

```
# Importing the Libraries
import pandas as pd
import numpy as np
```

```
import warnings
warnings.filterwarnings("ignore")
```

Pre processing steps

```
import pandas as pd
# Creating the Dataframe
df=pd.read_csv('/content/Dataset .csv')
```

df

	Average Cost for two	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	Aggregate rating	Rating color	Rating text	Votes
0	1100	1	0	0	0	3	4.8	0	1	314
1	1200	1	0	0	0	3	4.5	0	1	591
2	4000	1	0	0	0	4	4.4	1	5	270
3	1500	0	0	0	0	4	4.9	0	1	365
4	1500	1	0	0	0	4	4.8	0	1	229
...	...	...	...	...	...	...	...	...	...	...
9546	80	0	0	0	0	3	4.1	1	5	788
9547	105	0	0	0	0	3	4.2	1	5	1034
9548	170	0	0	0	0	4	3.7	5	2	661
9549	120	0	0	0	0	4	4.0	1	5	901
9550	55	0	0	0	0	2	4.0	1	5	591

```
df = df.drop('Restaurant ID', axis=1, errors='ignore')
df = df.drop('Restaurant Name', axis=1, errors='ignore')
df = df.drop('Country Code', axis=1, errors='ignore')
df = df.drop('City', axis=1, errors='ignore')
df = df.drop('Address', axis=1, errors='ignore')
df = df.drop('Locality', axis=1, errors='ignore')
df = df.drop('Locality Verbose', axis=1, errors='ignore')
df = df.drop('Longitude', axis=1, errors='ignore')
df = df.drop('Latitude', axis=1, errors='ignore')
df = df.drop('Cuisines', axis=1, errors='ignore')
df = df.drop('Currency', axis=1, errors='ignore')
```

df

	Average Cost for two	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	Aggregate rating	Rating color	Rating text	Votes
0	1100	Yes	No	No	No	3	4.8	Dark Green	Excellent	314
1	1200	Yes	No	No	No	3	4.5	Dark Green	Excellent	591
2	4000	Yes	No	No	No	4	4.4	Green	Very Good	270
3	1500	No	No	No	No	4	4.9	Dark Green	Excellent	365
4	1500	Yes	No	No	No	4	4.8	Dark Green	Excellent	229
...	...	...	...	...	...	...	...	...	...	...
9546	80	No	No	No	No	3	4.1	Green	Very Good	788
9547	105	No	No	No	No	3	4.2	Green	Very Good	1034
9548	170	No	No	No	No	4	3.7	Yellow	Good	661

df.shape

(9551, 10)

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 10 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Average Cost for two                  9551 non-null   int64
1   Has Table booking                    9551 non-null   int64
2   Has Online delivery                  9551 non-null   int64
3   Is delivering now                    9551 non-null   int64
4   Switch to order menu                 9551 non-null   int64
5   Price range                          9551 non-null   int64
6   Aggregate rating                     9551 non-null   float64
7   Rating color                         9551 non-null   int64
8   Rating text                          9551 non-null   int64
9   Votes                               9551 non-null   int64
dtypes: float64(1), int64(9)
memory usage: 746.3 KB
```

df.describe()

	Average Cost for two	Price range	Aggregate rating	Votes
count	9551.000000	9551.000000	9551.000000	9551.000000
mean	1199.210763	1.804837	2.666370	156.909748
std	16121.183073	0.905609	1.516378	430.169145
min	0.000000	1.000000	0.000000	0.000000
25%	250.000000	1.000000	2.500000	5.000000
50%	400.000000	2.000000	3.200000	31.000000
75%	700.000000	2.000000	3.700000	131.000000
max	800000.000000	4.000000	4.900000	10934.000000

```
# Checking for missing values
df.isnull().sum()
```

	0
Average Cost for two	0
Has Table booking	0
Has Online delivery	0
Is delivering now	0
Switch to order menu	0
Price range	0
Aggregate rating	0
Rating color	0
Rating text	0
Votes	0

dtype: int64

