**Zomato Restaurant Evaluation Project**

**Data Analysis: An In-depth Study**

**1. Problem Definition**

The problem statement for this project is to analyze the Zomato dataset and provide insights on the average cost for two people dining at a restaurant and classify restaurants into different price ranges based on their offerings and attributes. This analysis will assist food enthusiasts and cost-conscious diners in finding affordable and high-quality dining options. By predicting the average cost for two and categorizing restaurants into price ranges, users can make informed decisions on where to dine, ensuring a satisfying dining experience. The analysis aims to help users discover value-for-money restaurants and explore diverse cuisines in different regions, ultimately enhancing the overall dining experience.

**2. Data Analysis**

The Zomato.csv dataset contains extensive information about restaurants, with fields such as restaurant identifiers, location data (such as city, latitude, and longitude), cuisine types, cost details, rating information, and more. This dataset will be the primary source for our analysis.

The country\_code.csv dataset serves as a reference mapping country codes to country names. Utilizing this dataset, we can enhance the Zomato dataset by replacing country codes with readable country names, providing better context and readability to our analysis.

By combining these datasets, we can analyze various aspects of restaurant data across different countries, such as popular cuisines, cost trends, rating distributions, and more. This data analysis will provide valuable insights into the restaurant industry and consumer preferences in different geographical regions.

Overall, the combination of the Zomato.csv and country\_code.csv datasets will allow us to conduct a comprehensive analysis of restaurant data, leveraging the information provided in both datasets to gain meaningful insights and draw actionable conclusions.

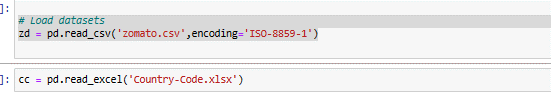
**Data Merging**

Before we begin with EDA, we need to merge the **Zomato.csv** and **country\_code.csv** datasets to enhance our dataset with country names.

**Merging Datasets**

We use the pandas library to merge the two datasets on the **Country Code** column. This will provide a complete dataset with readable country names instead of numeric codes.

Python





**Data Overview**

**Zomato.csv** includes the following key variables:

* **Restaurant Id**: Unique identifier for each restaurant.
* **Restaurant Name**: Name of the restaurant.
* **Country Code**: Numeric code representing the country.
* **City**: City where the restaurant is located.
* **Address**: Physical address of the restaurant.
* **Locality**: Specific area within the city.
* **Locality Verbose**: Detailed description of the locality.
* **Longitude & Latitude**: Geographic coordinates.
* **Cuisines**: Types of cuisines offered.
* **Average Cost for Two**: Cost for two people.
* **Currency**: Currency used in the cost.
* **Has Table Booking**: Availability of table booking (yes/no).
* **Has Online Delivery**: Availability of online delivery (yes/no).
* **Is Delivering**: Indicates if the restaurant is delivering (yes/no).
* **Switch to Order Menu**: Availability of an order menu (yes/no).
* **Price Range**: Price category of the restaurant.
* **Aggregate Rating**: Average customer rating.
* **Rating Color**: Color code based on the rating.
* **Rating Text**: Descriptive rating text.
* **Votes**: Number of ratings given by customers.

**country\_code.csv** contains:

* **Country Code**: Numeric code.
* **Country Name**: Corresponding country name.

**3. EDA Concluding Remarks**

EDA helps uncover patterns and insights within the dataset. Key steps in EDA include:

1. **Understanding Data Distribution**: Examine the distribution of key variables like **Average Cost for Two**, **Aggregate Rating**, and **Votes**.
2. **Identifying Missing Values**: Determine the extent of missing data and decide on strategies to handle it (e.g., imputation or removal).
3. **Visualizing Relationships**: Use visualizations to explore relationships between variables, such as **Average Cost for Two** versus **Aggregate Rating**.

**Key Insights from EDA**

1. **Geographical Spread** - The dataset includes restaurants from various countries, providing a global perspective on dining costs and cuisines. This diversity allows us to analyze and compare dining trends across different regions.
2. **Cuisine Diversity**: The dataset offers a wide range of cuisines, indicating the rich culinary diversity available. This insight is valuable for understanding global food trends and preferences.
3. **Cost Variation** There is significant variation in the average cost for two, influenced by geographic location, cuisine type, and restaurant features. For example, the cost in metropolitan cities is generally higher than in smaller towns or cities.
4. **Rating Distribution**: Customer ratings vary widely, with some restaurants enjoying high popularity while others receive lower ratings. Understanding these patterns can help in identifying factors that contribute to customer satisfaction.
5. **Availability of Services:** The availability of services like table booking and online delivery can impact customer preferences. Analyzing these features helps in understanding their influence on customer choices.

