Swiggy Data Set Analysis & Visualizati... Draft saved

File Edit View Run Settings Add-ons Help

Share Save Version 0

Run All Code Draft Session (53m)

Importing Libraries

[11]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading CSV

+ Code + Markdown

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 201 entries, 0 to 200
Data columns (total 9 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   name                         201 non-null    object
 1   online_order                 201 non-null    object
 2   book_table                   201 non-null    object
 3   rate                         197 non-null    object
 4   votes                        201 non-null    int64
 5   approx_cost(for two people)  201 non-null    object
 6   listed_in(type)              201 non-null    object
 7   listed_in(city)              193 non-null    object
 8   cuisines                     201 non-null    object
dtypes: int64(1), object(8)
memory usage: 14.3+ KB
```

+ Code + Markdown

Dropping Duplicates

[17]:
```
df.drop_duplicates(inplace = True)
df.shape
```

[17]:
```
(201, 9)
```

Cleaning Rate Column

[18]:
```
df['rate'].unique()
```

[18]:
```
array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
       '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
       '4.3/5', '2.9/5', '3.5/5', '2.6/5', '3.8 /5', '3.4/5', nan, 'NEW',
       '4.5/5'], dtype=object)
```

+ Code + Markdown

Removing "NEW" , "-" and "/5" from Rate Column

[19]:
```
def handlerate(value):
    if(value=='NEW' or value=='-'):
        return np.nan
    else:
        value = str(value).split('/')
        value = value[0]
        return float(value)

df['rate'] = df['rate'].apply(handlerate)
df['rate'].head()
```

[19]:
```
0    4.1
1    4.1
2    3.8
3    3.7
4    3.8
Name: rate, dtype: float64
```

Filling Null Values in Rate Column with Mean

[20]:
```
df['rate'].fillna(df['rate'].mean(), inplace = True)
df['rate'].isnull().sum()
```

[20]:
```
0
```

Filling Null Values in Rate Column with Mean

[20]:
```
df['rate'].fillna(df['rate'].mean(), inplace = True)
df['rate'].isnull().sum()
```

[20]:
```
0
```

+ Code + Markdown

[21]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 201 entries, 0 to 200
Data columns (total 9 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   name                         201 non-null    object
```

```
 1   online_order                 201 non-null    object
 2   book_table                   201 non-null    object
 3   rate                         201 non-null    float64
 4   votes                        201 non-null    int64
 5   approx_cost(for two people)  201 non-null    object
 6   listed_in(type)              201 non-null    object
 7   listed_in(city)              193 non-null    object
 8   cuisines                     201 non-null    object
dtypes: float64(1), int64(1), object(7)
memory usage: 15.7+ KB
```

Dropping Null Values

[22]:
```
df['listed_in(city)'].unique()
```

[22]:
```
array(['Banashankari', nan,
       ' it was little difficult to converse. Overall not a bad experience."")]"',
       " ('Rated 1.0'",
       " 'RATED\\n  FAASOS\\n\\nJumbo Chicken Wrap\\n\\nBhuna Chicken & Chicken Tikka Wrap\\n\\nJumbo Chicken Wrap:- The perfect taste
of bhuna ( spicy ) chicken with the combination of mayo sauce",
       ' Bannerghatta Road', " ('Rated 5.0'",
       'their must try would be the sushi',
       ' super large fries and Taiwanese baby corn for starters. The cheese balls were pretty good but they had given chili mayo as acc
ompaniment which was just mayo mixed with chilli powder',
       " ('Rated 4.0'"], dtype=object)
```

