



Restaurant Recommendation System

Project Hand-out, Faculty Development Program – NaanMudhalvan

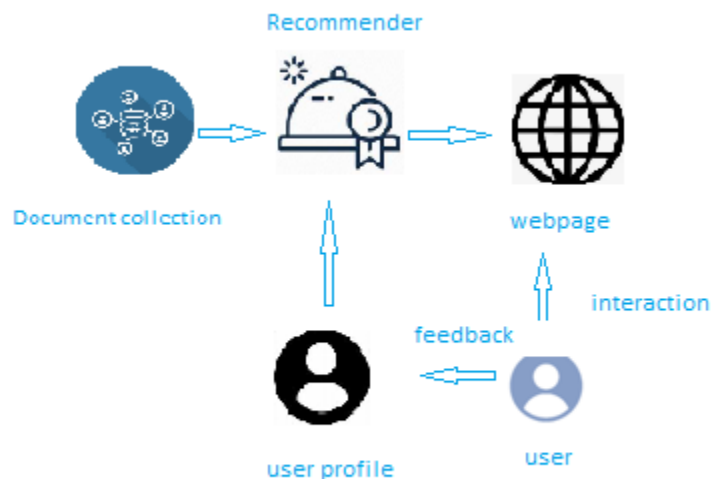
Restaurant Recommendation System

People often look for dining options that match their tastes, location, and mood. With the rise of digital platforms, helping users find the right restaurant has become essential. A Restaurant Recommendation System is designed to suggest personalized dining choices based on user preferences, reviews, cuisine type, and location.

By analyzing user data and restaurant features, the system helps users discover new places, improving their dining experience. It also benefits restaurant owners by increasing visibility and attracting more customers through targeted suggestions.

The main purpose of the Restaurant Recommendation System is to provide accurate and personalized restaurant recommendations that enhance user satisfaction and support restaurant growth.

Technical Architecture:



Project Flow:

- User interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the UI

To accomplish this, we must complete all the activities listed below,

- Data Collection & Preparation
 - Collect the dataset
 - Data Preparation
- Data Preprocessing
 - Importing Libraries
 - Import the dataset
 - Exploratory Data Analysis
 - Data Visualization
- Content Based Filtering
 - Merging datasets
 - Creating the recommendation system
 - Predicting the results
- Application Building
 - Create an HTML file
 - Build a Python Code
- Project Demonstration & Documentation
 - Record explanation Video for project end to end solution
 - Project Documentation-Step by step project development procedure

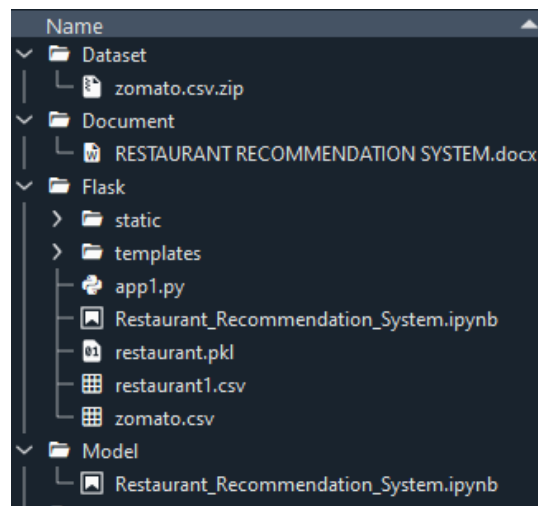
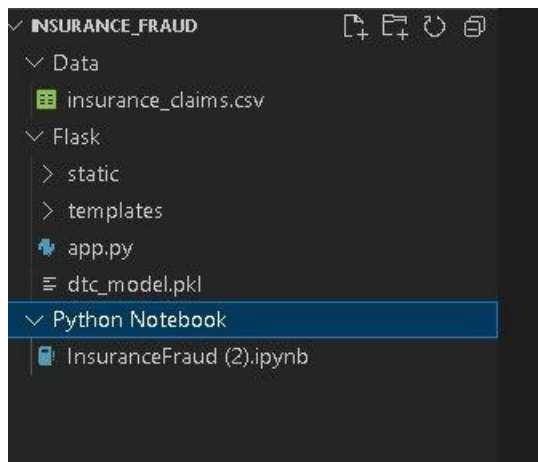
Prior Knowledge:

You must have prior knowledge of the following topics to complete this project.

- ML Concepts
 - Supervised learning: <https://www.javatpoint.com/supervised-machine-learning>
 - Unsupervised learning: <https://www.javatpoint.com/unsupervised-machine-learning>
 - Decision tree: <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>
 - Random forest: <https://www.javatpoint.com/machine-learning-random-forest-algorithm>
 - KNN: <https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning>
 - Xgboost: <https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/>
 - Evaluation metrics: <https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/>
- Flask Basics: https://www.youtube.com/watch?v=Ij4I_CvBnt0

Project Structure:

Create the Project folder which contains files as shown below



- We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
- restaurant.pkl is our saved model. Further we will use this model for flask integration.
- Data Folder contains the Dataset used
- The Notebook file contains procedure for building the model.