**# Distribution of Average Cost for Two**

plt.figure(figsize=(10, 6))

sns.histplot(combined\_data['Average Cost for two'], bins=30, kde=True, color='blue')

plt.title('Distribution of Average Cost for Two')

plt.xlabel('Average Cost for Two')

plt.ylabel('Frequency')

plt.show()

**# Distribution of Price Range**

plt.figure(figsize=(6, 4))

sns.countplot(x='Price range', data=combined\_data, palette='Set2')

plt.title('Distribution of Price Range')

plt.xlabel('Price Range')

plt.ylabel('Count')

plt.show()

**# Relationship between Aggregate Rating and Average Cost for Two**

plt.figure(figsize=(10, 6))

sns.scatterplot(x='Aggregate rating', y='Average Cost for two', data=combined\_data, color='green')

plt.title('Average Cost for Two vs. Aggregate Rating')

plt.xlabel('Aggregate Rating')

plt.ylabel('Average Cost for Two')

plt.show()

**# Boxplot of Price Range vs. Aggregate Rating**

plt.figure(figsize=(10, 6))

sns.boxplot(x='Price range', y='Aggregate rating', data=combined\_data, palette='Pastel1')

plt.title('Price Range vs. Aggregate Rating')

plt.xlabel('Price Range')

plt.ylabel('Aggregate Rating')

plt.show()

**#Top Cuisines by Frequency**:

plt.figure(figsize=(12, 6))

top\_cuisines = combined\_data['Cuisines'].value\_counts().head(10)

sns.barplot(x=top\_cuisines.values, y=top\_cuisines.index, palette='viridis')

plt.title('Top 10 Cuisines by Frequency')

plt.xlabel('Frequency')

plt.ylabel('Cuisines')

plt.show()

**#`Restaurant Count by Country:**

plt.figure(figsize=(10, 6))

country\_restaurant\_count = combined\_data['Country'].value\_counts()

sns.barplot(x=country\_restaurant\_count.values, y=country\_restaurant\_count.index, palette='magma')

plt.title('Number of Restaurants by Country')

plt.xlabel('Number of Restaurants')

plt.ylabel('Country')

plt.show()

**#Table Booking and Online Delivery**

fig, axes = plt.subplots(1, 2, figsize=(12, 6))

sns.countplot(x='Has Table booking', data=combined\_data, ax=axes[0], palette='Set1')

axes[0].set\_title('Table Booking Availability')

axes[0].set\_xlabel('Has Table booking')

axes[0].set\_ylabel('Count')

sns.countplot(x='Has Online delivery', data=combined\_data, ax=axes[1], palette='Set2')

axes[1].set\_title('Online Delivery Availability')

axes[1].set\_xlabel('Has Online delivery')

axes[1].set\_ylabel('Count')

plt.tight\_layout()

plt.show()

**#Distribution of Aggregate Rating:**

plt.figure(figsize=(10, 6))

sns.histplot(combined\_data['Aggregate rating'], bins=30, kde=True, color='purple')

plt.title('Distribution of Aggregate Rating')

plt.xlabel('Aggregate Rating')

plt.ylabel('Frequency')

plt.show()#Average Cost for Two by Price Range:

plt.figure(figsize=(10, 6))

sns.boxplot(x='Price range', y='Average Cost for two', data=combined\_data, palette='coolwarm')

plt.title('Average Cost for Two by Price Range')

plt.xlabel('Price Range')

plt.ylabel('Average Cost for Two')

plt.show()

**#Average Cost for Two by Price Range:**

plt.figure(figsize=(10, 6))

sns.boxplot(x='Price range', y='Average Cost for two', data=combined\_data, palette='coolwarm')

plt.title('Average Cost for Two by Price Range')

plt.xlabel('Price Range')

plt.ylabel('Average Cost for Two')

plt.show()

**# Distribution of Votes**

plt.hist(combined\_data['Votes'], color='skyblue', edgecolor='black')

plt.title('Distribution of Votes')

plt.xlabel('Votes')

plt.ylabel('Frequency')

plt.tight\_layout()

plt.show()

**# Correlation matrix**

correlation\_matrix = combined\_data[['Average Cost for two', 'Price range', 'Aggregate rating', 'Votes']].corr()

plt.figure(figsize=(8, 6))

sns.heatmap(correlation\_matrix, annot=True, cmap='Pastel1', linewidths=0.5)

plt.title('Correlation Matrix')

plt.show()

**# Heatmap: Rating Text vs. Has Online Delivery**

crosstab = pd.crosstab(combined\_data['Rating text'], combined\_data['Has Online delivery'])

plt.figure(figsize=(8, 6))

sns.heatmap(crosstab, annot=True, fmt='d', cmap='Pastel2')

plt.title('Heatmap: Rating Text vs. Has Online Delivery')

plt.xlabel('Has Online Delivery')

plt.ylabel('Rating Text')

plt.show()

