ZOMATO RESTAURANT PROJECT - PYTHON

Data Analytics

JANUARY 30, 2024

DONE BY: JAISHREE SRINIVASAN
Jais.srinivasan@gmail.com

Zomato Restaurants Dataset for Metropolitan areas

Navigating to dataset for reference https://www.kaggle.com/datasets/narsingraogoud/zomato-restaurants-dataset-for-metropolitan-areas

About the Client:

I'm working on the Ecommerce domain with restaurants tied up with food delivery partner "Zomato" application in Indian metropolitan areas. In this dataset, we have more than 127000 rows and 12 columns, a fairly large dataset. We have variables like Restaurant Name, Dining Rating, Delivery Rating, Dining Votes, Delivery Votes, Cuisine, Place Name, City, Item Name, Best Seller, Votes, Prices

As a Data analyst, Loaded the dataset for cleaning and processing the data, to find insightful results.

```
#importing libraries

import pandas as pd
import numpy as np
import matplotlib.pylab as plt
import seaborn as sns
plt.style.use('ggplot')
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestRegressor
import pickle
import warnings
warnings.filterwarnings('ignore')
```

```
#Loading the data and sampling the data transpose
df = pd.read_excel('/content/drive/MyDrive/Zomato/zomato_dataset.xlsx')
df.sample(5).T
```

	82838	27114	799	8700	33496
Restaurant Name	The Kebabish	Firangi Bake	Taj Mahal - Taj Mahal Hotel	Pista House Bakery	Shree Konar Vilas
Dining Rating	3.8	NaN	4.1	NaN	4.3
Delivery Rating	3.7	4.1	4.1	4.3	3.9
Dining Votes	337	0	0	0	208
Delivery Votes	0	737	0	0	0
Cuisine	Fast Food	Mexican	Beverages	Beverages	Biryani
Place Name	Kankaria	Wadala	Taj Mahal Hotel	Charminar	Purasavakkam
City	Ahmedabad	Mumbai	Hyderabad	Hyderabad	Chennai
Item Name	Egg Biryani [2 Eggs]	Margherita Pizza (Medium Pizza)	Ghee Roast Plain Dosa	Chicken Cheese Roll	Chicken Fried Rice
Best Seller	BESTSELLER	NaN	NaN	NaN	NaN
Votes	0	0	7	17	8
Prices	229.0	330.0	150.0	120.0	250.0

Observations:

- Column name have spaces, should be edited
- Each variable has its unique value whereas placename and city together make sense
- There are missing values in certain variables like Dining Rating, Delivery Rating, Best Seller
- Need to clean individual column for better results
- Remove NaN values and equalize the rows for better analysis

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 123657 entries, 0 to 123656
Data columns (total 12 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   Restaurant Name  123657 non-null  object
 1   Dining Rating    91421 non-null   float64
 2   Delivery Rating  122377 non-null  float64
 3   Dining Votes     123657 non-null  int64
 4   Delivery Votes   123657 non-null  int64
 5   Cuisine          123657 non-null  object
 6   Place Name       123657 non-null  object
 7   City             123657 non-null  object
 8   Item Name        123657 non-null  object
 9   Best Seller      27942 non-null   object
 10  Votes            123657 non-null  int64
 11  Prices           123657 non-null  float64
dtypes: float64(3), int64(3), object(6)
memory usage: 11.3+ MB
```

Cleaning and Preprocessing

- Dropping nulls values in all the variables if available Removed column name 'City'

```
[ ] df.columns = df.columns.str.replace(' ', '') #df.rename(columns={'Cuisine ':'Cuisine'})
    df['PlaceName']=df['PlaceName'] + ',' + df['City']
    # Remove column name 'City'
    df=df.drop(['City'], axis=1)
    df.head(10)
```

	RestaurantName	DiningRating	DeliveryRating	DiningVotes	DeliveryVotes	Cuisine	PlaceName	ItemName	BestSeller	Votes	Prices
0	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Platter Kebab Combo	BESTSELLER	84	249.0
1	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Chicken Rumali Shawarma	BESTSELLER	45	129.0
2	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Chicken Tandoori Salad	NaN	39	189.0
3	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Chicken BBQ Salad	BESTSELLER	43	189.0
4	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Special Doner Wrap Combo	MUST TRY	31	205.0
5	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Chicken Tandoori Pizza [8 inches]	BESTSELLER	48	199.0
6	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Special Zinger Tortilla Wrap	CHEF'S SPECIAL	27	165.0
7	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Chicken Popcorn [20 Pieces]	BESTSELLER	59	165.0
8	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Chicken Tandoori Sandwich	NaN	29	115.0
9	Doner King	3.9	4.2	39	0	Fast Food	Malakpet, Hyderabad	Chicken Bread Samoli Shawarma	NaN	31	129.0

- Substituting the null values with average values for better analysis