Milestone 1: Define Problem / Problem Understanding

Activity 1: Specify the business problem

Users often struggle to find restaurants that match their preferences due to information overload, while restaurants face challenges in reaching potential customers effectively and increasing footfall in a highly competitive market.

Activity 2: Business requirements

Develop a restaurant recommendation system to assist users in finding dining options based on their preferences, location, and other relevant factors. By analyzing user preferences, restaurant ratings, and location data, this project aims to provide personalized recommendations that enhance the dining experience for users.

Scenario 1 (Restaurant Visitors): Implement the recommendation system in mobile apps or websites to help users discover new dining experiences and find restaurants that match their preferences. Enhance user satisfaction, increase customer engagement, and promote loyalty by providing personalized recommendations tailored to individual tastes and preferences.

Scenario 2 (Restaurant Owners): Utilize the recommendation system to attract customers and increase foot traffic to their establishments. Optimize marketing efforts, target promotions, and improve customer retention by leveraging personalized recommendations that align with the restaurant's offerings and ambiance.

Scenario 3 (Food Delivery Platforms): Incorporate the restaurant recommendation system into food delivery apps to help users discover nearby restaurants and make informed choices when ordering food for delivery. Enhance user experience, increase order volume, and improve customer satisfaction by providing relevant and personalized restaurant recommendations based on user preferences and location.

Activity 3: Literature Survey

A Restaurant Recommendation System helps users find suitable dining options based on preferences such as cuisine, location, and price. Literature shows the use of techniques like content-based filtering, collaborative filtering, and hybrid models to enhance recommendation accuracy. Context-aware systems improve relevance by considering factors like time and location. Machine learning and deep learning models are also widely used to predict user preferences. Challenges such as data sparsity, cold-start problems, and privacy concerns still exist. Recent

studies suggest incorporating real-time feedback and contextual signals to further personalize recommendations.

Activity 4: Social or Business Impact.

Social Impact: - Enhanced dining experience: By providing personalized and location-based restaurant suggestions, a restaurant recommendation system helps users discover suitable dining options, improving convenience and satisfaction in their daily lives.

Business Model/Impact: - Increased customer engagement: For restaurants and food delivery platforms, the system drives customer traffic and boosts sales by targeting the right audience with relevant suggestions, ultimately enhancing marketing effectiveness and customer retention.

Milestone 2: Data Collection & preparation

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

Activity 1: Collect the dataset

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.

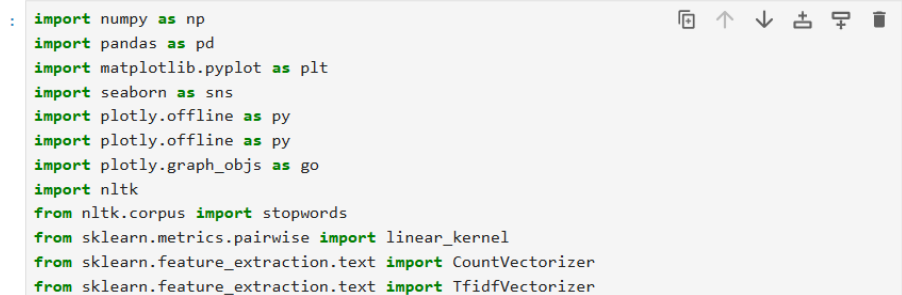
Link: <https://www.kaggle.com/datasets/himanshupoddar/zomato-bangalore-restaurants>

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analyzing techniques.

Note: There are several techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

Activity 1.1: Importing the libraries

Import the necessary libraries as shown in the image.



```
: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.offline as py
import plotly.offline as py
import plotly.graph_objs as go
import nltk
from nltk.corpus import stopwords
from sklearn.metrics.pairwise import linear_kernel
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

Activity 1.2: Read the Dataset

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

In pandas we have a function called read_csv() to read the dataset. As a parameter we have to give the directory of the csv file.


```
[11]: zomato_data=pd.read_csv("zomato.csv")
      zomato_df=zomato_data.copy()
      zomato_df.head(2)
```

```
[11]:
```

| | url | address | name | online_order | book_table | rate | votes | phone | location | rest_type | dish_liked | cuisines |
|---|---|---|----------------|--------------|------------|-------|-------|----------------------------|------------------|---------------|---|--------------------------------|
| 0 | https://www.zomato.com/bangalore/jalsa-banasha... | 942, 21st Main Road, 2nd Stage, Banashankari, ... | Jalsa | Yes | Yes | 4.1/5 | 775 | 42297555/v\n+91 9743772233 | 080 Banashankari | Casual Dining | Pasta, Lunch Buffet, Masala Papad, Paneer Laja... | North Indian, Mughlai, Chinese |
| 1 | https://www.zomato.com/bangalore/spice-elephan... | 2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ... | Spice Elephant | Yes | No | 4.1/5 | 787 | 080 41714161 | Banashankari | Casual Dining | Momos, Lunch Buffet, Chocolate Nirvana, Thai G... | Chinese, North Indian, Thai |

- For checking the null values, `df.isna().any()` function is used. To sum those null values we use `.sum()` function. From the below image we found that there are null values present in our dataset.

Activity 2: Data Preparation

As we have understood how the data is, let's pre-process the collected data.

The download data set is not suitable for training the machine learning model as it might have so much randomness, so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

- Handling missing values
- Handling Outliers

Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

Activity 2.1: Handling missing values

- For checking the null values, `df.isna().any()` function is used. To sum up those null values we use `.sum()` function. From the image below we found that there are null values present in our dataset.