```
# Checking for duplicated values
df.duplicated().sum()
```

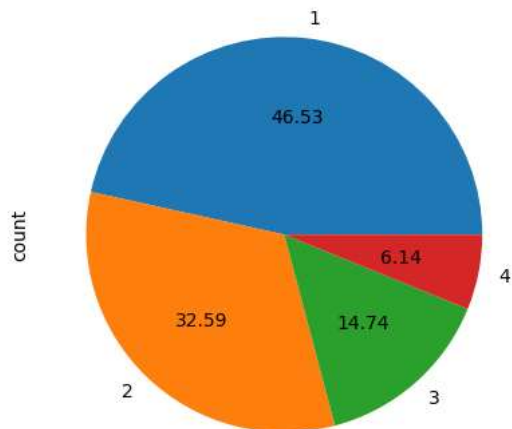
np.int64(2871)

df.dropna(inplace=True)

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

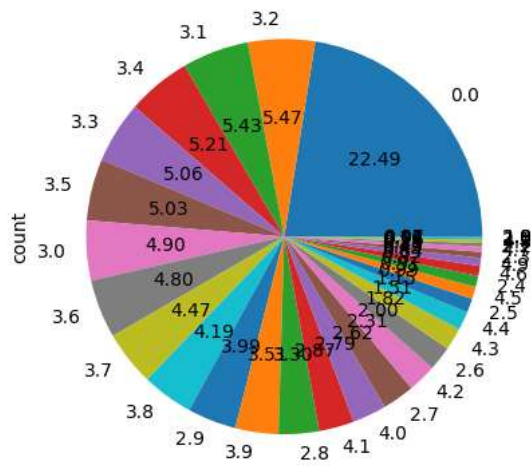
df['Price range'].value\_counts().plot(kind='pie', autopct = '%.2f')

&lt;Axes: ylabel='count'&gt;



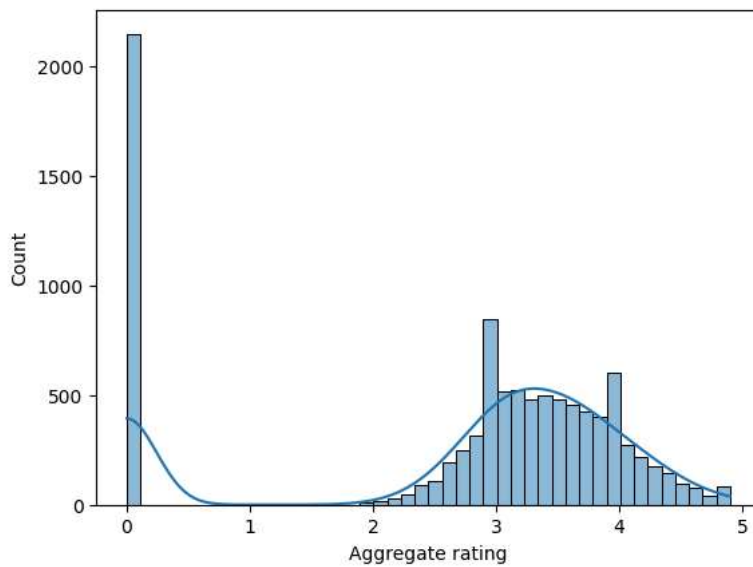
df['Aggregate rating'].value\_counts().plot(kind='pie', autopct = '%.2f')

<Axes: ylabel='count'>



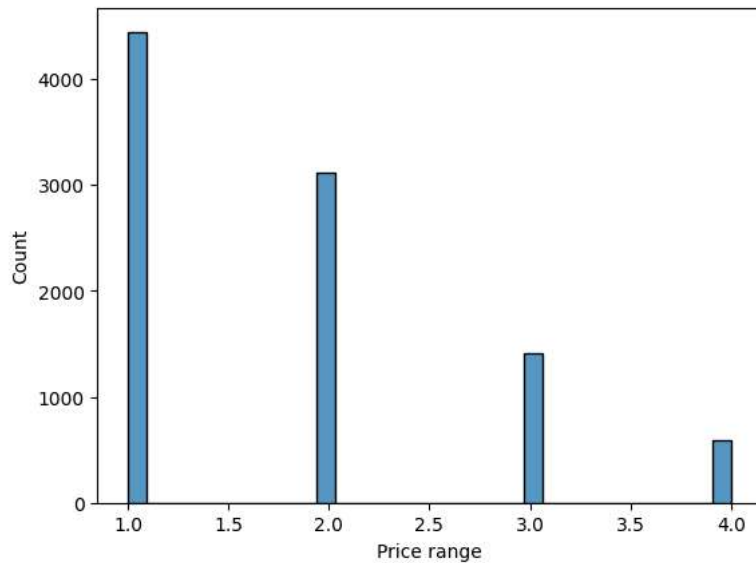
```
sns.histplot(df['Aggregate rating'], kde=True)
```

<Axes: xlabel='Aggregate rating', ylabel='Count'>



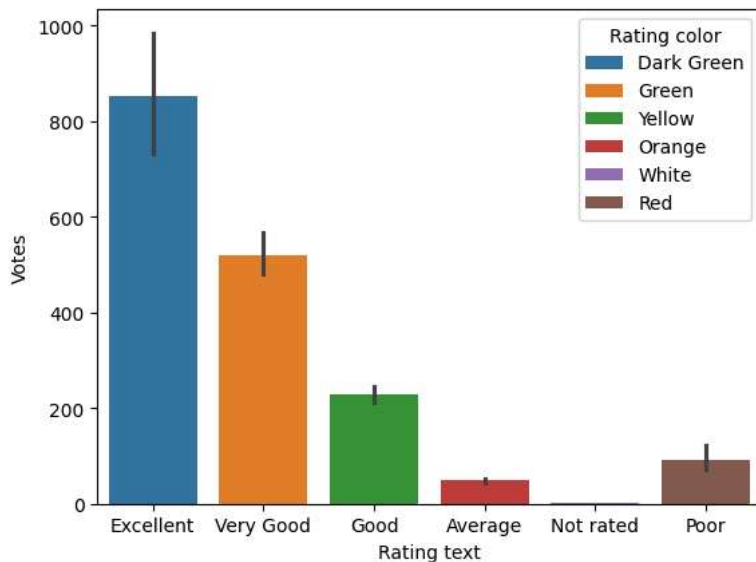
```
sns.histplot(df['Price range'])
```