[23]:
```
df = df.drop(['listed_in(city)'], axis = 1)
```

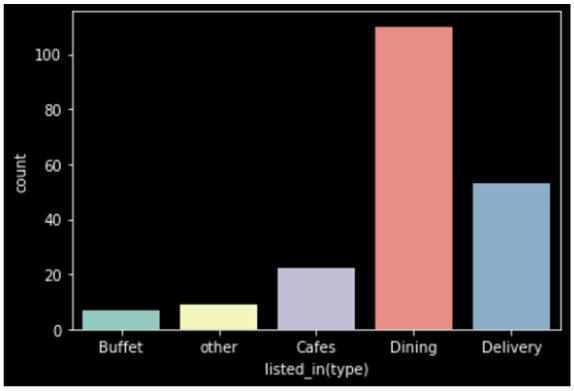
[24]:
```
df.head()
```

[24]:

	name	online_order	book_table	rate	votes	approx_cost(for two people)	listed_in(type)	cuisines
0	Jagdish Farshan	Yes	Yes	4.1	775	800	Buffet	Chinese, North Indian, Thai
1	The Spice	Yes	No	4.1	787	800	Buffet	Cafe, Mexican, Italian
2	Blue Lagoon	Yes	No	3.8	918	800	Buffet	South Indian, North Indian
3	Udupi Bhojana	No	No	3.7	88	300	Buffet	North Indian, Rajasthani
4	Ghee God	No	No	3.8	166	600	Buffet	North Indian

Type of Restaurent

```
sns.countplot(x=df['listed_in(type)'])
plt.xtable("Type of Restaurent")
```



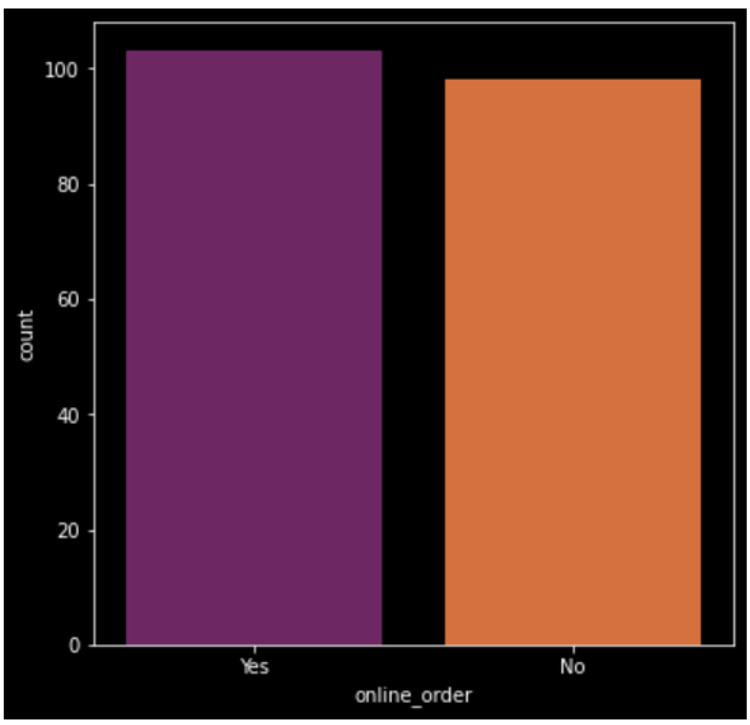
+ Code + Markdown

Visualizing Online Order

+ Code + Markdown

```
plt.figure(figsize = (6,6))
sns.countplot(df['online_order'], palette = 'inferno')
```