```
] zomato_df.isnull().sum()

:] url                0
   address            0
   name              0
   online_order      0
   book_table        0
   rate             775
   votes            775
   phone           1208
   location          21
   rest_type         227
   dish_liked       28078
   cuisines          45
   approx_cost(for two people) 346
   reviews_list      0
   menu_item          0
   listed_in(type)    0
   listed_in(city)    0
   dtype: int64
```

- We will first drop unnecessary columns. Then we will drop rows with null values and also we will change the rating and cost to float values. Also we will rename our columns for our clarity.

```

): # Drop unnecessary columns
zomato_df = zomato_df.drop(['phone', 'dish_liked', 'url'], axis=1, errors='ignore')

zomato_df.dropna(how='any', inplace=True)

print("Number of duplicates before removal:", zomato_df.duplicated().sum())
zomato_df.drop_duplicates(inplace=True)

zomato_df = zomato_df.rename(columns={
    'approx_cost(for two people)': 'cost',
    'listed_in(type)': 'type',
    'listed_in(city)': 'city'
})

zomato_df = zomato_df[~zomato_df['rate'].isin(['NEW', '-'])].reset_index(drop=True)

zomato_df['rate'] = (
    zomato_df['rate']
    .str.replace('/5', '') # Remove '/5' if present
    .str.strip()           # Remove whitespace
    .astype(float)         # Convert to float
)

zomato_df['cost'] = (
    zomato_df['cost']
    .astype(str)
    .str.replace(',', '.') |
    .astype(float)
)

```

Number of duplicates before removal: 34

- As we have removed the missing values then there will be no missing values. Using `df.isna().sum()` Function we will know that there will be no missing values seen

```

: zomato_df.isnull().sum()

: address      0
  name         0
  online_order  0
  book_table    0
  rate         0
  votes        0
  location     0
  rest_type    0
  cuisines     0
  cost         0
  reviews_list 0
  menu_item    0
  type         0
  city         0
  dtype: int64

```

Activity 2.2: Handling Outliers

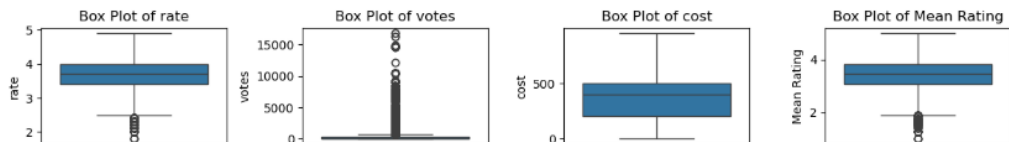
With the help of boxplot, outliers are visualized. And here we are going to find upper bound and lower bound of rate, votes cost and rating feature with some mathematical formula.

- From the diagram below, we could visualize rate, vote cost and rating feature has outliers. Boxplots from seaborn library are used here.

```
[23]: print("\n--- Step 4: Outlier Detection (using Box Plots) ---")

plt.figure(figsize=(15, 8))
for i, col in enumerate(numerical_cols):
    plt.subplot(5, 5, i + 1) # Adjust subplot grid based on number of numerical columns
    sns.boxplot(y=zomato_df[col])
    plt.title(f'Box Plot of {col}')
plt.tight_layout()
plt.show()
```

--- Step 4: Outlier Detection (using Box Plots) ---



Before Outlier analysis, we are calculating mean rating for the restaurants. If we have the same restaurant name we calculate mean rating and then assign that value to all the rows with same restaurant name. Using the head () function, we will get to the top 5 restaurants.

```
# Computing Mean Rating
restaurants = list(zomato_df['name'].unique())
zomato_df['Mean Rating'] = 0.0

for restaurant in restaurants:
    # Calculate mean rating for each restaurant
    mean_rating = zomato_df.loc[zomato_df['name'] == restaurant, 'rate'].mean()
    # Assign the mean rating to all rows for this restaurant
    zomato_df.loc[zomato_df['name'] == restaurant, 'Mean Rating'] = mean_rating

# Scaling the mean rating values
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(1, 5))
zomato_df[['Mean Rating']] = scaler.fit_transform(zomato_df[['Mean Rating']]).round(2)

