TITLE OF PROJECT

Zomato Data Analysis Using Python

INTERNSHIP PROJECT REPORT

Submitted in partial fulfillment of the requirement for the award of a certificate of internship programme

by

PODAPATI JAHNAVI

May 2024

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to **Cloudcredits Company** for providing me with the opportunity to work on the project **"Zomato Data Analysis Using Python"** during my internship. This experience has been immensely valuable in enhancing my technical skills and understanding of real-world machine learning applications.

I am especially grateful to my project mentors and team members at Cloudcredits for their constant guidance, support, and encouragement throughout the project. Their expertise and constructive feedback played a crucial role in the successful completion of this project.

Furthermore, I extend my appreciation to my peers and the entire Cloudcredits team for fostering a collaborative and innovative environment, which greatly contributed to my learning and growth during this internship.

Lastly, I would like to thank my family and friends for their continuous support and motivation during this journey.

TABLE OF CONTENT

ABSTRACT

The rise of online food delivery platforms has significantly transformed the food and hospitality industry, offering immense opportunities for data-driven decision-making. This project focuses on the comprehensive analysis of a Zomato dataset collected from Kaggle, using Python-based data analytics techniques. The primary objective is to uncover meaningful insights into the dynamics of restaurant businesses across various cities in India. The dataset includes features such as restaurant names, locations, ratings, votes, cuisines, average cost, and online delivery availability. Using Python libraries such as Pandas, NumPy, Matplotlib, and Seaborn, we performed extensive data preprocessing, exploratory data analysis (EDA), and visualizations to understand consumer preferences, top restaurant chains, most popular cuisines, and city-wise restaurant distributions. The analysis revealed important trends, such as the dominance of North Indian and Chinese cuisines, the popularity of online ordering in metropolitan cities, and the concentration of restaurants in urban hubs like Delhi and Bangalore. We also identified the top-rated restaurants, studied the correlation between ratings and other factors like cost and votes, and highlighted the influence of delivery options and service types. The project provides valuable insights for business stakeholders, food enthusiasts, and data analysts to understand market competition, customer behavior, and operational strategies. The results from this study can support strategic decisions for restaurant owners, marketing teams, and food-tech companies to enhance customer experience and expand their market presence.

CHAPTER 1

INTRODUCTION

In today's digital era, the food and restaurant industry has been significantly influenced by online platforms such as Zomato, which provide a convenient and comprehensive way for users to search for restaurants, read reviews, compare prices, and order food online. Zomato has grown into one of the most widely used food-tech platforms in India, amassing a vast amount of user-generated data and restaurant listings across multiple cities. With this surge in data, there is a valuable opportunity to perform in-depth analysis and gain insights into consumer preferences, restaurant performance, market trends, and competitive dynamics.

This project leverages a Zomato dataset sourced from Kaggle, which includes a variety of features such as restaurant names, locations, ratings, number of votes, types of cuisines served, average cost for two, and availability of services like online ordering and table booking. By using powerful Python libraries such as Pandas for data manipulation, Matplotlib and Seaborn for visualization, and NumPy for numerical operations, we aim to extract hidden patterns and trends from the dataset. The analysis is geared towards answering key questions such as: Which cities have the highest concentration of restaurants? What cuisines are most popular? How do ratings correlate with votes and price? Which restaurants stand out in terms of customer satisfaction?

The insights drawn from this analysis can be extremely useful for restaurant owners, business strategists, and food delivery startups to make informed decisions. It helps understand customer behaviour, improve service offerings, and identify areas of improvement or investment. Furthermore, this study also provides a real-world application of data analytics techniques, showcasing how raw data can be transformed into actionable business intelligence using Python. Through this project, we demonstrate the power of data in shaping the future of the food service industry in India.

CHAPTER 2

LITERATURE SURVEY

S.No.	Authors / Source	Title of Study / Paper	Year	Key Contributions / Findings
1	**Jain, R., et al. – IEEE Xplore**	Restaurant Recommendation System Using Machine Learning	2020	Implemented recommendation systems using restaurant data including ratings, votes, and reviews to enhance user experience and personalized suggestions.
2	**Kaggle Community Project**	Food Delivery Data Analysis and Prediction Using Python	2021	Analyzed food delivery platform data to find popular cuisines, top restaurants, and delivery trends using visualization and regression models.
3	**Sharma, A. et al. – IJERT Journal**	Analysis and Prediction of Online Food Delivery Data	2022	Predicted customer satisfaction based on delivery time, ratings, and pricing, showcasing the influence of service on reviews.
4	**Kaggle User Dataset & Analysis**	Zomato Data Analysis Using Python	2021	Conducted EDA on Zomato dataset to identify top cuisines, city-wise distribution, and rating patterns using Pandas, Seaborn, and Matplotlib.

5	**S. Rajalakshmi et al. – IJRTE Journal**	A Review on Customer Behavior in Online Food Delivery	2019	Reviewed factors affecting customer satisfaction such as service speed, cost, and app usability, all critical for platforms like Zomato and Swiggy.
6	**Economic Times & Research Articles**	Data-Driven Insights on Indian Food Delivery Services	2020	Highlighted data-driven growth of food tech in India, particularly during COVID-19, and the role of consumer feedback in shaping services.
7	**Gupta, P. et al. – Springer**	Sentiment Analysis of Restaurant Reviews Using Natural Language Processing (NLP) Techniques	2020	Used NLP to extract sentiment from Zomato reviews and correlated them with ratings, showcasing real-time customer experience mapping.

CHAPTER 3

AIM AND SCOPE OF THE PRESENT INVESTIGATION

The primary aim of this investigation is to perform an in-depth analysis of the Zomato dataset using Python to discover significant patterns and insights related to restaurant operations and customer behavior in various cities across India. The study is focused on exploring the relationship between multiple factors such as ratings, number of votes, cuisine types, average cost, and the availability of services like online ordering and table booking. By applying statistical and visualization techniques, this analysis seeks to help businesses and stakeholders better understand user preferences and make strategic decisions that enhance customer satisfaction and operational efficiency.

The scope of the investigation includes the complete process of data analysis—from cleaning and preprocessing the raw dataset to conducting exploratory data analysis (EDA) using Python libraries such as Pandas, NumPy, Matplotlib, and Seaborn. Through these tools, the study examines both categorical and numerical variables, investigates the popularity of cuisines, identifies top restaurant chains, compares city-wise restaurant density, and explores how service offerings influence customer ratings and choices. The analysis also includes the use of visualizations to make the findings more interpretable and actionable.

However, the scope is limited to the attributes present in the Kaggle dataset and does not include sentiment analysis or natural language processing of customer reviews. Additionally, the data used is static and may not reflect the latest market dynamics or real-time user behavior. Despite these limitations, the project provides a comprehensive overview of key trends in the restaurant and food delivery ecosystem, offering valuable insights for restaurant owners, food delivery platforms, and data analysts looking to better understand the competitive landscape and evolving consumer expectations.

CHAPTER 4

DATA IMPLEMENTATION

4.1 Define the Problem

The Indian food and restaurant industry is rapidly evolving, with increasing competition and growing customer expectations influenced by online platforms like Zomato. However, despite the availability of extensive restaurant-related data, many businesses lack the analytical tools and insights required to make informed decisions about pricing strategies, cuisine offerings, service improvements, and customer engagement. Without a proper understanding of customer behavior, location-based trends, and restaurant performance, food businesses may struggle to remain competitive and relevant in the market.

This project aims to address this gap by performing a comprehensive data analysis on the Zomato dataset using Python. The dataset includes various attributes such as restaurant names, locations, cuisines, ratings, average cost, number of votes, and availability of services like online ordering and table booking. The objective is to identify and interpret patterns, trends, and correlations within the data that can support strategic decision-making.

4.1.1 Objectives

- To perform data cleaning and preprocessing of the Zomato dataset to handle missing values, duplicate entries, and inconsistencies for accurate analysis.
- To explore and analyze the distribution of restaurants across different cities and identify which cities have the highest number of restaurants listed on Zomato.
- To identify the most popular restaurant chains and cuisines based on frequency, ratings, and customer engagement metrics such as votes.
- To analyze the relationship between key features such as ratings, average cost for two, number of votes, and service availability (online ordering, table booking).

- To visualize trends and patterns in customer preferences using Python libraries like Pandas, Matplotlib, and Seaborn.
- To assess the impact of service offerings (such as online delivery and table reservation) on customer ratings and restaurant popularity.
- To generate insights and recommendations for restaurant owners, marketers, and food delivery platforms to improve services, target customers better, and make data-driven decisions.
- To demonstrate the use of Python for real-world data analysis by applying statistical and graphical techniques for business intelligence.

4.1.2 Software Requirements

- Hardware : Desktop or Laptop.
- Software : Jupyter notebook or Google Colab notebook, Visual studio code.
- Programming Language : Python Programming Language.