```
#determines the null values
df.isnull().sum()
```

```
RestaurantName        0
DiningRating      32236
DeliveryRating     1280
DiningVotes           0
DeliveryVotes         0
Cuisine               0
PlaceName             0
ItemName              0
BestSeller        95715
Votes                 0
Prices                0
dtype: int64
```

```
#substituting the null values with average values for better analysis
df['DiningRating'].fillna(df['DiningRating'].mean(),inplace=True)
df['DeliveryRating'].fillna(df['DeliveryRating'].mean(),inplace=True)
#dropping nulls values in all the variables if available
df.dropna(axis=0,inplace=True)
df.isnull().sum()
```

```
RestaurantName    0
DiningRating      0
DeliveryRating    0
DiningVotes       0
DeliveryVotes     0
Cuisine           0
PlaceName         0
ItemName          0
BestSeller        0
Votes             0
Prices            0
dtype: int64
```

Insights

1. Correlation between each variable been found using heatmap

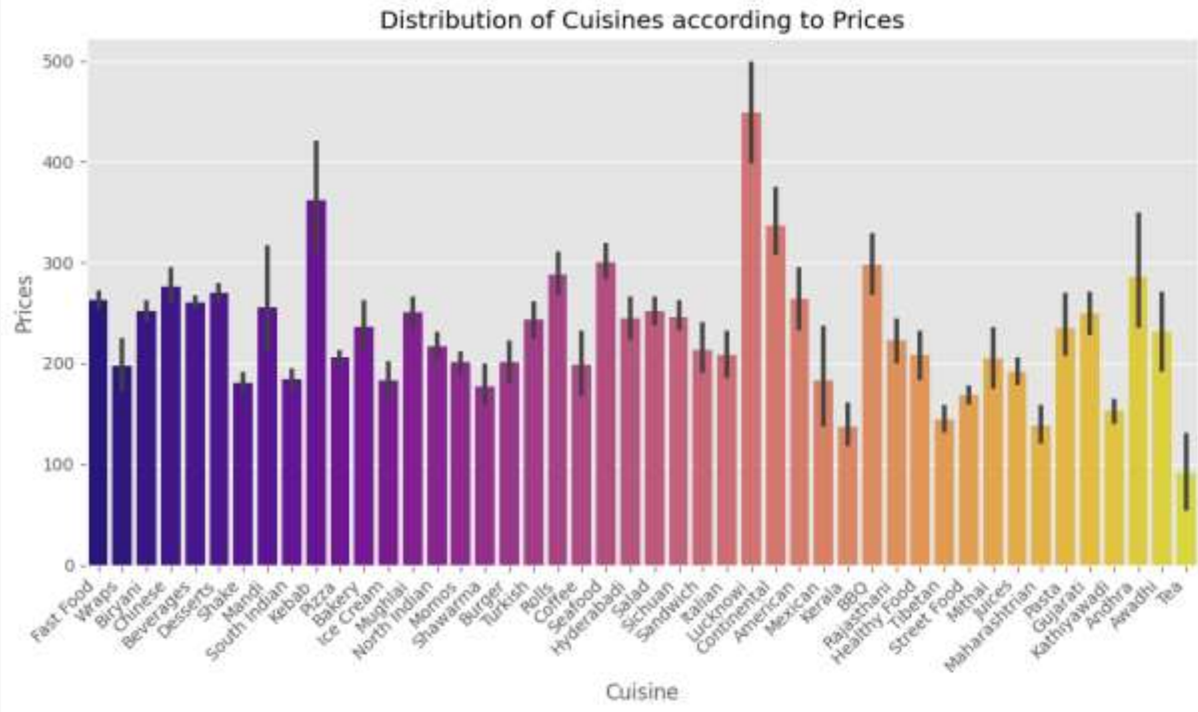
```
plt.figure(figsize=(15,10))
sns.heatmap(df.corr(numeric_only=True), annot=True,
cmap='coolwarm', fmt=".2f", )
plt.show();
```

2. Average delivery rating shows the restaurants received reviews for takeouts Utilized histogram to visualize it to find the average delivery rating Result: 3.95-4.2 is the average value.