**# Heatmap: Rating Text vs. Has Table Booking**

crosstab2 = pd.crosstab(combined\_data['Rating text'], combined\_data['Has Table booking'])

plt.figure(figsize=(8, 6))

sns.heatmap(crosstab2, annot=True, fmt='d', cmap='Pastel2')

plt.title('Heatmap: Rating Text vs. Has Table Booking')

plt.xlabel('Has Table Booking')

plt.ylabel('Rating Text')

plt.show()

**# Encode categorical columns**

combined\_data['Has Table booking'] = label\_encoder.fit\_transform(combined\_data['Has Table booking'])

combined\_data['Has Online delivery'] = label\_encoder.fit\_transform(combined\_data['Has Online delivery'])

combined\_data['Is delivering now'] = label\_encoder.fit\_transform(combined\_data['Is delivering now'])

combined\_data['Switch to order menu'] = label\_encoder.fit\_transform(combined\_data['Switch to order menu'])

combined\_data['Cuisines'] = label\_encoder.fit\_transform(combined\_data['Cuisines'])

combined\_data['City'] = label\_encoder.fit\_transform(combined\_data['City'])

combined\_data['Rating text'] = label\_encoder.fit\_transform(combined\_data['Rating text'])

combined\_data['Rating color'] = label\_encoder.fit\_transform(combined\_data['Rating color'])

combined\_data['Locality'] = label\_encoder.fit\_transform(combined\_data['Locality'])

combined\_data['Locality Verbose'] = label\_encoder.fit\_transform(combined\_data['Locality Verbose'])

combined\_data['Price range'] = label\_encoder.fit\_transform(combined\_data['Price range'])

**EDA Concluding Remarks**

Initial exploration of the dataset reveals a rich variety of information about restaurants worldwide. The diversity in cuisine types and the range of dining costs offer a comprehensive view of the global food industry. These insights will guide us in refining our approach to building predictive models for **Average Cost for Two** and **Price Range**. By understanding the underlying patterns and relationships in the data, we can develop more accurate and reliable models.

**4. Pre-processing Pipeline**

The pre-processing pipeline involves several key steps to clean and prepare the data for analysis and modeling:

1. Merging Datasets: Combine Zomato.csv and country\_code.csv to add country names.
2. Handling Missing Values: Identify and handle missing values in critical columns such as Average Cost for Two, Cuisines, and Aggregate Rating.
3. Data Type Conversion: Ensure appropriate data types for each column, such as converting cost columns to numeric types.
4. Currency Standardization: Standardize costs to a single currency for consistent comparison.
5. Feature Engineering: Create new features that might help in prediction, such as binary indicators for table booking and online delivery.
6. Encoding Categorical Variables: Convert categorical variables like City, Cuisines, and Rating Text into numerical format using techniques like one-hot encoding or label encoding.
7. Scaling Features: Scale numerical features to ensure all features contribute equally to the model.

**5. Building Machine Learning Models**

The model-building process involves selecting appropriate algorithms, training the models, and evaluating their performance:

**Model Selection**

For predicting the Average Cost for Two, regression models are suitable. For predicting the Price Range, classification models are appropriate**.**

**Regression Models:**

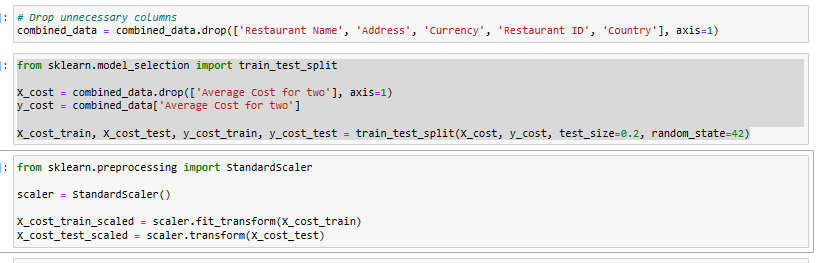
* Linear Regression: Simple yet effective for basic linear relationships.
* Random Forest Regressor: Captures non-linear relationships and interactions between features.