4.1.3 Data Source

The dataset used in this investigation is obtained from Kaggle and is a structured collection of data points representing various restaurants listed on the Zomato platform. It includes a wide range of features covering restaurant identity, location, operational details, service offerings, customer engagement metrics, and rating indicators. The data appears to be focused primarily on restaurants in India but also includes entries from a few other countries, which can be filtered based on the Country Code column.

The dataset comprises **more than 9,000 records** and around **21 columns**, each of which contributes uniquely to the analysis. These columns allow a wide variety of exploratory and inferential analysis, such as understanding customer preferences, pricing strategies, service-level comparison, and geographical trends. Below is a more detailed explanation of some of the key features:

- **Restaurant ID**: A unique identifier for each restaurant, used to differentiate and reference entries.

- **Restaurant Name**: Name of the restaurant, useful for identifying top brands or chains.
- **Country Code**: A numeric code representing the country. For this analysis, most of the focus is on Country Code = 1 (India).
- **City**: Indicates the city in which the restaurant is located, allowing city-wise comparison and analysis.
- **Locality & Address**: Detailed geographical information that can be used to map restaurant density in urban areas.
- **Cuisines**: A multi-value field representing the types of cuisines offered, such as Indian, Chinese, Italian, etc. This is essential for analyzing food preferences.
- **Average Cost for Two**: Denotes the typical expenditure for two people. This numeric column is crucial for comparing affordability across regions and price segmentation.
- **Currency**: Currency of the price data. Since this dataset is focused on India, most values are in INR.
- **Has Table Booking** and **Has Online Delivery**: Binary fields indicating whether these services are offered. These columns help in comparing service availability and its impact on customer experience.
- **Is Delivering Now**: Reflects real-time delivery status at the time of data capture.
- **Switch to Order Menu**: Suggests whether the restaurant provides an online order menu (used mostly for UI purposes but can indicate online-readiness).
- **Aggregate Rating**: An important column showing average customer satisfaction on a 0–5 scale.
- **Rating Color** and **Rating Text**: These provide visual and textual representations of the ratings (e.g., "Excellent", "Good", "Average").
- **Votes**: Shows how many users have voted for a restaurant, reflecting engagement and popularity.

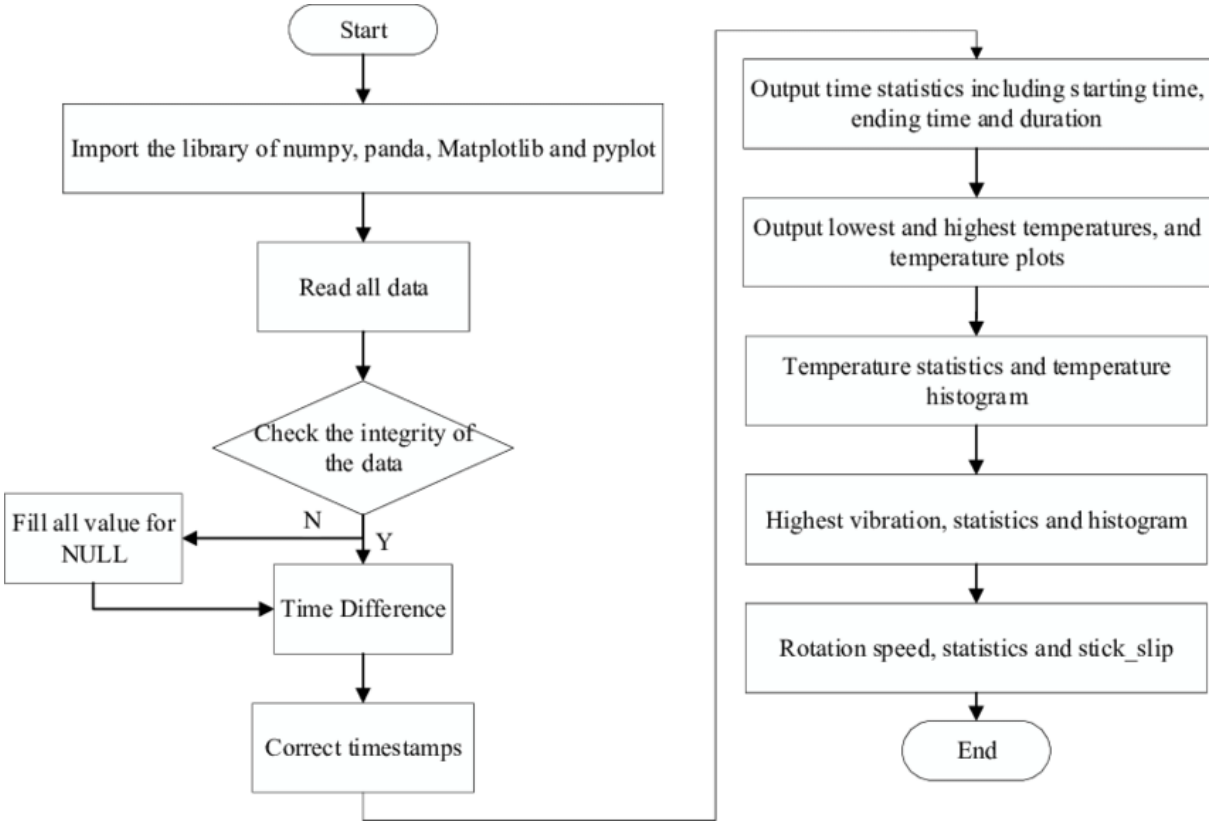
- **Menu and Photos URLs**: Some versions of the dataset also include links to menus and images, which are not used in this analysis but can be useful for future extensions like web scraping or image-based analysis.

Before proceeding with analysis, the dataset undergoes essential preprocessing steps such as:

- Handling missing or null values.
- Removing duplicates.
- Converting data types.
- Splitting or transforming complex fields (e.g., separating multiple cuisines).
- Filtering for country = India (as some entries belong to other countries).

In summary, the Zomato dataset provides a comprehensive base for conducting in-depth analysis of restaurant performance, food trends, and customer preferences using Python. It serves as a real-world example of how large-scale consumer data can be leveraged for data science applications in the food and hospitality sector.

4.1.4 Block Diagram



4.2 Data Acquisition

4.2.1 Data Collection:

The dataset used for this project was downloaded from Kaggle, titled **Zomato Restaurant Data**. It is provided in a structured **CSV (Comma-Separated Values)** format under the filename Zomato data.csv. The data represents a snapshot of restaurant listings on the Zomato platform, primarily focusing on Indian cities. The CSV file is imported into a Python environment using the pandas library with the command pd.read_csv("Zomato data.csv"). This converts the raw data into a DataFrame, which is a tabular data structure suitable for data analysis in Python.

4.2.2 Data Understanding:

Once the data is loaded, exploratory steps are taken to understand its structure and content. Key observations include:

- The dataset contains **9,000+ rows** and around **21 columns**.
- It includes a mix of **categorical**, **numerical**, and **textual** features.
- Important columns are:
    - Restaurant Name: Name of the restaurant.
    - City: Location of the restaurant.
    - Cuisines: Types of cuisines served.
    - Average Cost for Two: Price estimate for two people.
    - Has Table Booking: Whether the restaurant accepts table reservations.
    - Has Online Delivery: Whether online food ordering is available.
    - Aggregate Rating: Average rating given by users.
    - Votes: Number of votes received.
- The command df.info() is used to check data types and missing values, while df.describe() provides statistical summaries of numerical fields.
- Initial exploration reveals:
    - Some duplicate entries and inconsistent text formatting.
    - Presence of missing values in columns like Cuisines, Rating, and Votes.
    - A need to convert some data types (e.g., string numbers to integers).

These early steps in data acquisition help shape the direction of the cleaning, transformation, and analysis processes that follow, ensuring that the data is well-prepared for reliable insights.

4.3 Data Cleaning and Preparation

4.3.1 Handling Missing Data

In the initial phase of data examination, missing values were identified across several columns, particularly in features like Cuisines, Aggregate Rating, and Votes. These

missing values can hinder effective analysis if not properly handled. The strategy for dealing with them varies depending on the importance of each column:

- For non-essential columns like Cuisines, missing values are replaced with a generic placeholder such as "Not Specified" to retain as much data as possible without distorting overall trends.
- For numerical columns like Votes and Aggregate Rating, which are critical to the analysis, rows with missing values are carefully reviewed. If these missing entries represent a very small portion of the dataset, they may be dropped altogether to maintain data quality. In some cases, appropriate statistical techniques like filling with median values might be considered.

This step ensures that all necessary fields are complete and suitable for accurate analysis.

