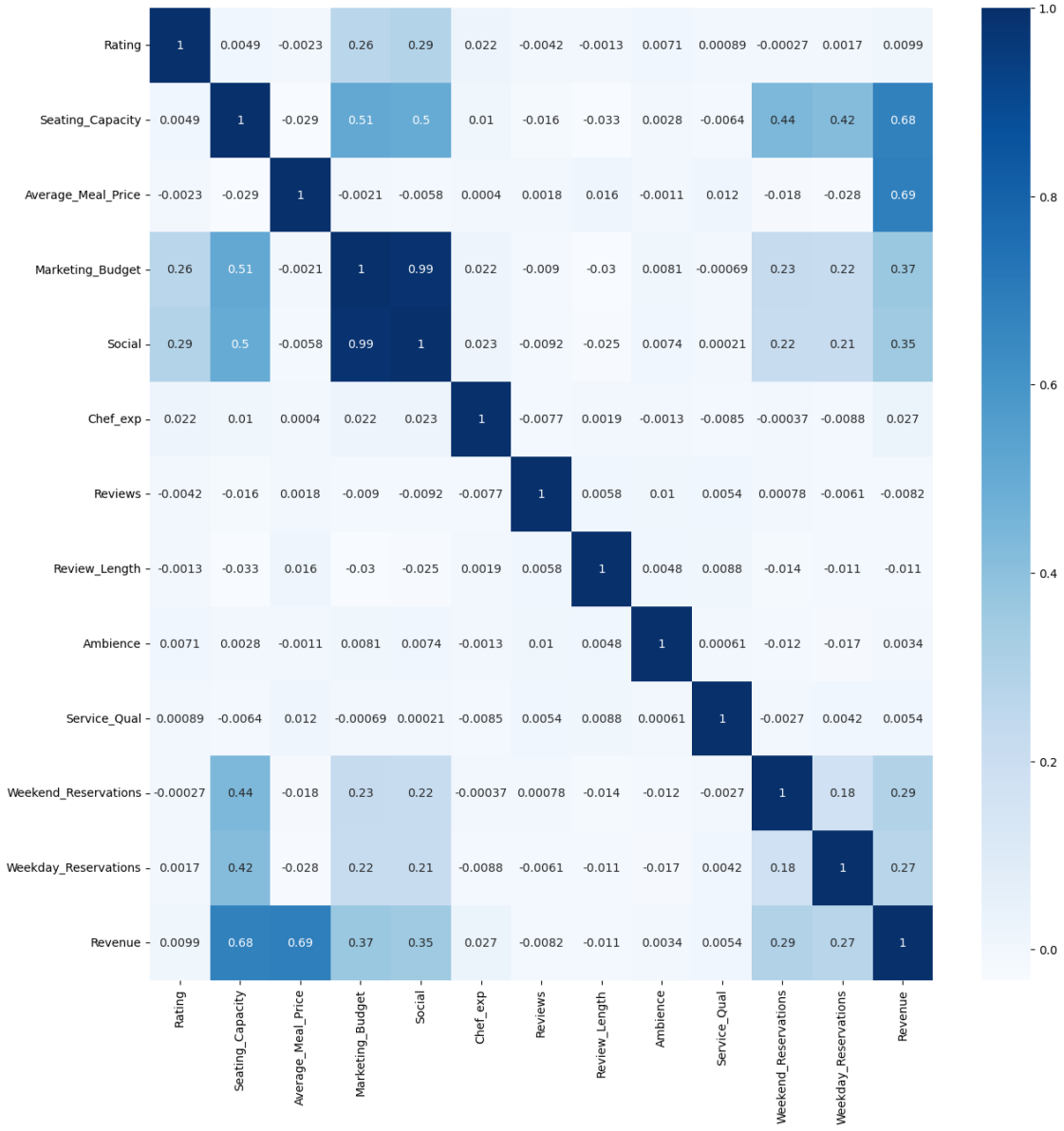


[56]:
```
revenue_data1['Parking'].value_counts().plot(kind = 'pie', autopct = '%1.1f%%')
plt.show()
```



Heatmap

[58]:
```
plt.figure(figsize=(15,15))
sns.heatmap(corr_data, annot=True, cmap="Blues")
plt.show()
```



1.6 Feature Engineering / Data preprocessing

[60]:
```
num_col = Numerical_columns.drop('Revenue')
num_col
```

[60]:
```
Index(['Rating', 'Seating_Capacity', 'Average_Meal_Price', 'Marketing_Budget',
       'Social', 'Chef_exp', 'Reviews', 'Review_Length', 'Ambience',
       'Service_Qual', 'Weekend_Reservations', 'Weekday_Reservations'],
      dtype='object')
```