```
ab=df['DeliveryRating'].plot(kind='hist', bins=10, title='Average Delivery rating', color ='green', ec ='black')
ab.set_xlabel('Delivery ratings');
```

3. Depending on cuisines distribution of prices been visualized Utilized barplot to represent the cuisines highest to lowest price Result: Lucknowi, Kebab and Continental takes first 3 position and Maharastrian, Kerala and tea are the cuisines take last 3.

```
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='Cuisine', y='Prices', palette='plasma')
plt.xlabel('Cuisine')
plt.ylabel('Prices')
plt.title('Distribution of Cuisines according to Prices')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show();
```



4. Top 10 Restaurant Names by Dining Rating and their special dishes Utilized Pie chart and subplot graphs to show the Top 10 restaurants and their special dishes Result: Natural Icecream- Tender coconut icecream tops in Dining Rating among others.

```
#Top 10 Restaurant Names by Dining Rating and their special dishes
rest_dining=
df.groupby('RestaurantName')['DiningRating'].mean().reset_index()
sorted_dining=
rest_dining.sort_values('DiningRating',ascending=False).head(10)

Restaurant= sorted_dining['RestaurantName']
ratings= sorted_dining['DiningRating']
```

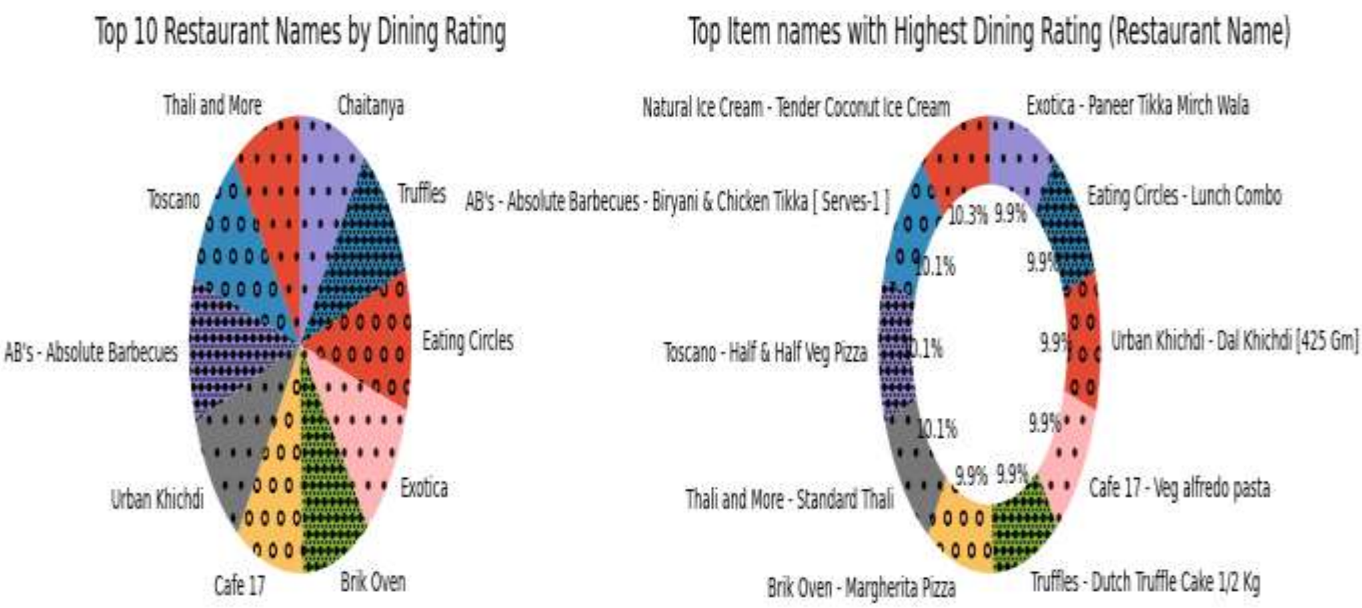
```
plt.figure(figsize=(14, 6))
# First Pie Chart
plt.subplot(1, 2, 1)  # 1 row, 2 columns, subplot 1
plt.pie(ratings, labels=Restaurant, startangle=90, hatch=['.', 'o',
'..oo..'])
plt.title('Top 10 Restaurant Names by Dining Rating')