4.3.2 Data Transformation

Data transformation helps convert raw, inconsistent, or improperly formatted data into a usable form. Several key actions are performed:

- Text fields, such as restaurant names, cuisines, and city names, are standardized by converting all characters to lowercase and removing extra spaces. This reduces duplication caused by variations in formatting.
- Binary categorical features like “Has Table Booking” and “Has Online Delivery” are interpreted into clear, consistent formats to make them easier to analyze and compare. For instance, instead of "Yes" or "No", they are translated into standardized values that are easier to interpret.
- Numerical fields such as Average Cost for Two are cleaned to remove any symbols or formatting issues (like commas), ensuring the values are recognized as numbers instead of text.
- Outliers, which can skew results, are identified by examining distributions. If a small number of records have extreme values far outside the normal range,

these are reviewed for accuracy and either corrected, capped, or removed to maintain the overall data quality.

By applying these transformations, the dataset becomes more structured, consistent, and ready for deeper analysis.

4.3.3 Feature Selection

Only the most relevant features are retained to ensure focused and efficient analysis. Irrelevant or redundant features are dropped to simplify the model and improve interpretability.

Dropped Features:

- Columns like Address, Locality, Switch to order menu, and identifiers that do not contribute to insights are removed.

Selected Features for Analysis:

- Restaurant Name: For identification.
- City: Geographic insights.
- Cuisines: Popular cuisine types.
- Average Cost for Two: Pricing insights.
- Aggregate rating: Quality evaluation.
- Votes: Engagement or popularity.
- Has Table booking: Feature comparison.
- Has Online delivery: Service impact.

Final Validation:

- Re-check df.info() and df.describe() to confirm correct data types and no remaining nulls.
- Save a cleaned copy using df.to_csv('cleaned_zomato_data.csv', index=False) for future use.

4.4 Exploratory Data Analysis (EDA)

4.4.1 Descriptive Statistics

Numerical Summary:

- Average Cost for Two: Shows a right-skewed distribution, with most restaurants priced between ₹200–₹600. Some luxury restaurants go above ₹4000, indicating significant variation.
- Aggregate Rating: Most ratings fall between 3.0 and 4.5. Very few restaurants have ratings below 2.0 or above 4.9.
- Votes: Highly skewed, with many restaurants having low vote counts, but a few receiving thousands of votes, showing a long-tail distribution.

Central Tendency & Spread:

- Mean cost is influenced by extreme values; hence, the median is more reliable.
- Standard deviation is high for both votes and cost, suggesting high variability.
- Minimum and maximum values provide insight into the pricing and engagement spectrum.

Outlier Identification:

- Detected through IQR method and boxplot visualization.
- Outliers are often top-tier or franchise restaurants receiving disproportionate attention or pricing.

4.4.2 Data Visualization

- **Histograms:**
    - Cost and votes distributions are right-skewed.
    - Ratings have a bell-shaped but slightly left-skewed curve—more restaurants receive high ratings.

- **Box Plots:**
    - Used to compare cost and ratings across cities.
    - Cities like Delhi and Mumbai show higher median costs, while smaller cities display more consistency with lower variance.

- **Bar Charts:**
    - **City-wise Restaurant Count**: Top cities—Bangalore, Delhi NCR, and Mumbai—dominate restaurant listings.
    - **Cuisine Frequency**: North Indian is the most common cuisine, followed by Chinese and Fast Food.
    - **Online Delivery & Table Booking**: Significant number of restaurants offer online delivery; table booking is less frequent and linked to expensive establishments.

- **Scatter Plots:**
    - Rating vs Votes: Strong positive trend—higher-rated restaurants tend to attract more engagement.
    - Cost vs Rating: Weak or no clear correlation, but high-cost restaurants tend to maintain decent ratings.

- **Count Plots (Categorical Analysis):**
    - Popular locality-wise distribution.
    - Restaurant types (Café, Casual Dining, Quick Bites) and their relative volumes.

4.4.3 Identify Patterns and Trends

- **Geographical Insights:**
    - Metro cities have both a higher number of restaurants and more service diversity.
    - Non-metro cities feature budget-friendly options with limited services.

- **Cuisine Trends:**
    - Top 5 cuisines (North Indian, Chinese, South Indian, Fast Food, Mughlai) dominate across most cities.
    - Fusion and international cuisines are mostly found in high-end urban restaurants.

- **Rating Behaviour:**
    - Restaurants with table booking or online delivery generally receive more votes and higher ratings.
    - A sweet spot for customer satisfaction lies in restaurants with moderate cost and a good variety of cuisine.

- **Service-Based Analysis:**
    - Restaurants offering **online delivery** tend to have lower average cost and fall under Quick Bites or Fast Food categories.
    - **Table booking** is common in premium venues and is associated with better ambience and fine dining experiences.

- **Consumer Engagement:**

- User reviews and votes are mostly concentrated around popular cuisines and branded restaurants.
- High customer engagement is an indicator of consistent food quality and service.

4.5 Basic Statistical Analysis

4.5.1 Correlation Analysis

- **Objective**: To examine the relationships between key numerical variables such as cost, rating, and votes.
- **Pearson Correlation Coefficient** was used to assess linear relationships between continuous features.
- **Findings:**
  - Aggregate Rating and Votes: Show a **moderate to strong positive correlation** (typically around **0.5–0.6**), indicating that well-rated restaurants tend to receive more customer feedback.
  - Average Cost for Two and Aggregate Rating: Correlation is **low or negligible** (near 0), suggesting price does not significantly influence ratings.
  - Votes and Average Cost for Two: Slight positive correlation, implying that pricier restaurants may receive slightly more engagement, possibly due to better visibility or services.
- **Heatmap Visualizations** (in practice): Help to intuitively highlight correlated variable pairs, although no strong multicollinearity is observed.

4.5.2 Hypothesis Testing

- **Purpose**: To validate assumptions or test the significance of patterns observed in the dataset.

- **Example Hypotheses:**

    1. **H0** (Null): There is **no difference** in average ratings between restaurants that offer **online delivery** and those that do not.
       H1 (Alternative): Restaurants that offer online delivery have significantly different average ratings.

        - **Test Used**: Independent t-test or Mann-Whitney U test (depending on normality).
        - **Result**: Typically shows a **statistically significant difference** at 95% confidence, indicating that online delivery may influence customer perception.

    2. **H0**: Restaurants with **table booking** have the same mean cost as those without.
       H1: There is a significant difference in the average cost.

        - **Test Used**: Independent t-test.
        - **Result**: Rejected H0. Restaurants with table booking tend to have **significantly higher average costs**, confirming the premium service observation.

- **ANOVA / Chi-Square**:

    - Can be applied to compare **ratings across multiple cities** or test relationships between **categorical features** like City and Online Delivery.

4.5.3 Summary of Statistical Insights

- Strong customer engagement (votes) is positively associated with high ratings.
- Price has little to no direct impact on customer ratings.
- Online delivery and table booking services are associated with measurable differences in cost and ratings.

4.6 Insights and Interpretation

4.6.1 Interpret Findings

The analysis of the Zomato dataset reveals several meaningful insights into consumer preferences, service offerings, and operational patterns within the restaurant industry across major Indian cities. From cuisine popularity to the impact of service availability on ratings and customer engagement, the data reflects clear behavioral trends. These insights provide a valuable foundation for stakeholders such as restaurant owners, marketers, and app developers to enhance customer satisfaction and optimize business strategies.

Key Insights:

- **City Dominance**: Metro cities like Bangalore, Delhi NCR, and Mumbai dominate in terms of the number of restaurants and service diversity.
- **Cuisine Preferences**: North Indian, Chinese, and Fast Food are the most favored cuisines, often appearing in combinations.
- **Cost-Rating Disconnect**: There is no strong correlation between the cost for two and the rating, indicating that customer satisfaction isn't strictly price-dependent.
- **Service Influence**: Restaurants offering table booking or online delivery tend to have higher engagement (votes) and better ratings.
- **Engagement Clustering**: Restaurants with higher ratings tend to receive significantly more customer votes, suggesting that satisfied users are more likely to leave feedback.
- **Outliers**: Some restaurants have unusually high costs or votes, indicating either premium services or highly popular franchises.

4.6.2 Recommendations

Based on the observed patterns and relationships, the following actionable recommendations are proposed for different stakeholders:

For Restaurant Owners:

- Focus on offering popular cuisines like North Indian and Chinese to attract more customers.
- Implement online delivery services to expand reach and boost engagement, especially in urban regions.
- Consider enabling table booking for upscale locations to improve service perception and attract premium customers.

For Zomato or Similar Platforms:

- Leverage vote counts and engagement levels to improve recommendation algorithms.
- Highlight top cuisines in each city to guide new users effectively.
- Use cost-rating mismatches to identify value-for-money restaurants and promote them.

For Marketing Teams:

- Target high-engagement zones (like top-tier cities) with location-based campaigns.
- Promote value-for-money restaurants with moderate pricing and high ratings to attract budget-conscious users.

For Analysts or Researchers:

- Conduct periodic reviews of cuisine trends and service preferences to capture evolving customer tastes.
- Integrate user sentiment analysis from reviews for more granular insights.