1.6.1 Seperating the Target and independent variables

[62]:
```
x = revenue_data1.drop('Revenue',axis=1)
x.head()
```

[62]:

	Location	Cuisine	Rating	Seating_Capacity	Average_Meal_Price	\
0	Rural	Japanese	4.0	38	73.98	
1	Downtown	Mexican	3.2	76	28.11	
2	Rural	Italian	4.7	48	48.29	
3	Rural	Italian	4.4	34	51.55	
4	Downtown	Japanese	4.9	88	75.98	

	Marketing_Budget	Social	Chef_exp	Reviews	Review_Length	Ambience	\
0	2224	23406	13	185	161.924906	1.3	
1	4416	42741	8	533	148.759717	2.6	
2	2796	37285	18	853	56.849189	5.3	
3	1167	15214	13	82	205.433265	4.6	
4	3639	40171	9	78	241.681584	8.6	

	Service_Qual	Parking	Weekend_Reservations	Weekday_Reservations
0	7.0	Yes	13	4
1	3.4	Yes	48	6
2	6.7	No	27	14
3	2.8	Yes	9	17
4	2.1	No	37	26

[63]:
```
y = revenue_data['Revenue']
y
```

[63]:
```
0        638945.52
1        490207.83
2        541368.62
3        404556.80
4       1491046.35
           ...
8363     434653.45
8364     414977.92
8365     930395.87
8366     311493.48
8367     534142.98
Name: Revenue, Length: 8368, dtype: float64
```

1.6.2 Label Encoding

[65]:
```
Categorical_columns
```

[65]:
```
Index(['Location', 'Cuisine', 'Parking'], dtype='object')
```

Using one hot encoding for categorical columns

[67]:
```
x = pd.get_dummies(x, columns=Categorical_columns)
x = x.astype(int)
x.head()
```

[67]:
```
   Rating  Seating_Capacity  Average_Meal_Price  Marketing_Budget  Social  \
0       4                38                  73              2224   23406
1       3                76                  28              4416   42741
2       4                48                  48              2796   37285
3       4                34                  51              1167   15214
4       4                88                  75              3639   40171

   Chef_exp  Reviews  Review_Length  Ambience  Service_Qual  …  \
0        13      185            161         1             7  …
1         8      533            148         2             3  …
2        18      853             56         5             6  …
3        13       82            205         4             2  …
4         9       78            241         8             2  …

   Location_Rural  Location_Suburban  Cuisine_American  Cuisine_French  \
0               1                  0                 0               0
1               0                  0                 0               0
2               1                  0                 0               0
3               1                  0                 0               0
4               0                  0                 0               0

   Cuisine_Indian  Cuisine_Italian  Cuisine_Japanese  Cuisine_Mexican  \
0               0                0                 1                0
1               0                0                 0                1
2               0                1                 0                0
3               0                1                 0                0
4               0                0                 1                0

   Parking_No  Parking_Yes
0           0            1
1           0            1
2           1            0
3           0            1
4           1            0

[5 rows x 23 columns]
```

[68]:
```
x.columns
```

[68]:
```
Index(['Rating', 'Seating_Capacity', 'Average_Meal_Price', 'Marketing_Budget',
       'Social', 'Chef_exp', 'Reviews', 'Review_Length', 'Ambience',
       'Service_Qual', 'Weekend_Reservations', 'Weekday_Reservations',
```

```
       'Location_Downtown', 'Location_Rural', 'Location_Suburban',
       'Cuisine_American', 'Cuisine_French', 'Cuisine_Indian',
       'Cuisine_Italian', 'Cuisine_Japanese', 'Cuisine_Mexican', 'Parking_No',
       'Parking_Yes'],
      dtype='object')
```