# Checking the mean rating with restaurant name and rating
print(zomato_df[['name', 'rate', 'Mean Rating']].head())
```

| | name | rate | Mean Rating |
|---|-----------------------|------|-------------|
| 0 | Jalsa | 4.1 | 3.99 |
| 1 | Spice Elephant | 4.1 | 3.97 |
| 2 | San Churro Cafe | 3.8 | 3.58 |
| 3 | Addhuri Udupi Bhojana | 3.7 | 3.45 |
| 4 | Grand Village | 3.8 | 3.58 |

Milestone 3: Exploratory Data Analysis

Activity 1: Descriptive statistical

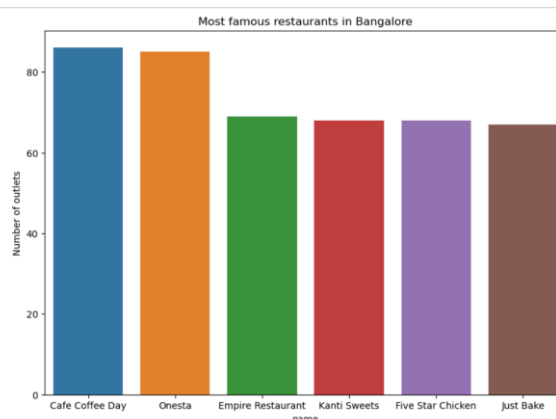
Descriptive analysis is to study the basic features of data with the statistical process. Here pandas have a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.