top_items                                                          =
df.loc[df.groupby('RestaurantName')['DiningRating'].idxmax()]
top_items_sorted      =      top_items.sort_values('DiningRating',
ascending=False).head(10)

items = top_items_sorted['ItemName']
restaurants_top_items = top_items_sorted['RestaurantName']
ratings_top_items = top_items_sorted['DiningRating']

plt.subplot(1, 2, 2)  # 1 row, 2 columns, subplot 2
plt.pie(ratings_top_items,  labels=[f"{restaurant}  -  {item}"  for
restaurant,    item    in    zip(restaurants_top_items,    items)],
autopct='%1.1f%%',                                    startangle=90,
wedgeprops=dict(width=0.3),hatch=['.', 'o', '..oo..'])
plt.title('Top  Item  names  with  Highest  Dining  Rating  (Restaurant
Name)')
plt.tight_layout()

plt.show()
```



5. Most popular Restaurant by Place in India Calculated Total Rating by adding both Dining and Delivery Rating Result: Top 10 popular restaurant been listed as per place name and Rating: 9.3 Place Name : Connaught Place, New Delhi Restaurant Name :Natural Ice Cream

```
# Most popular Restaurant by Place in India
df['Total_rating'] = df['DiningRating'] + df['DeliveryRating']
rating_max = df.groupby(['PlaceName', 'RestaurantName'],
as_index=False)['Total_rating'].max()
rating_max =
rating_max.loc[rating_max.groupby('PlaceName')['Total_rating'].idxma
x()]
rating_max =
rating_max.set_index('PlaceName').round({'Total_rating': 1})
rating_max = rating_max.sort_values(by='Total_rating',
ascending=False)
rating_max.head(10)
```

PlaceName	RestaurantName	Total_rating
Connaught Place, New Delhi	Natural Ice Cream	9.3
12th Square Building, Hyderabad	Exotica	8.9
Rajinder Nagar, New Delhi	Kings Kulfi	8.9
St. Marks Road, Bangalore	Truffles	8.9
Dadar West, Mumbai	Chaitanya	8.9
C Scheme, Jaipur	Thali and More	8.8
Kaloor, Kochi	Al Taza	8.8
Carter Road, Mumbai	Boojee Cafe	8.8
Nungambakkam, Chennai	Toscano	8.8
City Centre 1, Kolkata	Momo I Am	8.7

Machine learning

For customer retention, customer satisfaction and to improve the delivery service we took Delivery Rating as target variable, have done the following steps:

Initially, Split the data into training and testing sets and Defined preprocessing steps. Also, defined the pipeline and train the model. Later, Evaluated the model.

```python
X = df.drop('DeliveryRating', axis=1)
y = df['DeliveryRating']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)


numeric_features = X.select_dtypes(include=['int64',
'float64']).columns
categorical_features = X.select_dtypes(include=['object']).columns

numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ])

pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(random_state=42))  # Use a
regression model
])

pipeline.fit(X_train, y_train)
y_pred = pipeline.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

- Calculated the R2 score and Mean Squared error (MSE)

```
Mean Squared Error: 1.3271764449324307e-06
R-squared: 0.9999789803836363
```

- Created a pickle file to use the model later with more data been added also.

```
# Save the entire pipeline to a pickle file
with open('/content/drive/MyDrive/Zomato/delivery_rating_model.pkl',
'wb') as file:
    pickle.dump(pipeline, file)
```

- For Stakeholder collaboration created cleaned zomato dataset file in csv format

```
#For a colloborative environment, to share it to my stakeholders unable
to understand cleaning process
#exporting this excel file and sharing it.
df.to_csv('/content/drive/MyDrive/Zomato/cleaned_dataset.csv',
index=False)
```

Conclusion:

MSE measures the average squared difference between the predicted values and the actual values. It is calculated as the average of the squared differences between each predicted and actual value. In the above dataset the MSE value is approximately 1.33e-06, which is a very low value. Lower MSE values indicate better model performance, as it means that the model's predictions are close to the actual values.

R-squared is a statistical measure of how well the regression predictions approximate the real data points. It is a value between 0 and 1, where 1 indicates a perfect fit, and 0 indicates that the model does not explain the variance in the target variable. In the above dataset, the R-squared value is approximately 0.999979, which is very close to 1. This suggests that the model explains a high percentage (about 99.9979%) of the variance in the delivery ratings.

R-squared is particularly useful for understanding the proportion of the target variable's variability that can be explained by the model. A high R-squared indicates a good fit of the model to the data.