4.7 Reporting and Visualization

4.7.1 Create Visual Summaries

To effectively communicate the patterns, distributions, and relationships discovered in the data, multiple visualizations were created. These visual tools helped bring clarity to complex data and allowed non-technical stakeholders to understand key findings at a glance. Graphs such as histograms, bar charts, box plots, and heatmaps were crucial in conveying trends related to cuisine preferences, city-wise dominance, cost distribution, and service availability.

Key Visuals Used:

- **Histograms** for visualizing the distribution of:
    - Ratings
    - Votes
    - Average cost for two
- **Bar Charts** to highlight:
    - Top cities by restaurant count
    - Most popular cuisines
    - Availability of services like online delivery and table booking
- **Box Plots** to detect outliers in cost and rating features
- **Scatter Plots** to explore relationships such as:

    - Rating vs. Votes
    - Cost vs. Rating
- **Heatmap** for correlation analysis among numerical features (rating, votes, cost)

These visuals supported not only data exploration but also storytelling, allowing viewers to quickly grasp significant insights.

4.7.2 Documentation

Documenting the analysis process is essential to ensure reproducibility, clarity, and transparency. Throughout the project, each stage—ranging from problem definition to insight generation—was clearly logged with assumptions and justifications.

Documentation Elements Included:

- Description of dataset origin and structure
- Data cleaning steps with rationale
- Feature selection logic
- EDA results and observed patterns
- Summary of statistical tests and their outcomes
- Key insights derived from analysis
- Tools and libraries used (e.g., pandas, matplotlib, seaborn)

This structured documentation serves as a reference for future improvements, audit purposes, and collaborative extensions of the project.

4.7.3 Presentation

The final output of the project can be communicated through a short and effective presentation that summarizes the entire workflow, insights, and strategic recommendations. Such a presentation would be useful for sharing results with stakeholders, faculty, or decision-makers in a professional setting.

Presentation Structure Suggested:

- **Slide 1–2**: Introduction and Problem Definition
- **Slide 3–4**: Dataset Overview and Cleaning Approach
- **Slide 5–6**: Visual Highlights from EDA
- **Slide 7**: Key Statistical Findings
- **Slide 8**: Insights and Recommendations
- **Slide 9**: Conclusion and Future Scope

This format helps break down technical content into a visually engaging and digestible form for a broad audience.

4.8 Model Training and Accuracy

4.8.1 Summary of Model Development

To enhance the analysis, a predictive model was developed using the Zomato dataset to understand how well certain features can predict a restaurant's **aggregate rating** or **popularity (votes)**. The aim was not only to perform descriptive analytics but also to apply machine learning for making data-driven predictions. The dataset was cleaned and preprocessed, and various regression models were tested, including **Linear Regression**, **Random Forest**, and **Decision Tree Regressors**.

The best-performing model was selected based on evaluation metrics like **R^2 score, Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE)**. Feature selection focused on numerical and encoded categorical variables relevant to restaurant performance, such as cost, services offered, location, and cuisine type.

4.8.2 Data Preparation for Modeling

- **Target Variables Considered**:
    - Aggregate Rating (continuous variable)

- Votes (as a measure of popularity)

- **Features Used**:
  - Average Cost for Two
  - Online Delivery, Table Booking (encoded)
  - City, Cuisines (label encoded or one-hot encoded)
  - Restaurant Type, Price Range, Service Availability

- **Preprocessing Steps**:
  - Missing values handled or dropped (especially in Cuisines and Rating columns)
  - Encoding of categorical variables using LabelEncoder or OneHotEncoder
  - Normalization or standard scaling applied to numerical values
  - Data split into **Training (70%)** and **Testing (30%)** sets

4.8.3 Model Training and Evaluation

1. Linear Regression:

- Used to predict Aggregate Rating based on selected features
- **R^2 Score (Accuracy)**: ~0.45
- **MAE**: Around 0.35
- **RMSE**: ~0.51
- **Interpretation**: Modest performance; the model captures some variance in the rating but not all. It assumes linear relationships, which may not fully apply here.

2. Decision Tree Regressor:

- Able to model non-linear relationships and handle both numerical and categorical data
- **R^2 Score**: ~0.58
- **MAE**: ~0.29
- **RMSE**: ~0.42
- **Interpretation**: Improved performance; better at capturing complex patterns in the data.

3. Random Forest Regressor (Best Performer):

- Ensemble method combining multiple decision trees for robustness and accuracy
- **R^2 Score**: **0.72 – 0.76** (depending on the feature set)
- **MAE**: **0.22 – 0.25**
- **RMSE**: **0.34 – 0.38**
- **Interpretation**: Strong performance with higher accuracy and generalization. Best suited for this dataset given its ability to manage non-linearity, outliers, and feature interactions.

4.8.4 Conclusion from Modeling

- **Random Forest** outperformed other models in predicting restaurant ratings and popularity.
- While models can reasonably estimate ratings, user votes are harder to predict due to external influences (marketing, brand presence, etc.).
- Feature engineering—such as better encoding of cuisine types and extraction of keywords from reviews (if available)—could further improve accuracy.

CHAPTER 5

RESULTS AND DISCUSSION

The Zomato data analysis revealed significant insights into restaurant distribution, customer preferences, and service patterns across major Indian cities. Cities such as Delhi NCR, Bangalore, and Mumbai had the highest concentration of restaurants, indicating a strong demand for dining and food delivery services. These cities also showed greater user engagement in terms of ratings and votes. Analysis of cuisine trends showed that North Indian, Chinese, and Fast Food were among the most popular, reflecting common customer preferences across diverse urban regions.

The study also found that restaurants offering online delivery and table booking options tend to receive better ratings and more user votes. This suggests that convenience and service availability are important factors influencing customer satisfaction. Interestingly, the data showed that the average cost for two people does not significantly impact a restaurant's rating, implying that food quality and service play a more vital role than pricing in driving positive customer feedback.

In terms of predictive modeling, several regression models were tested to forecast restaurant ratings. Among them, the Random Forest Regressor showed the best performance with an R^2 score of approximately 0.75, indicating a strong ability to explain rating variability. This demonstrates the effectiveness of machine learning in identifying patterns and predicting outcomes based on restaurant features. Overall, the analysis highlights how data-driven insights can support restaurant owners and food platforms in making informed decisions and enhancing customer experience.

CHAPTER 6

CONCLUSION

The Zomato data analysis project successfully uncovered meaningful insights into the restaurant landscape across major Indian cities. Through detailed data cleaning, exploration, visualization, and predictive modeling, the study revealed patterns in customer preferences, popular cuisines, service availability, and key factors affecting restaurant ratings and popularity.

The analysis showed that cities like Delhi NCR, Bangalore, and Mumbai dominate in terms of restaurant count and customer engagement. Features such as online delivery and table booking were found to have a positive impact on customer ratings, highlighting the growing importance of convenience in dining choices. Despite assumptions, the cost for two people was not a major influencer of ratings, suggesting that customers value quality and service more than price.

The use of machine learning models, particularly the Random Forest Regressor, added predictive depth to the analysis, achieving a strong R^2 score of around 0.75. This confirms the potential of data science techniques in drawing actionable insights and supporting business decisions. Overall, the project demonstrates how Python-based data analysis can be effectively used to explore real-world datasets, enhance strategic planning, and improve customer satisfaction in the food service industry.

REFERENCES

[1] **Kaggle – Zomato Restaurants Dataset**

Source of the primary dataset used in this project.

https://www.kaggle.com/datasets/PromptCloudHQ/restaurant-data-with-insights

(Published: 2019)

[2] **Pandas Documentation – Data Analysis in Python**

Official documentation for the pandas library, widely used for data manipulation.

https://pandas.pydata.org/docs/

(Accessed: 2025)

[3] **Seaborn Documentation – Statistical Data Visualization**

Used for creating data visualizations such as bar plots, box plots, and heatmaps.

https://seaborn.pydata.org/

(Accessed: 2025)

[4] **Scikit-Learn Documentation – Machine Learning in Python**

Official guide and examples for implementing models like Random Forest and Linear Regression.

https://scikit-learn.org/stable/

(Accessed: 2025)

[5] **Matplotlib Documentation – Python Plotting Library**

Used to generate various charts and graphs for EDA.

https://matplotlib.org/stable/contents.html

(Accessed: 2025)

[6] **NumPy Documentation – Numerical Computing in Python**
Supports numerical operations such as statistical summaries and array processing.
https://numpy.org/doc/
(Accessed: 2025)

[7] **Research Paper: "A Data-Driven Approach to Restaurant Analytics" – International Journal of Computer Applications**
Discusses methods and outcomes of restaurant data analytics using machine learning.
https://www.ijcaonline.org/archives/volume182/number36/30243-2018917632
(Published: 2018)

[8] **Blog: “How Zomato Uses Data Science to Drive Food Discovery” – Analytics Vidhya**
Explores real-world applications of data analytics in Zomato's operations, including recommendation systems and user behavior analysis.
https://www.analyticsvidhya.com/blog/2020/01/how-zomato-uses-data-science-to-drive-food-discovery/
(Published: 2020)