```
[16]: # Descriptive statistics for numerical columns
print("\nDescriptive statistics for numerical columns:")
print(zomato_df[numerical_cols].describe())
```

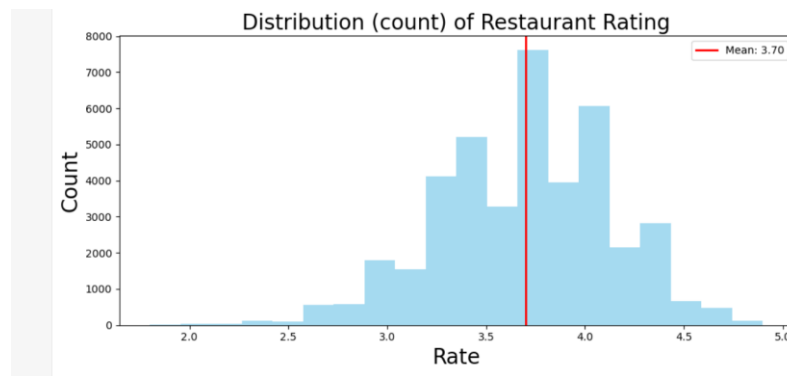
```
Descriptive statistics for numerical columns:
count    41237.000000    41237.000000    41237.000000    41237.000000
mean      3.702030      352.772001      369.586259      3.454193
std       0.440034      884.409230      242.522954      0.537738
min       1.800000       0.000000       1.000000      1.000000
25%       3.400000      21.000000      200.000000      3.060000
50%       3.700000      73.000000      400.000000      3.450000
75%       4.000000      277.000000      500.000000      3.840000
max       4.900000     16832.000000      950.000000      5.000000
```

Activity 2: Visual analysis

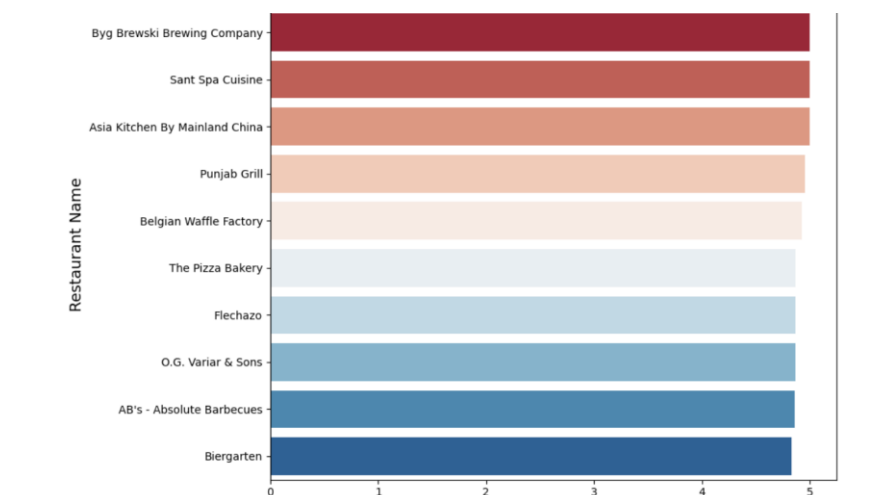
Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.



The graph "Most famous restaurants in Bangalore" shows **Cafe Coffee Day** leading with 40 outlets, followed by **Onesta** (20) and **Empire Restaurant**, highlighting the city's thriving café and quick-service dining culture. **Kanti Sweets** and **Five Star Chicken** have a smaller presence, while **Just Bake** trails with minimal outlets. This reflects Bangalore's diverse food scene, dominated by popular chains and niche eateries.



The graph *"Distribution (count) of Restaurant Rating"* displays the frequency of ratings given to restaurants, with a mean rating of **3.70**. Ratings range from **2.0 to 5.0** in increments of 0.5, suggesting most reviews cluster around the **3.5 to 4.5** range, indicating generally positive feedback. The data reflects a balanced distribution, with fewer extremes at the lowest (2.0) and highest (5.0) ends.