```
<Axes: xlabel='Price range', ylabel='Count'>
```

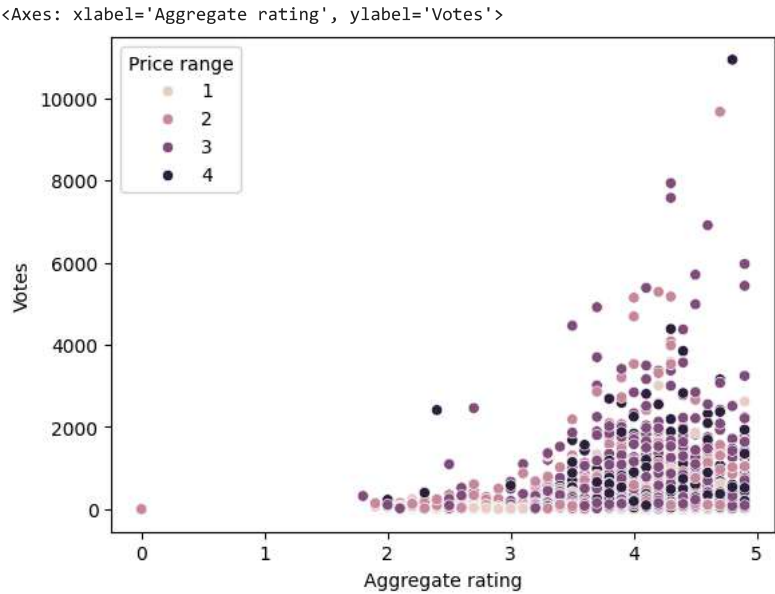