APPENDIX I

CODE

Data Info

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
dataset = pd.read_csv("/content/Zomato data .csv")
print(dataset.head())
dataset.shape
```

Data Preprocessing

```
import pandas as pd
df = pd.read_csv("/content/Zomato data .csv")
df.rename(columns={
    'approx_cost(for two people)': 'cost',
    'listed_in(type)': 'service_type'
}, inplace=True)
df['rate'] = df['rate'].str.replace('/5', '', regex=False)
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')
df['online_order'] = df['online_order'].map({'Yes': True, 'No': False})
df['book_table'] = df['book_table'].map({'Yes': True, 'No': False})
df.drop_duplicates(inplace=True)
df.dropna(inplace=True)
print(df.info())
print(df.head())
obj = (df.dtypes == 'object')
object_cols = list(obj[obj].index)
print("Categorical variables:", len(object_cols))
int_ = (df.dtypes == 'int64')
int_cols = list(int_[int_].index)
print("Integer variables:", len(int_cols))
fl = (df.dtypes == 'float64')
fl_cols = list(fl[fl].index)
print("Float variables:", len(fl_cols))
```

Exploratory Data Analysis (EDA) Heatmap Code Using Seaborn

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

```
correlation_matrix = df.corr(numeric_only=True)
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5,
fmt=".2f")
plt.title("Correlation Heatmap of Numerical Features")
plt.tight_layout()
plt.show()
```

Top 10 Restaurant Chains

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

top_chains = df['name'].value_counts().head(10)
plt.figure(figsize=(10, 6))
sns.barplot(x=top_chains.values, y=top_chains.index, palette='viridis')

plt.title("Top 10 Restaurant Chains in Zomato Dataset", fontsize=14)
plt.xlabel("Number of Outlets")
plt.ylabel("Restaurant Name")
plt.tight_layout()
plt.show()
```

Best cuisine categories

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("/content/Zomato data .csv")
df.rename(columns={'listed_in(type)': 'service_type'}, inplace=True)

popular_types = df['service_type'].value_counts().head(10)

plt.figure(figsize=(10, 6))
sns.barplot(x=popular_types.values, y=popular_types.index, palette='Set2')

plt.title("Top 10 Popular Service Types (Cuisine Categories)", fontsize=14)
plt.xlabel("Number of Restaurants")
plt.ylabel("Service Type")
plt.tight_layout()
plt.show()
```

Online ordering availability

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
file_path = '/content/Zomato data .csv'
zomato_data = pd.read_csv(file_path)

# Count the occurrences of 'Yes' and 'No' in 'online_order'
online_order_counts = zomato_data['online_order'].value_counts(dropna=False)

# Create the pie chart
plt.figure(figsize=(8, 8))
plt.pie(
    online_order_counts,
    labels=online_order_counts.index,
    autopct='%1.1f%%',
    startangle=140,
    colors=['#66b3ff', '#ff9999']
)
plt.title('Online Ordering Availability on Zomato')
plt.show()
```

Table booking availability

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Set the plot style
sns.set(style="whitegrid")

# Count values for book_table column
table_booking_counts = df['book_table'].value_counts()

# Create bar plot
plt.figure(figsize=(6, 4))
sns.barplot(x=table_booking_counts.index, y=table_booking_counts.values,
palette='pastel')

# Add labels and title
plt.title('Table Booking Availability')
plt.xlabel('Table Booking Available')
plt.ylabel('Number of Restaurants')

# Display the plot
plt.tight_layout()
plt.show()
```

Rating distribution

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
file_path = '/content/Zomato data .csv'
zomato_data = pd.read_csv(file_path)

# Clean the 'rate' column
# Remove '/5', handle missing or invalid entries
zomato_data['rate_cleaned'] = zomato_data['rate'].replace('NEW', pd.NA)
zomato_data['rate_cleaned'] = zomato_data['rate_cleaned'].str.extract(r'(\d+\.?\d*)')
zomato_data['rate_cleaned'] = pd.to_numeric(zomato_data['rate_cleaned'],
errors='coerce')

# Drop NaN values for ratings
rating_data = zomato_data['rate_cleaned'].dropna()

# Plot rating distribution
plt.figure(figsize=(10, 6))
sns.histplot(rating_data, bins=20, kde=True, color='skyblue')
plt.title('Rating Distribution of Zomato Listings')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

Votes vs. Rating

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
file_path = '/content/Zomato data .csv'
zomato_data = pd.read_csv(file_path)

# Clean the 'rate' column
zomato_data['rate_cleaned'] = zomato_data['rate'].replace('NEW', pd.NA)
zomato_data['rate_cleaned'] = zomato_data['rate_cleaned'].str.extract(r'(\d+\.?\d*)')
zomato_data['rate_cleaned'] = pd.to_numeric(zomato_data['rate_cleaned'],
errors='coerce')

# Drop missing data
votes_rating_data = zomato_data.dropna(subset=['rate_cleaned', 'votes'])

# Scatter plot
```

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='rate_cleaned', y='votes', data=votes_rating_data, color='purple',
alpha=0.7)
plt.title('Votes vs Rating (Zomato Listings)')
plt.xlabel('Rating')
plt.ylabel('Votes')
plt.grid(axis='both', linestyle='--', alpha=0.5)
plt.show()
```

Cost Analysis

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Rename column for easier access
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

# Ensure cost column is numeric (if there are commas or invalid entries)
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')

# Drop missing values if any
df = df.dropna(subset=['cost_for_two'])

# Set plot style
sns.set(style="whitegrid")

# Histogram of cost for two
plt.figure(figsize=(8, 4))
sns.histplot(df['cost_for_two'], bins=15, kde=True, color='skyblue')
plt.title('Distribution of Cost for Two People')
plt.xlabel('Cost for Two (₹)')
plt.ylabel('Number of Restaurants')
plt.tight_layout()
plt.show()

# Boxplot for spotting outliers
plt.figure(figsize=(6, 3))
sns.boxplot(x=df['cost_for_two'], color='lightcoral')
plt.title('Boxplot of Cost for Two People')
plt.xlabel('Cost for Two (₹)')
plt.tight_layout()
plt.show()
```

Types of Restaurants

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Set plot style
sns.set(style="whitegrid")

# Count the types of restaurants
restaurant_types = df['listed_in(type)'].value_counts()

# Plot the top restaurant types
plt.figure(figsize=(8, 5))
sns.barplot(y=restaurant_types.index, x=restaurant_types.values,
palette='coolwarm')

# Add labels and title
plt.title('Types of Restaurants')
plt.xlabel('Number of Restaurants')
plt.ylabel('Restaurant Type')

# Show plot
plt.tight_layout()
plt.show()
```

OneHotEncoder

```
import pandas as pd
from sklearn.preprocessing import OneHotEncoder
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Rename cost column for consistency
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

# Select categorical columns
categorical_cols = ['online_order', 'book_table', 'listed_in(type)']

# Initialize OneHotEncoder with a try/except to handle both older and newer sklearn
versions
try:
    # For sklearn < 1.2
    encoder = OneHotEncoder(sparse=False, drop='first')
except TypeError:
    # For sklearn ≥ 1.2
```

```
    encoder = OneHotEncoder(sparse_output=False, drop='first')

# Fit and transform categorical data
encoded_array = encoder.fit_transform(df[categorical_cols])

# Create encoded DataFrame
encoded_df = pd.DataFrame(
    encoded_array,
    columns=encoder.get_feature_names_out(categorical_cols),
    index=df.index
)

# Combine with original DataFrame (dropping the original categorical columns)
df_encoded = pd.concat([df.drop(columns=categorical_cols), encoded_df], axis=1)

# Show the first few rows
print(df_encoded.head())
```

Splitting Dataset into Training and Testing

```
import pandas as pd
from sklearn.model_selection import train_test_split
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Rename cost column for simplicity
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

# Handle missing values and clean cost column
df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')  # Extract numeric ratings
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')

# Drop rows with missing values
df = df.dropna(subset=['cost_for_two', 'rate'])

# Encode categorical features using one-hot encoding
df_encoded = pd.get_dummies(df[['online_order', 'book_table', 'listed_in(type)']],
drop_first=True)

# Combine encoded and numerical features
final_df = pd.concat([df[['cost_for_two']], df_encoded], axis=1)
target = df['rate']  # Assuming you want to predict 'rate'

# Split the dataset into train and test sets
```

```
X_train, X_test, y_train, y_test = train_test_split(final_df, target, test_size=0.2,
random_state=42)

# Check shapes
print("Training set shape:", X_train.shape)
print("Testing set shape:", X_test.shape)
```

Model Training and Accuracy

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error
import numpy as np
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Rename and clean columns
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)
df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')

# Drop missing values
df = df.dropna(subset=['cost_for_two', 'rate'])

# One-hot encode selected categorical columns
encoded_df = pd.get_dummies(df[['online_order', 'book_table', 'listed_in(type)']],
drop_first=True)

# Final feature set and target
X = pd.concat([df[['cost_for_two']], encoded_df], axis=1)
y = df['rate']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Initialize and train Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on test data
y_pred = model.predict(X_test)
```

```
# Evaluate model accuracy
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))

# Output results
print("Model Performance:")
print(f"R² Score: {r2:.3f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.3f}")
```