This horizontal bar chart compares Bangalore restaurants by their mean customer ratings (0-5 scale). Top performers like **Byg Brewski** and **Punjab Grill** likely score highest (4+), while **Bergarten** appears at the bottom with a low rating. The visualization highlights the city's diverse dining options and their relative popularity.

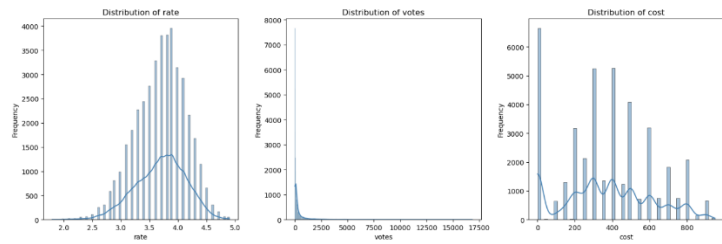
Activity 2.1: Univariate analysis

In simple words, univariate analysis is understanding the data with single feature. Here we

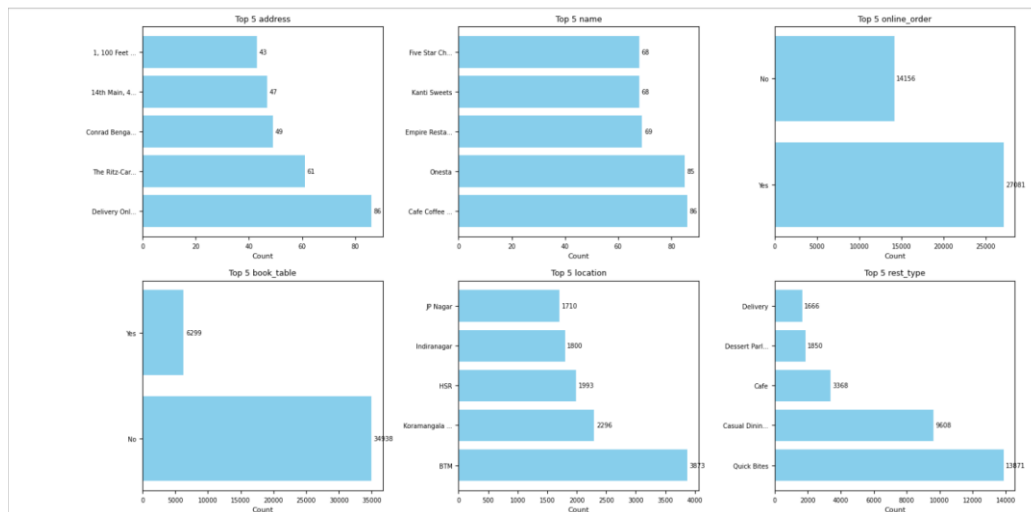
have displayed two different graphs such as bar plot and histogram.

Seaborn package provides a wonderful function histplot. It is more useful for categorical features.

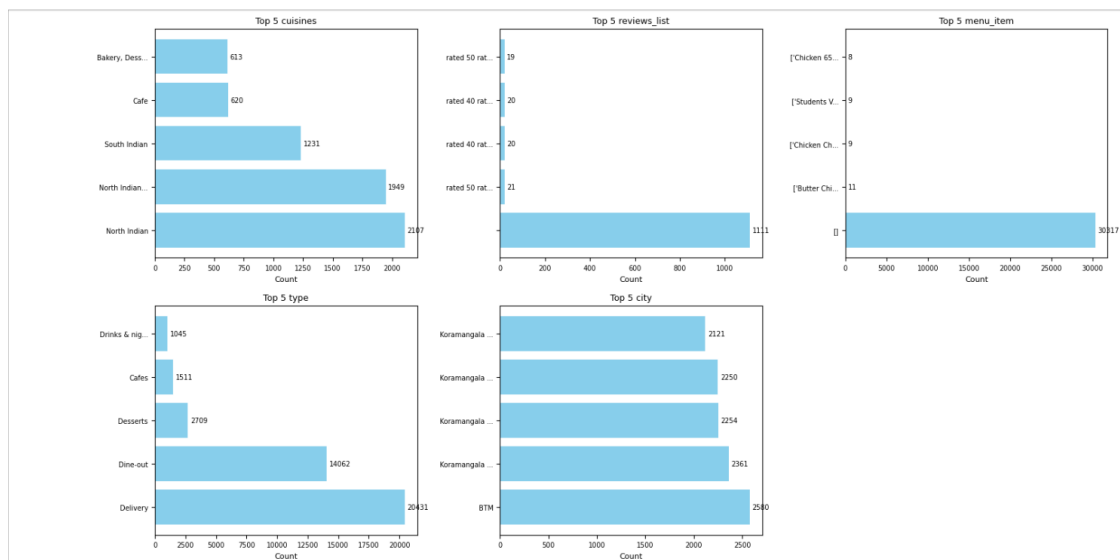
With the help of we can Number of unique values in the feature.



1. Ratings: Peaks at 3.5-4.0 (most common), sharply declines beyond 4.0, with very few extreme ratings (2.0 or 5.0), indicating generally moderate satisfaction.
2. Votes: Highly right-skewed, with most restaurants having under 2,500 votes, while a few outliers reach 15,000+, suggesting most attract limited engagement.
3. Cost: Concentrated below ₹400, with frequencies dropping sharply above this range, revealing Bangalore's predominantly mid-range dining scene.



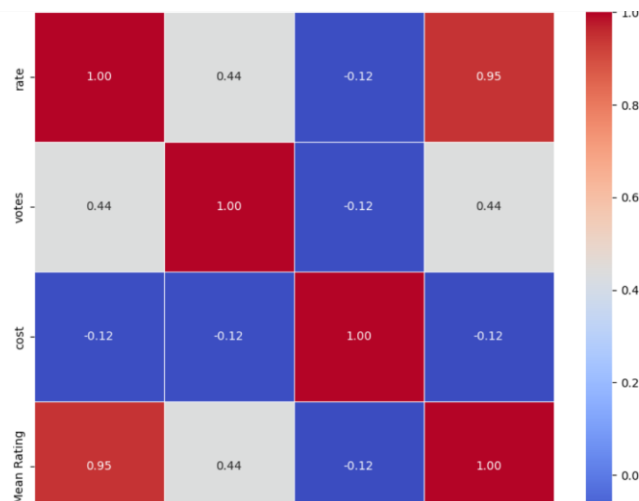
The image shows bar charts of the top 5 categories across various restaurant attributes. Most restaurants support **online ordering** but do **not offer table booking**. **Cafe Coffee Day** is the most frequent restaurant name, and **Delivery Only** is a common address type. **BTM** is the most popular location, and **Quick Bites** is the dominant restaurant type. This indicates a preference for fast, convenient food options in the dataset.