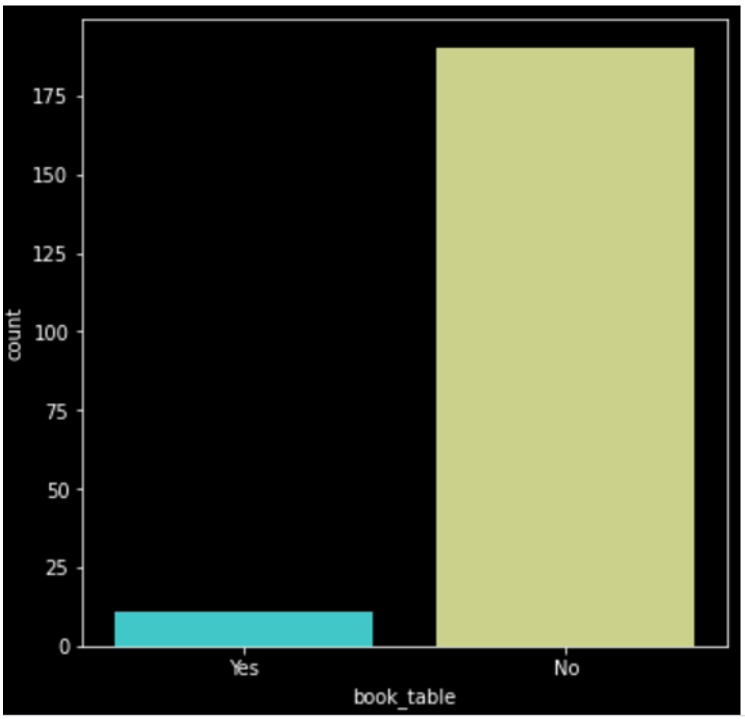
```
<AxesSubplot:xlabel='online_order', ylabel='count'>
```



Visualizing Book Table

```
plt.figure(figsize = (6,6))
sns.countplot(df['book_table'], palette = 'rainbow')
```

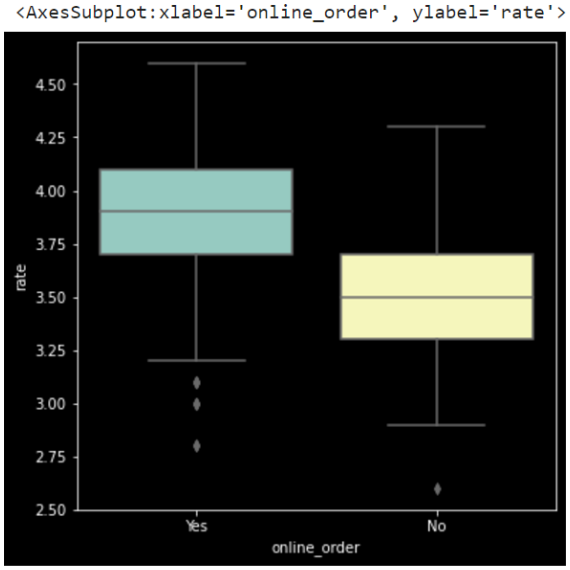
```
<AxesSubplot:xlabel='book_table', ylabel='count'>
```



Visualizing Online Order vs Rate

```
]: plt.figure(figsize = (6,6))
sns.boxplot(x = 'online_order', y = 'rate', data = df)
```

```
[28]: <AxesSubplot:xlabel='online_order', ylabel='rate'>
```



[31]:
```
df.head()
```

[31]:

	name	online_order	book_table	rate	votes	approx_cost(for two people)	listed_in(type)	cuisines
0	Jagdish Farshan	Yes	Yes	4.1	775	800	Buffet	Chinese, North Indian, Thai
1	The Spice	Yes	No	4.1	787	800	Buffet	Cafe, Mexican, Italian
2	Blue Lagoon	Yes	No	3.8	918	800	Buffet	South Indian, North Indian
3	Udupi Bhojana	No	No	3.7	88	300	Buffet	North Indian, Rajasthani
4	Ghee God	No	No	3.8	166	600	Buffet	North Indian

Visualizing Top Cuisines

```
df6 = df[['cuisines', 'votes']]
df6.drop_duplicates()
df7 = df6.groupby(['cuisines'])['votes'].sum()
df7 = df7.to_frame()
df7 = df7.sort_values('votes', ascending=False)
df7.head()
```

[38]:

cuisines	votes
North Indian, Chinese, Fast Food	4884
South Indian, Chinese, North Indian	4401
South Indian	4055
Biryani	2761
Cafe, Italian, Continental	2706