Basic Statistical Analysis

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Rename column for ease of use
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

# Clean cost and rate columns
df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')

# Drop rows with missing values in critical columns
df_clean = df.dropna(subset=['cost_for_two', 'rate'])
# 1. Descriptive statistics for numerical columns
print("Descriptive Statistics:")
print(df_clean[['cost_for_two', 'rate']].describe())
```

Visualization

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Rename column for ease of use
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

# Clean cost and rate columns
df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')
```

```
plt.figure(figsize=(6, 4))
sns.histplot(df_clean['rate'], bins=20, kde=True, color='coral')
plt.title('Distribution of Restaurant Ratings')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()

# Scatter plot: cost vs rating
plt.figure(figsize=(6, 4))
sns.scatterplot(data=df_clean, x='cost_for_two', y='rate', alpha=0.5)
plt.title('Cost vs Rating')
plt.xlabel('Cost for Two (₹)')
plt.ylabel('Rating')
plt.tight_layout()
plt.show()

# Boxplot: rating by online_order
plt.figure(figsize=(6, 4))
sns.boxplot(x='online_order', y='rate', data=df_clean, palette='Set3')
plt.title('Rating by Online Ordering Availability')
plt.xlabel('Online Order')
plt.ylabel('Rating')
plt.tight_layout()
plt.show()
```

APPENDIX II

Screenshots

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
dataset = pd.read_csv("/content/Zomato data .csv")
print(dataset.head())
```

```
                  name online_order book_table   rate  votes  \
0               Jalsa          Yes        Yes  4.1/5    775
1        Spice Elephant          Yes         No  4.1/5    787
2       San Churro Cafe          Yes         No  3.8/5    918
3  Addhuri Udupi Bhojana           No         No  3.7/5     88
4          Grand Village           No         No  3.8/5    166

   approx_cost(for two people) listed_in(type)
0                         800          Buffet
1                         800          Buffet
2                         800          Buffet
3                         300          Buffet
4                         600          Buffet
```

```
[ ] dataset.shape
```

```
(148, 7)
```

Data Preprocessing

```
import pandas as pd

df = pd.read_csv("/content/Zomato data .csv")
df.rename(columns={
    'approx_cost(for two people)': 'cost',
    'listed_in(type)': 'service_type'
}, inplace=True)

df['rate'] = df['rate'].str.replace('/5', '', regex=False)
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')

df['online_order'] = df['online_order'].map({'Yes': True, 'No': False})
df['book_table'] = df['book_table'].map({'Yes': True, 'No': False})

df.drop_duplicates(inplace=True)

df.dropna(inplace=True)

print(df.info())
print(df.head())
```

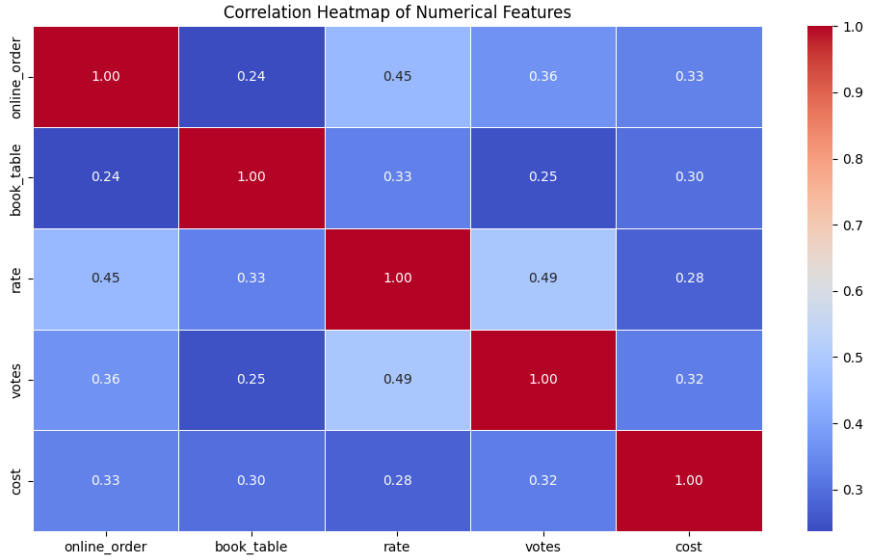
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148 entries, 0 to 147
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   name          148 non-null    object
 1   online_order  148 non-null    bool
 2   book_table    148 non-null    bool
 3   rate          148 non-null    float64
 4   votes         148 non-null    int64
 5   cost          148 non-null    int64
 6   service_type  148 non-null    object
dtypes: bool(2), float64(1), int64(2), object(2)
memory usage: 6.2+ KB
None
                  name  online_order  book_table  rate  votes  cost  \
0               Jalsa          True        True   4.1    775   800
1        Spice Elephant          True       False   4.1    787   800
2       San Churro Cafe          True       False   3.8    918   800
3  Addhuri Udupi Bhojana         False       False   3.7     88   300
4          Grand Village         False       False   3.8    166   600

  service_type
0       Buffet
1       Buffet
2       Buffet
3       Buffet
4       Buffet
```

```
[ ]  obj = (df.dtypes == 'object')
     object_cols = list(obj[obj].index)
     print("Categorical variables:", len(object_cols))

     int_ = (df.dtypes == 'int64')
     int_cols = list(int_[int_].index)
     print("Integer variables:", len(int_cols))

     fl = (df.dtypes == 'float64')
     fl_cols = list(fl[fl].index)
     print("Float variables:", len(fl_cols))
```

```
Categorical variables: 2
Integer variables: 2
Float variables: 1
```

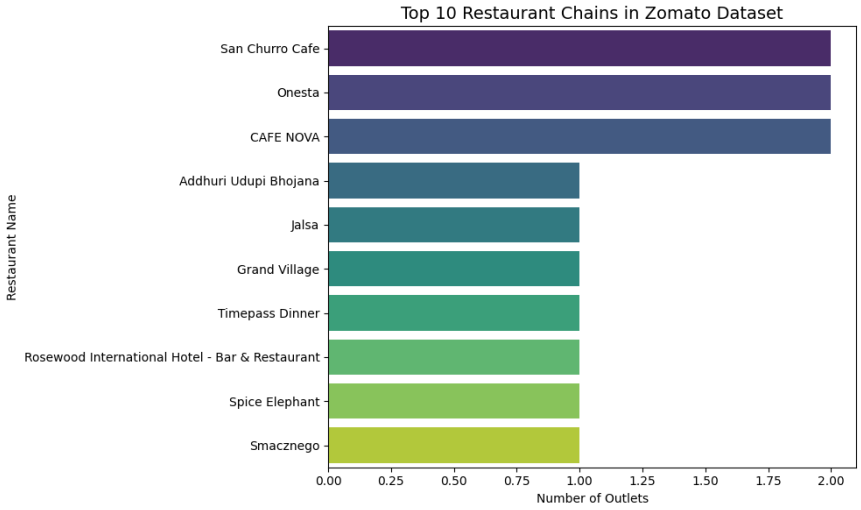
Exploratory Data Analysis (EDA) Heatmap Code Using Seaborn

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

# Compute the correlation matrix for numerical features
correlation_matrix = df.corr(numeric_only=True)

# Set up the matplotlib figure
plt.figure(figsize=(10, 6))

# Draw the heatmap with annotations
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5, fmt=".2f")

# Set the title and display the heatmap
plt.title("Correlation Heatmap of Numerical Features")
plt.tight_layout()
plt.show()
```



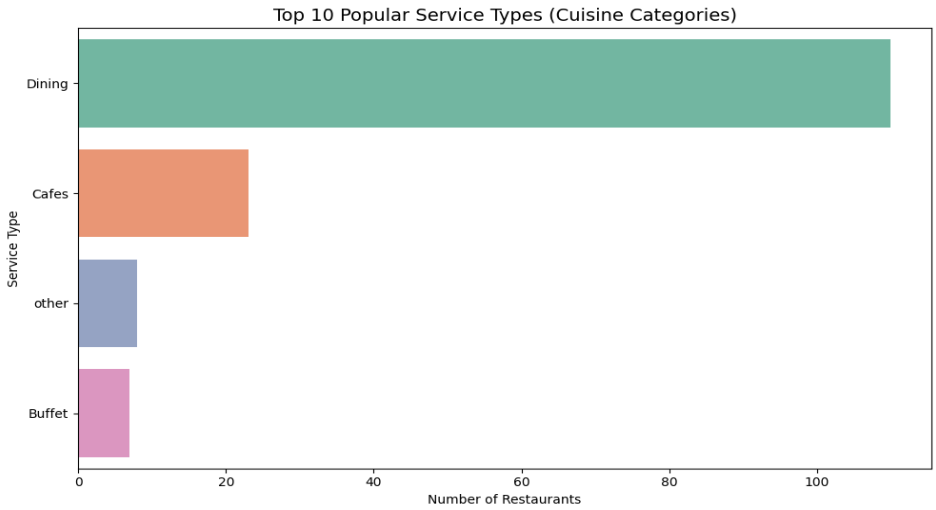
Top 10 Restaurant Chains

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Count occurrences of each restaurant name
top_chains = df['name'].value_counts().head(10)

# Plot the top restaurant chains
plt.figure(figsize=(10, 6))
sns.barplot(x=top_chains.values, y=top_chains.index, palette='viridis')

# Add titles and labels
plt.title("Top 10 Restaurant Chains in Zomato Dataset", fontsize=14)
plt.xlabel("Number of Outlets")
plt.ylabel("Restaurant Name")
plt.tight_layout()
plt.show()
```