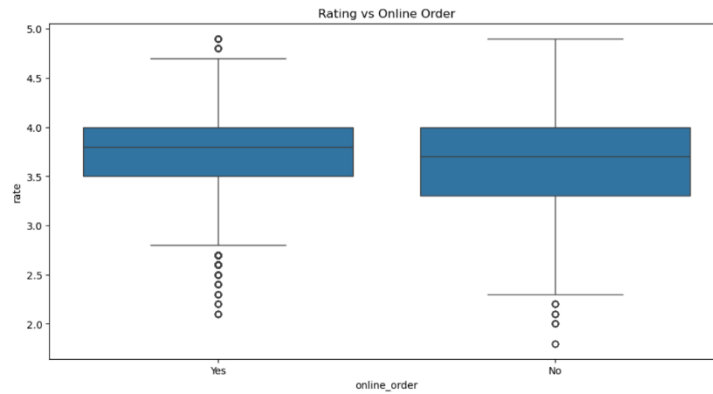
The image displays bar charts showing the top 5 insights from various restaurant features. **North Indian** is the most common cuisine, and **Delivery** is the most popular service type. **Koramangala** and **BTM** dominate the top city locations. Most reviews are rated but not detailed suggesting missing or empty values. Overall, delivery and dine-out options are more prevalent.

Activity 2.2: Bivariate analysis

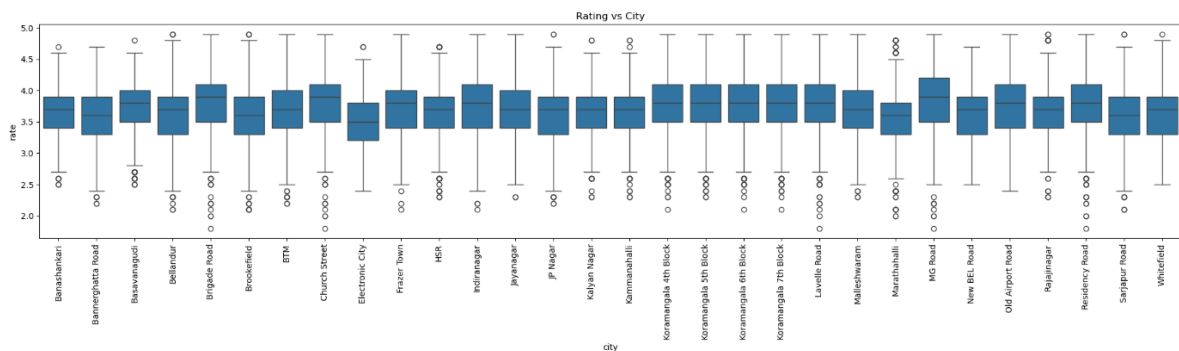
In simple words, multivariate analysis is to find the relation between two features. Here we have a boxplot and correlation matrix from seaborn package.



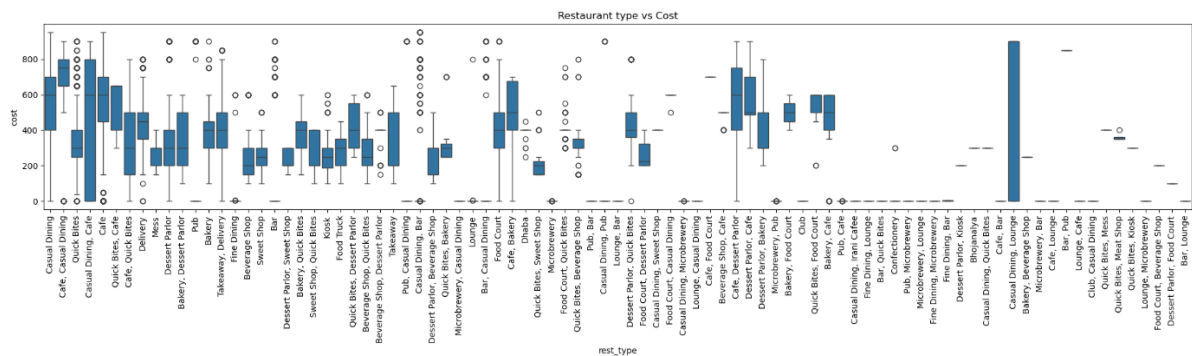
The heatmap shows a strong correlation (0.95) between **rate** and **Mean Rating**, indicating consistency in customer feedback. **Votes** moderately correlate with both **rate** and **Mean Rating** (0.44), suggesting popular places get better ratings. **Cost** has a weak negative correlation with all variables (-0.12), implying price doesn't significantly impact ratings. Overall, ratings and popularity are closely linked, but not strongly influenced by cost.



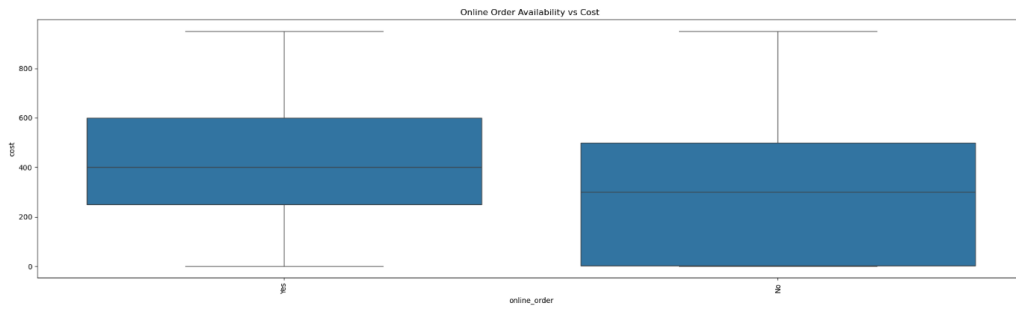
The plot shows restaurants with **online orders (Yes)** tend to earn higher ratings (3.5–4.5), while those without (**No**) have more scattered and lower scores (2.0–3.5). This suggests online ordering correlates with better customer satisfaction. Digital convenience appears to play a key role in dining experiences.