[69]:
```
x.shape
```

[69]:
```
(8368, 23)
```

1.6.3 Splitting Testing and Training Data

[71]:
```
from sklearn.model_selection import train_test_split
```

[72]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
↪random_state=42)
```

[73]:
```
x_train.shape, x_test.shape
```

[73]:
```
((6694, 23), (1674, 23))
```

1.6.4 Feature Scaling

[75]:
```
from sklearn.preprocessing import StandardScaler
```

[76]:
```
scaler = StandardScaler()
```

[77]:
```
x_train[num_col]
```

[77]:

	Rating	Seating_Capacity	Average_Meal_Price	Marketing_Budget	Social	\
7356	3	79	58	2758	28697	
8307	3	50	42	2551	25307	
208	4	86	53	7360	73821	
5303	4	87	74	3891	45053	
747	5	57	69	2952	33049	
…	…	…	…	…	…	
5734	3	64	45	3898	39153	
5191	4	75	64	3612	38504	
5390	4	82	40	4317	47772	
860	3	50	35	1958	23722	
7270	3	74	47	2122	27241	

	Chef_exp	Reviews	Review_Length	Ambience	Service_Qual	\
7356	13	66	122	3	1	
8307	18	735	109	1	5	
208	8	579	59	1	8	
5303	4	898	175	7	1	

```
747          13       981             141          3              8
…         …       …              …         …            …
5734          5       797             208          6              8
5191         15       574             120          2              3
5390         18       834             231          4              4
860           5       587             118          9              8
7270          6       963             293         10              5

      Weekend_Reservations  Weekday_Reservations
7356                    24                     2
8307                    41                    46
208                     84                    10
5303                     2                    54
747                     50                    35
…                      …                     …
5734                    14                    11
5191                    18                     4
5390                    62                    49
860                      7                    16
7270                    62                    37

[6694 rows x 12 columns]
```

[78]:
```
x_train[num_col] = scaler.fit_transform(x_train[num_col])
x_train
```

[78]:
```
         Rating  Seating_Capacity  Average_Meal_Price  Marketing_Budget  \
7356 -1.005580          1.074555            0.749255         -0.256601
8307 -1.005580         -0.596350           -0.373255         -0.369612
208   0.810269          1.477877            0.398471          2.255852
5303  0.810269          1.535495            1.871766          0.361958
747   2.626119         -0.193028            1.520981         -0.150687
…         …                …                   …                …
5734 -1.005580          0.210294           -0.162784          0.365780
5191  0.810269          0.844086            1.170197          0.209639
5390  0.810269          1.247407           -0.513569          0.594532
860  -1.005580         -0.596350           -0.864353         -0.693359
7270 -1.005580          0.786468           -0.022470         -0.603824

         Social  Chef_exp   Reviews  Review_Length  Ambience  Service_Qual  \
7356 -0.406898  0.534265 -1.655499      -0.719870 -0.811092     -1.568594
8307 -0.588351  1.438119  0.765152      -0.900657 -1.592646     -0.025246
208   2.008412 -0.369589  0.200695      -1.595994 -1.592646      1.132265
5303  0.468574 -1.092672  1.354937       0.017187  0.752014     -1.568594
747  -0.173952  0.534265  1.655256      -0.455642 -0.811092      1.132265
…         …        …        …              …        …            …
5734  0.152771 -0.911902  0.989487       0.476109  0.361238      1.132265
```

```
5191  0.118032  0.895807  0.182604      -0.747683 -1.201869     -0.796920
5390  0.614112  1.438119  1.123365       0.795964 -0.420316     -0.411083
860  -0.673190 -0.911902  0.229642      -0.775497  1.533568      1.132265
7270 -0.484832 -0.731131  1.590127       1.658182  1.924345     -0.025246

      …  Location_Rural  Location_Suburban  Cuisine_American  \
7356  …               0                  0                 0
8307  …               0                  1                 1
208   …               0                  0                 0
5303  …               0                  0                 0
747   …               0                  1                 0
…    …              …                  …                 …
5734  …               0                  1                 1
5191  …               0                  0                 0
5390  …               0                  0                 1
860   …               0                  1                 0
7270  …               0                  0                 0

      Cuisine_French  Cuisine_Indian  Cuisine_Italian  Cuisine_Japanese  \
7356               1               0                0                 0
8307               0               0                0                 0
208                0               0                1                 0
5303               0               0                0                 1
747                0               0                0                 1
…                …               …                …                 …
5734               0               0                0                 0
5191               1               0                0                 0
5390               0               0                0                 0
860                0               0                0                 0
7270               0               0                1                 0

      Cuisine_Mexican  Parking_No  Parking_Yes
7356                0           1            0
8307                0           1            0
208                 0           0            1
5303                0           1            0
747                 0           1            0
…                 …           …            …
5734                0           1            0
5191                0           0            1
5390                0           1            0
860                 1           1            0
7270                0           0            1

[6694 rows x 23 columns]
```

[79]:
```
x_test[num_col] = scaler.transform(x_test[num_col])
x_test
```

[79]:

```
        Rating  Seating_Capacity  Average_Meal_Price  Marketing_Budget  \
2412 -1.005580         -0.481115           -0.513569         -0.694451
6832  0.810269         -0.999672           -1.285294         -0.699911
5154  0.810269         -1.287759            0.258157         -1.049317
7081 -1.005580          0.095059            1.450824         -0.814014
6601  0.810269          0.555999           -1.004667          0.323742
…          …                 …                   …                …
1971 -1.005580          0.037442           -1.074824         -0.735943
2348 -1.005580         -0.596350            1.661295         -1.092993
7882 -1.005580          1.132173           -0.232941          1.570142
3898 -1.005580          1.016938            1.310511         -0.040951
664   0.810269         -0.250645            1.941923         -0.092816

        Social  Chef_exp   Reviews  Review_Length  Ambience  Service_Qual  \
2412 -0.494306 -1.634985  0.703641       0.253602 -0.029539      0.360591
6832 -0.559233  0.895807  1.257242       1.004565 -1.592646      1.132265
5154 -1.109267  0.895807 -0.892035       1.282700  1.142791     -0.796920
7081 -1.056972 -1.454214  0.917121      -0.121880  0.752014     -0.025246
6601  0.555982 -1.454214 -0.653226       0.740337 -1.592646     -0.411083
…          …        …         …              …         …            …
1971 -0.603124  0.715036 -1.612079       0.378762 -1.201869      0.360591
2348 -0.847096 -0.911902  1.517761       1.129726  0.752014     -0.025246
7882  1.624041  1.618890  0.909884       0.893311  0.752014      0.360591
3898 -0.215810  1.257348  1.535852       0.768151 -0.420316      1.518102
664  -0.140713  0.353494  0.103001       1.741622 -0.420316     -0.796920

      …  Location_Rural  Location_Suburban  Cuisine_American  \
2412  …               0                  1                 0
6832  …               1                  0                 0
5154  …               1                  0                 0
7081  …               0                  1                 0
6601  …               0                  1                 0
…   …               …                  …                 …
1971  …               0                  1                 0
2348  …               0                  1                 0
7882  …               0                  0                 1
3898  …               0                  0                 0
664   …               0                  1                 0

      Cuisine_French  Cuisine_Indian  Cuisine_Italian  Cuisine_Japanese  \
2412               0               1                0                 0
6832               0               0                0                 0
5154               0               0                1                 0
7081               0               0                0                 1
```

```
6601                0                0                0                0
…                 …                …                …                …
1971                0                0                0                0
2348                0                0                0                1
7882                0                0                0                0
3898                0                0                0                1
664                 0                0                0                1

      Cuisine_Mexican  Parking_No  Parking_Yes
2412                0           0            1
6832                1           0            1
5154                0           1            0
7081                0           0            1
6601                1           1            0
…                 …           …            …
1971                1           1            0
2348                0           0            1
7882                0           0            1
3898                0           1            0
664                 0           0            1

[1674 rows x 23 columns]
```

1.7 Model Development

1.7.1 Importing Models

[82]:
```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
```

[83]:
```
lr_model = LinearRegression()
rf_model = RandomForestRegressor()
knn_model = KNeighborsRegressor()
```

[84]:
```
models = [lr_model,rf_model,knn_model]
models
```

[84]:
```
[LinearRegression(), RandomForestRegressor(), KNeighborsRegressor()]
```

1.7.2 Model Training

[86]:
```
for model in models:
    model.fit(x_train,y_train)
```

1.7.3 Model Evaluation

[88]:
```
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

[89]:
```
for model in models:
    model.fit(x_train,y_train)
    y_pred = model.predict(x_test)
    print(model.__class__.__name__)

    accuracy = round(r2_score(y_test, y_pred)*100, 2)
    print(f"Accuracy of {model} = {accuracy} %")

    mse = mean_squared_error(y_test, y_pred)
    print(f"Mean Squared Error of {model} = {mse}")

    mae = mean_absolute_error(y_test, y_pred)
    print(f"Mean Squared Error of {model} = {mae}")

    print("------------------------------------------------------------")
    print(" ")
```

```
LinearRegression
Accuracy of LinearRegression() = 95.51 %
Mean Squared Error of LinearRegression() = 3216912608.570561
Mean Squared Error of LinearRegression() = 43386.16748606759
------------------------------------------------------------

RandomForestRegressor
Accuracy of RandomForestRegressor() = 99.9 %
Mean Squared Error of RandomForestRegressor() = 69820991.78311788
Mean Squared Error of RandomForestRegressor() = 6570.7004007765745
------------------------------------------------------------

KNeighborsRegressor
Accuracy of KNeighborsRegressor() = 92.36 %
Mean Squared Error of KNeighborsRegressor() = 5472748763.850608
Mean Squared Error of KNeighborsRegressor() = 57979.219498207895
------------------------------------------------------------
```