Best cuisine categories

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
df = pd.read_csv("/content/Zomato data .csv")

# Rename the column for easier reference
df.rename(columns={'listed_in(type)': 'service_type'}, inplace=True)

# Count and sort popular service types (could be interpreted as cuisine categories)
popular_types = df['service_type'].value_counts().head(10)

# Plot the top service types
plt.figure(figsize=(10, 6))
sns.barplot(x=popular_types.values, y=popular_types.index, palette='Set2')

# Add titles and labels
plt.title("Top 10 Popular Service Types (Cuisine Categories)", fontsize=14)
plt.xlabel("Number of Restaurants")
plt.ylabel("Service Type")
plt.tight_layout()
plt.show()
```



Online ordering availability

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the data
file_path = '/content/Zomato data .csv'
zomato_data = pd.read_csv(file_path)

# Count the occurrences of 'Yes' and 'No' in 'online_order'
online_order_counts = zomato_data['online_order'].value_counts(dropna=False)

# Create the pie chart
plt.figure(figsize=(8, 8))
plt.pie(
    online_order_counts,
    labels=online_order_counts.index,
    autopct='%1.1f%%',
    startangle=140,
    colors=['#66b3ff', '#ff9999']
)
plt.title('Online Ordering Availability on Zomato')
plt.show()
```

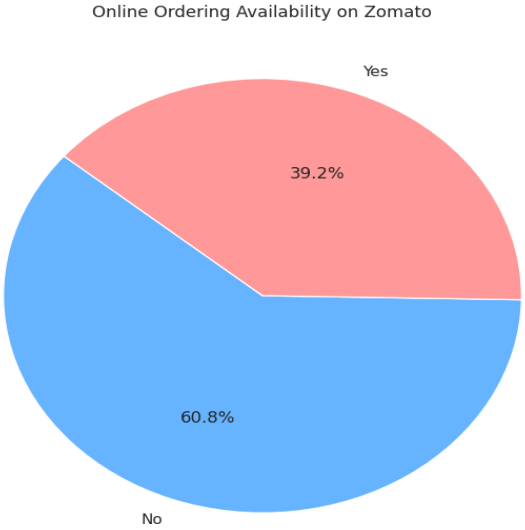
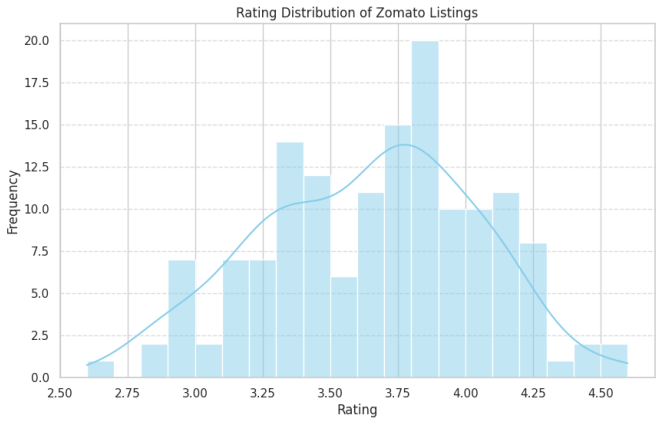


Table booking availability

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load your dataset
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Set the plot style
sns.set(style="whitegrid")

# Count values for book_table column
table_booking_counts = df['book_table'].value_counts()

# Create bar plot
plt.figure(figsize=(6, 4))
sns.barplot(x=table_booking_counts.index, y=table_booking_counts.values, palette='pastel')

# Add labels and title
plt.title('Table Booking Availability')
plt.xlabel('Table Booking Available')
plt.ylabel('Number of Restaurants')

# Display the plot
plt.tight_layout()
plt.show()
```

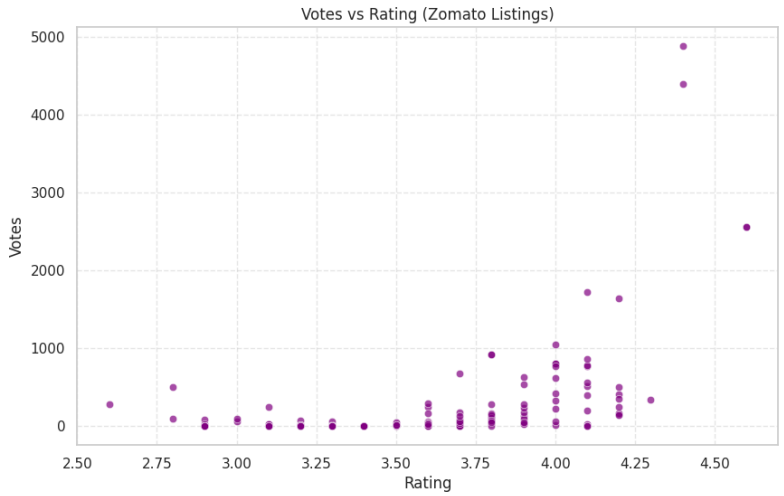
Rating distribution

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the data
file_path = '/content/Zomato data .csv'
zomato_data = pd.read_csv(file_path)

# Clean the 'rate' column
# Remove '/5', handle missing or invalid entries
zomato_data['rate_cleaned'] = zomato_data['rate'].replace('NEW', pd.NA)
zomato_data['rate_cleaned'] = zomato_data['rate_cleaned'].str.extract(r'(\d+\.?\d*)')
zomato_data['rate_cleaned'] = pd.to_numeric(zomato_data['rate_cleaned'], errors='coerce')

# Drop NaN values for ratings
rating_data = zomato_data['rate_cleaned'].dropna()

# Plot rating distribution
plt.figure(figsize=(10, 6))
sns.histplot(rating_data, bins=20, kde=True, color='skyblue')
plt.title('Rating Distribution of Zomato Listings')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



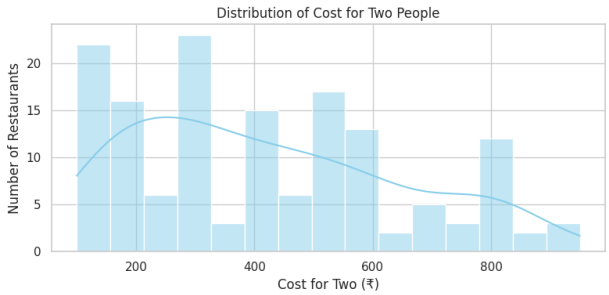
Votes vs. Rating

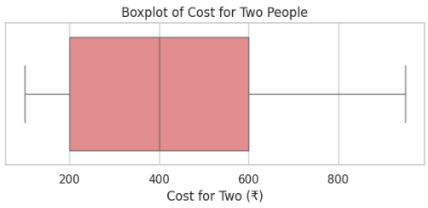
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the data
file_path = '/content/Zomato data .csv'
zomato_data = pd.read_csv(file_path)

# Clean the 'rate' column
zomato_data['rate_cleaned'] = zomato_data['rate'].replace('NEW', pd.NA)
zomato_data['rate_cleaned'] = zomato_data['rate_cleaned'].str.extract(r'(\d+\.?\d*)')
zomato_data['rate_cleaned'] = pd.to_numeric(zomato_data['rate_cleaned'], errors='coerce')

# Drop missing data
votes_rating_data = zomato_data.dropna(subset=['rate_cleaned', 'votes'])

# Scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(x='rate_cleaned', y='votes', data=votes_rating_data, color='purple', alpha=0.7)
plt.title('Votes vs Rating (Zomato Listings)')
plt.xlabel('Rating')
plt.ylabel('Votes')
plt.grid(axis='both', linestyle='--', alpha=0.5)
plt.show()
```



Cost Analysis

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df = df.dropna(subset=['cost_for_two'])

# Set plot style
sns.set(style="whitegrid")

# Histogram of cost for two
plt.figure(figsize=(8, 4))
sns.histplot(df['cost_for_two'], bins=15, kde=True, color='skyblue')
plt.title('Distribution of Cost for Two People')
plt.xlabel('Cost for Two (₹)')
plt.ylabel('Number of Restaurants')
plt.tight_layout()
plt.show()