```
sns.barplot(x=df["Rating text"],y=df["Votes"],hue =df["Rating color"])
```

```
<Axes: xlabel='Rating text', ylabel='Votes'>
```



```
sns.scatterplot(x=df["Aggregate rating"],y=df["Votes"],hue=df["Price range"])
```

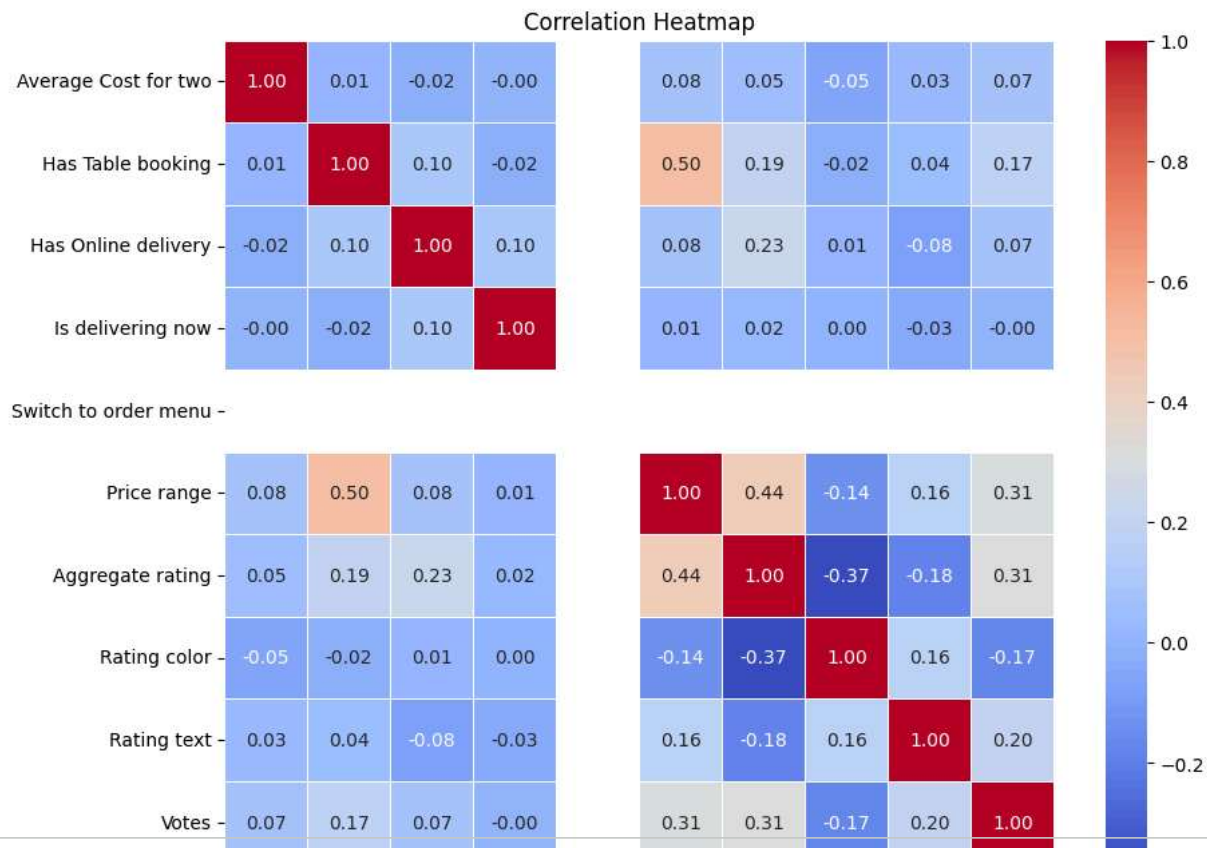


```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
df['Has Table booking'] = label_encoder.fit_transform(df['Has Table booking'])
df['Has Online delivery'] = label_encoder.fit_transform(df['Has Online delivery'])
df['Is delivering now'] = label_encoder.fit_transform(df['Is delivering now'])
df['Switch to order menu'] = label_encoder.fit_transform(df['Switch to order menu'])
df['Rating color'] = label_encoder.fit_transform(df['Rating color'])
df['Rating text'] = label_encoder.fit_transform(df['Rating text'])
```

df

	Average Cost for two	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	Aggregate rating	Rating color	Rating text	Votes
0	1100	1	0	0	0	3	4.8	0	1	314
1	1200	1	0	0	0	3	4.5	0	1	591
2	4000	1	0	0	0	4	4.4	1	5	270
3	1500	0	0	0	0	4	4.9	0	1	365
4	1500	1	0	0	0	4	4.8	0	1	229
...	...	...	...	...	...	...	...	...	...	...
9546	80	0	0	0	0	3	4.1	1	5	788
9547	105	0	0	0	0	3	4.2	1	5	1034
9548	170	0	0	0	0	4	3.7	5	2	661
9549	120	0	0	0	0	4	4.0	1	5	901
9550	55	0	0	0	0	2	4.0	1	5	591

```
correlation_matrix = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```



```
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import r2_score
```

```
x = df.drop('Aggregate rating', axis=1)
y = df['Aggregate rating']
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.1,random_state=353)
x_train.head()
y_train.head()
```

Aggregate rating	
8696	0.0