1.7.4 Cross Validation

[91]:
```
from sklearn.model_selection import cross_val_score
```

[92]:
```
%%time
for model in models:
    print(f"Model: {model.__class__.__name__}")
```

```
    r2_scores = cross_val_score(model, x_train, y_train, cv=5, scoring='r2')
    r2_mean = round(np.mean(r2_scores), 2)

    mse_scores = cross_val_score(model, x_train, y_train, cv=5,␣
↪scoring='neg_mean_squared_error')
    mse_mean = -round(np.mean(mse_scores), 2)

    mae_scores = cross_val_score(model, x_train, y_train, cv=5,␣
↪scoring='neg_mean_absolute_error')
    mae_mean = -round(np.mean(mae_scores), 2)

    print(f"R2 (mean): {r2_mean}")
    print(f"Mean Squared Error (mean): {mse_mean}")
    print(f"Mean Absolute Error (mean): {mae_mean}")

    print("-------------------------------------------------------\n")
```

```
Model: LinearRegression
R2 (mean): 0.96
Mean Squared Error (mean): 2993231421.29
Mean Absolute Error (mean): 41589.15
-------------------------------------------------------

Model: RandomForestRegressor
R2 (mean): 1.0
Mean Squared Error (mean): 88982071.96
Mean Absolute Error (mean): 7301.13
-------------------------------------------------------

Model: KNeighborsRegressor
R2 (mean): 0.91
Mean Squared Error (mean): 6188179903.42
Mean Absolute Error (mean): 61237.17
-------------------------------------------------------

CPU times: total: 1min 6s
Wall time: 1min
```

The R^2 score of the Random Forest Regressor is 0.999, which shows that the model is already performing very well. Tuning the hyperparameters further may lead to overfitting.

1.8 Customizing Prediction

[95]:
```
def predict_restaurant_revenue(model, scaler, num_col):
    print("Choose Location:")
    print("1. Downtown")
    print("2. Rural")
```

```
print("3. Suburban")
in_location = input("Enter the Location: ")
while in_location not in ['1', '2', '3']:
    print("Invalid input. Please enter 1, 2, or 3.")
    in_location = input("Enter the Location: ")

location_dict = {'1': 'Downtown', '2': 'Rural', '3': 'Suburban'}
in_location = location_dict[in_location]
print(" ")
print("Choose Cuisine:")
print("1. Japanese")
print("2. Mexican")
print("3. Italian")
print("4. Indian")
print("5. French")
print("6. American")
in_cuisine = input("Enter the Cuisine: ")
while in_cuisine not in ['1', '2', '3', '4', '5', '6']:
    print("Invalid input. Please enter a number between 1 and 6.")
    in_cuisine = input("Enter the Cuisine: ")

cuisine_dict = {
    '1': 'Japanese', '2': 'Mexican', '3': 'Italian',
    '4': 'Indian', '5': 'French', '6': 'American'
    }
in_cuisine = cuisine_dict[in_cuisine]
print(" ")
in_rating = float(input("Enter Rating = "))
in_seating = float(input("Enter Seating Capacity = "))
in_avg_meal_price = float(input("Enter Average Meal Price = "))
in_budget = float(input("Enter Marketing Budget = "))
in_social = float(input("Enter Social = "))
in_chef_exp = float(input("Enter Chef Experience = "))
in_reviews = float(input("Enter Reviews = "))
in_review_len = float(input("Enter Review Length = "))
in_ambience = float(input("Enter Ambience = "))
in_service_qual = float(input("Enter Service Quality = "))
print(" ")
print("Choose Parking Option:")
print("1. Yes")
print("2. No")
in_parking = input("Is Parking Available?: ")
while in_parking not in ['1', '2']:
    print("Invalid input. Please enter 1 or 2.")
    in_parking = input("Is Parking Available?: ")

in_parking = 'Yes' if in_parking == '1' else 'No'
```

```
    in_weekend_res = float(input("Enter Weekend Reservations = "))
    in_weeekday_res = float(input("Enter Weekday Reservations = "))

    input_data = {
        "Location": [in_location],
        "Cuisine": [in_cuisine],
        "Rating": [in_rating],
        "Seating_Capacity": [in_seating],
        "Average_Meal_Price": [in_avg_meal_price],
        "Marketing_Budget": [in_budget],
        "Social": [in_social],
        "Chef_exp": [in_chef_exp],
        "Reviews":[in_reviews],
        "Review_Length": [in_review_len],
        "Ambience": [in_ambience],
        "Service_Qual": [in_service_qual],
        "Parking": [in_parking],
        "Weekend_Reservations": [in_weekend_res],
        "Weekday_Reservations": [in_weeekday_res],
        }

    input_df = pd.DataFrame(input_data)
    input_df_encoded = pd.get_dummies(input_df, columns=Categorical_columns)
    input_df_encoded[num_col] = scaler.transform(input_df_encoded[num_col])

    for col in ['Location_Downtown', 'Location_Rural', 'Location_Suburban']:
        if col not in input_df_encoded.columns:
            input_df_encoded[col] = 0

    for col in ['Cuisine_American', 'Cuisine_French', 'Cuisine_Indian',
       'Cuisine_Italian', 'Cuisine_Japanese', 'Cuisine_Mexican']:
        if col not in input_df_encoded.columns:
            input_df_encoded[col] = 0

    for col in ['Parking_No', 'Parking_Yes']:
        if col not in input_df_encoded.columns:
            input_df_encoded[col] = 0

    input_for_prediction = input_df_encoded.values.reshape(1, -1)
    prediction = rf_model.predict(input_for_prediction)[0]
    return f"Predicted Revenue is $ {prediction:.1f}"
```