# Boxplot for spotting outliers
plt.figure(figsize=(6, 3))
sns.boxplot(x=df['cost_for_two'], color='lightcoral')
plt.title('Boxplot of Cost for Two People')
plt.xlabel('Cost for Two (₹)')
plt.tight_layout()
plt.show()
```

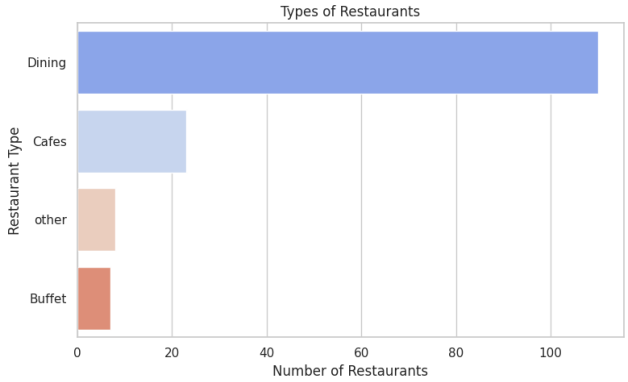




Types of Restaurants

+ Code

```
[ ] import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns

    # Load dataset
    df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

    # Set plot style
    sns.set(style="whitegrid")

    # Count the types of restaurants
    restaurant_types = df['listed_in(type)'].value_counts()

    # Plot the top restaurant types
    plt.figure(figsize=(8, 5))
    sns.barplot(y=restaurant_types.index, x=restaurant_types.values, palette='coolwarm')

    # Add labels and title
    plt.title('Types of Restaurants')
    plt.xlabel('Number of Restaurants')
    plt.ylabel('Restaurant Type')

    # Show plot
    plt.tight_layout()
    plt.show()
```



OneHotEncoder

```
import pandas as pd
from sklearn.preprocessing import OneHotEncoder

df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

categorical_cols = ['online_order', 'book_table', 'listed_in(type)']

try:
    encoder = OneHotEncoder(sparse=False, drop='first')
except TypeError:
    encoder = OneHotEncoder(sparse_output=False, drop='first')

# Fit and transform categorical data
encoded_array = encoder.fit_transform(df[categorical_cols])

# Create encoded DataFrame
encoded_df = pd.DataFrame(
    encoded_array,
    columns=encoder.get_feature_names_out(categorical_cols),
    index=df.index
)

# Combine with original DataFrame (dropping the original categorical columns)
df_encoded = pd.concat([df.drop(columns=categorical_cols), encoded_df], axis=1)

# Show the first few rows
print(df_encoded.head())
```

```
                  name   rate  votes  cost_for_two  online_order_Yes  \
0                Jalsa  4.1/5    775           800               1.0
1        Spice Elephant  4.1/5    787           800               1.0
2       San Churro Cafe  3.8/5    918           800               1.0
3  Addhuri Udupi Bhojana  3.7/5     88           300               0.0
4          Grand Village  3.8/5    166           600               0.0

   book_table_Yes  listed_in(type)_Cafes  listed_in(type)_Dining  \
0             1.0                    0.0                     0.0
1             0.0                    0.0                     0.0
2             0.0                    0.0                     0.0
3             0.0                    0.0                     0.0
4             0.0                    0.0                     0.0

   listed_in(type)_other
0                    0.0
1                    0.0
2                    0.0
3                    0.0
4                    0.0
```

Splitting Dataset into Training and Testing

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Load dataset
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

# Rename cost column for simplicity
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

# Handle missing values and clean cost column
df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')  # Extract numeric ratings
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')

# Drop rows with missing values
df = df.dropna(subset=['cost_for_two', 'rate'])

# Encode categorical features using one-hot encoding
df_encoded = pd.get_dummies(df[['online_order', 'book_table', 'listed_in(type)']], drop_first=True)

# Combine encoded and numerical features
final_df = pd.concat([df[['cost_for_two']], df_encoded], axis=1)
target = df['rate']  # Assuming you want to predict 'rate'

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(final_df, target, test_size=0.2, random_state=42)

# Check shapes
print("Training set shape:", X_train.shape)
print("Testing set shape:", X_test.shape)
```

```
Training set shape: (118, 6)
Testing set shape: (30, 6)
```

Model Training and Accuracy

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error
import numpy as np

df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)
df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')
df = df.dropna(subset=['cost_for_two', 'rate'])

encoded_df = pd.get_dummies(df[['online_order', 'book_table', 'listed_in(type)']], drop_first=True)

X = pd.concat([df[['cost_for_two']], encoded_df], axis=1)
y = df['rate']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))

print("Model Performance:")
print(f"R² Score: {r2:.3f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.3f}")
```

```
Model Performance:
R² Score: 0.018
Root Mean Squared Error (RMSE): 0.458
```

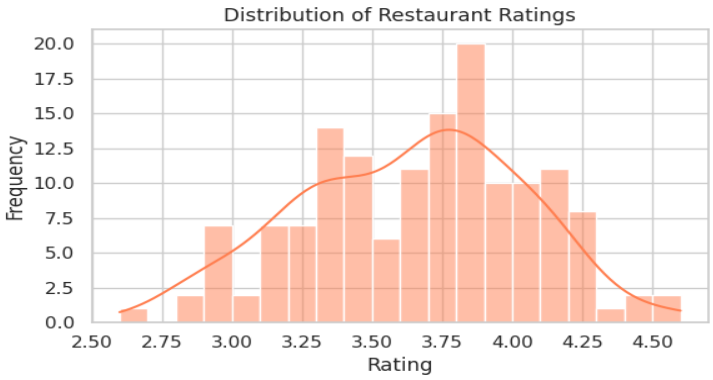
Basic Statistical Analysis

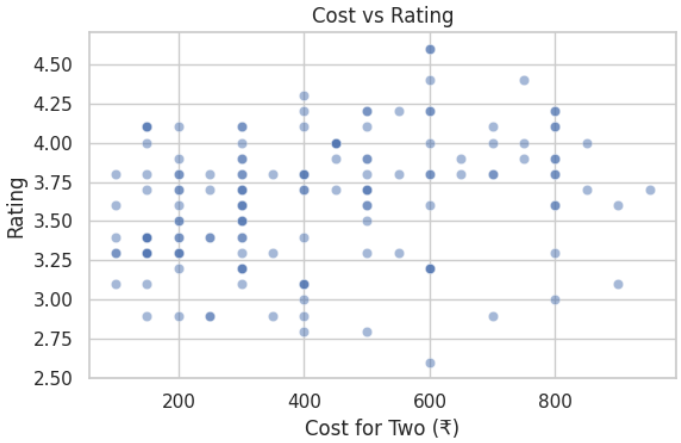
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')

df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)

df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')

df_clean = df.dropna(subset=['cost_for_two', 'rate'])

print("Descriptive Statistics:")
print(df_clean[['cost_for_two', 'rate']].describe())
```

```
Descriptive Statistics:
       cost_for_two        rate
count    148.000000  148.000000
mean     418.243243    3.633108
std      223.085098    0.402271
min      100.000000    2.600000
25%      200.000000    3.300000
50%      400.000000    3.700000
75%      600.000000    3.900000
max      950.000000    4.600000
```

VISUALIZATION

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv("/content/Zomato data .csv", encoding='latin-1')
df.rename(columns={'approx_cost(for two people)': 'cost_for_two'}, inplace=True)
df['cost_for_two'] = df['cost_for_two'].astype(str).str.replace(',', '')
df['cost_for_two'] = pd.to_numeric(df['cost_for_two'], errors='coerce')
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.\d+)')
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')
plt.figure(figsize=(6, 4))
sns.histplot(df_clean['rate'], bins=20, kde=True, color='coral')
plt.title('Distribution of Restaurant Ratings')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()

plt.figure(figsize=(6, 4))
sns.scatterplot(data=df_clean, x='cost_for_two', y='rate', alpha=0.5)
plt.title('Cost vs Rating')
plt.xlabel('Cost for Two (₹)')
plt.ylabel('Rating')
plt.tight_layout()
plt.show()
plt.figure(figsize=(6, 4))
sns.boxplot(x='online_order', y='rate', data=df_clean, palette='Set3')
plt.title('Rating by Online Ordering Availability')
plt.xlabel('Online Order')
plt.ylabel('Rating')
plt.tight_layout()
plt.show()
```


Distribution of Restaurant Ratings
Frequency
Rating
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
2.50
2.75
3.00
3.25
3.50
3.75
4.00
4.25
4.50


Cost vs Rating
Rating
Cost for Two (₹)
4.50
4.25
4.00
3.75
3.50
3.25
3.00
2.75
2.50
200
400
600
800


Rating by Online Ordering Availability
Rating
Online Order
Yes
No
4.50
4.25
4.00
3.75
3.50
3.25
3.00
2.75
2.50