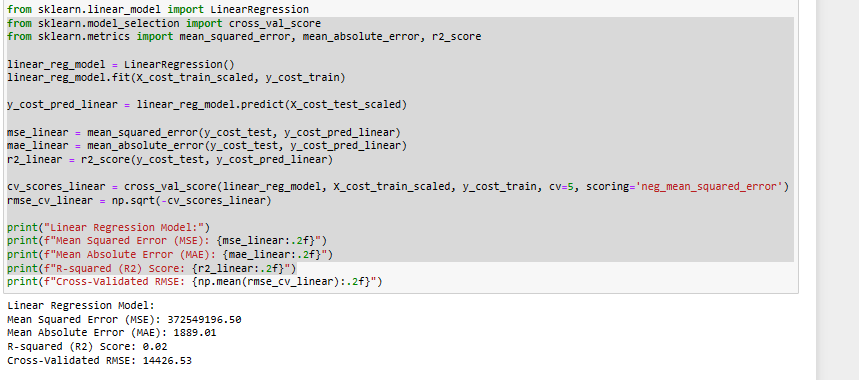
**Classification Models:**

* Logistic Regression: Effective for binary classification, extended to multi-class problems.
* Random Forest Classifier: Handles multi-class classification with robustness.

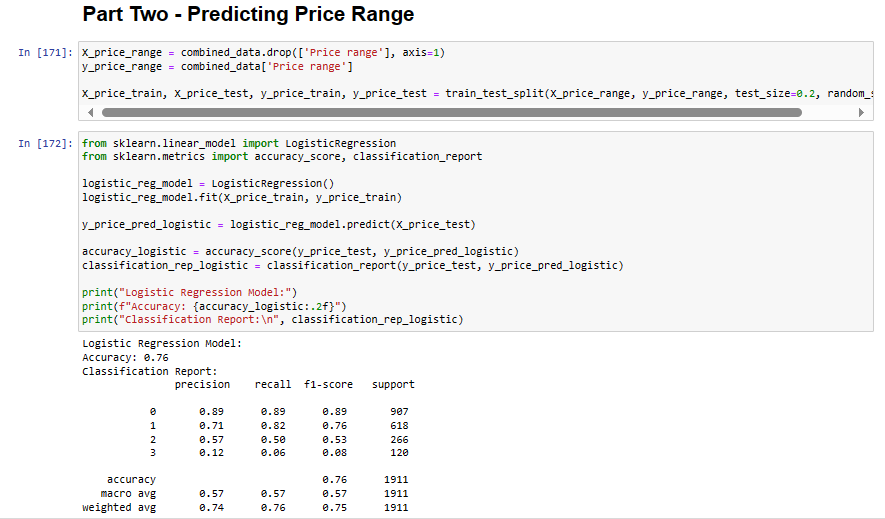
**Model Training and Evaluation**

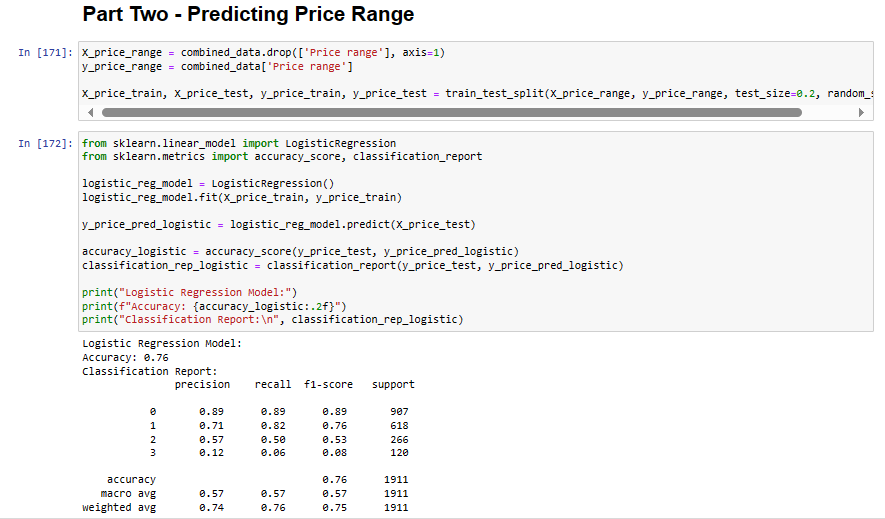
1. Data Splitting: Divide the dataset into training and test sets to evaluate model performance.
2. Hyperparameter Tuning: Use techniques like Grid Search or Random Search to find the best hyperparameters.
3. Model Training: Train the selected models on the training data.
4. Evaluation Metrics: Use metrics such as Mean Absolute Error (MAE) for regression and accuracy, precision, recall, and F1-score for classification.











**6. Concluding Remarks**

The Zomato Data Analysis project offers valuable insights into dining costs and restaurant characteristics worldwide. By predicting the average cost for two and classifying restaurants into price ranges, the analysis assists users in making informed dining decisions. The project demonstrates the effective use of data analysis and machine learning techniques to extract meaningful information from complex datasets, ultimately enhancing the dining experience for food enthusiasts globally.