1.8.1 Test Case 1

[150]:
```
result = predict_restaurant_revenue(rf_model, scaler, num_col)
print("------------------------------------ ")
print(" ")
print(result)
```

```
Choose Location:
1. Downtown
2. Rural
3. Suburban

Enter the Location:  2


Choose Cuisine:
1. Japanese
2. Mexican
3. Italian
4. Indian
5. French
6. American

Enter the Cuisine:  1


Enter Rating =  4.9
Enter Seating Capacity =  39
Enter Average Meal Price =  67.77
Enter Marketing Budget =  1052
Enter Social =  15791
Enter Chef Experience =  5
Enter Reviews =  365
Enter Review Length =  158.65
Enter Ambience =  3.9
Enter Service Quality =  3


Choose Parking Option:
1. Yes
2. No

Is Parking Available?:  1
Enter Weekend Reservations =  18
Enter Weekday Reservations =  16

--------------------------------------

Predicted Revenue is $ 610433.9
```

1.8.2 Test Case 2

[152]:
```
result = predict_restaurant_revenue(rf_model, scaler, num_col)
print("------------------------------------ ")
print(" ")
print(result)
```

```
Choose Location:
1. Downtown
2. Rural
3. Suburban

Enter the Location:  3


Choose Cuisine:
1. Japanese
2. Mexican
3. Italian
4. Indian
5. French
6. American

Enter the Cuisine:  6


Enter Rating =  3.8
Enter Seating Capacity =  60
Enter Average Meal Price =  39.43
Enter Marketing Budget =  2175
Enter Social =  21775
Enter Chef Experience =  9
Enter Reviews =  690
Enter Review Length =  118.93
Enter Ambience =  2.3
Enter Service Quality =  2.4


Choose Parking Option:
1. Yes
2. No

Is Parking Available?:  2
Enter Weekend Reservations =  51
Enter Weekday Reservations =  29

------------------------------------

Predicted Revenue is $ 539021.5
```

1.8.3 Test Case 3

[154]:
```
result = predict_restaurant_revenue(rf_model, scaler, num_col)
print("------------------------------------ ")
print(" ")
print(result)
```

```
Choose Location:
1. Downtown
2. Rural
3. Suburban

Enter the Location:  2


Choose Cuisine:
1. Japanese
2. Mexican
3. Italian
4. Indian
5. French
6. American

Enter the Cuisine:  5


Enter Rating =  4.2
Enter Seating Capacity =  50
Enter Average Meal Price =  59.19
Enter Marketing Budget =  1368
Enter Social =  18544
Enter Chef Experience =  18
Enter Reviews =  151
Enter Review Length =  177.85
Enter Ambience =  8
Enter Service Quality =  3.8


Choose Parking Option:
1. Yes
2. No

Is Parking Available?:  2
Enter Weekend Reservations =  22
Enter Weekday Reservations =  0

------------------------------------

Predicted Revenue is $ 670684.9
```