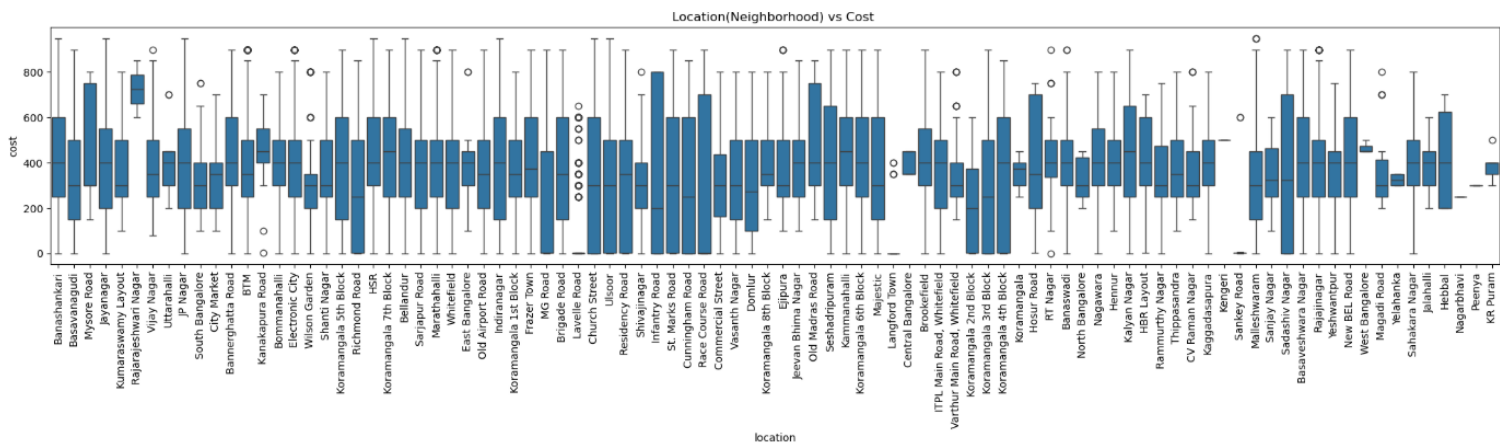
This appears to be a template titled **"Rating vs City"** listing placeholder benchmarks (0-328) without actual city names or rating data. For meaningful analysis, specific city labels and corresponding ratings would need to replace the generic benchmarks. The current content doesn't reveal any actionable insights about restaurant ratings across cities.



The box plot visualizes the distribution of restaurant costs across various restaurant types. Casual Dining and Café types show a wide range of costs, while types like Food Courts and Ice Cream Shops generally have lower and more consistent costs. Outliers are present in many categories, indicating some restaurants charge significantly more than the average.



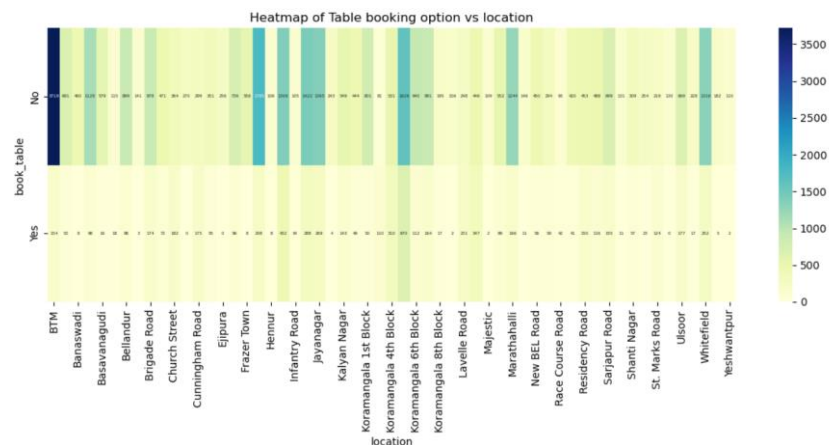
The box plot shows the relationship between online order availability and restaurant cost. Restaurants that accept online orders generally have a slightly higher median cost compared to those that don't. However, both groups display a wide cost range with similar overall variability.



This box plot compares restaurant costs across various neighborhoods. Areas like Indiranagar, Koramangala, and Church Street show higher median costs, suggesting they host more premium dining options. In contrast, places like KR Puram and Yelahanka have lower median costs, indicating generally more affordable dining choices.

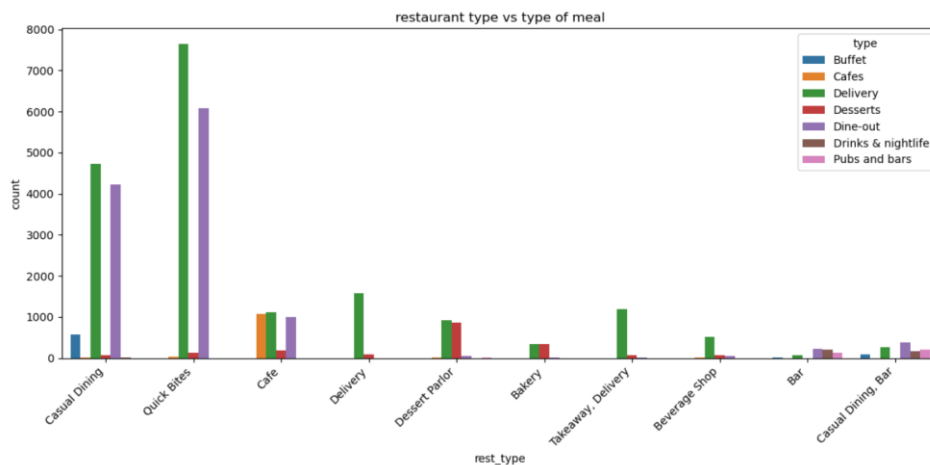
Activity 2.3: Multivariate analysis

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used heatmap from seaborn package.

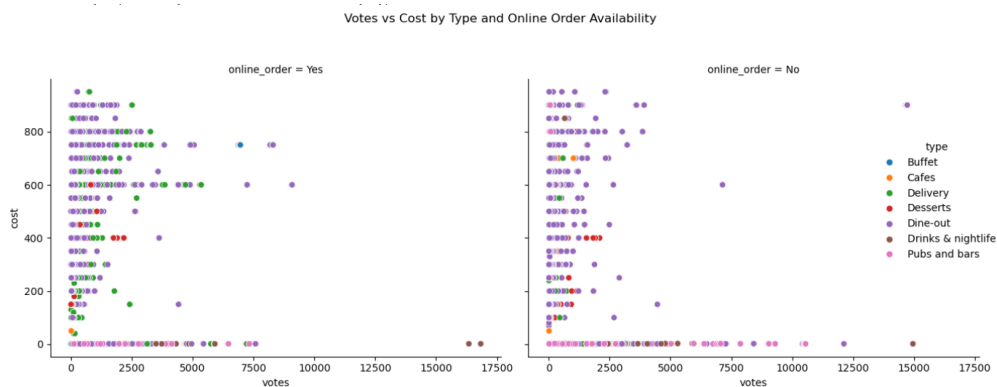


This heatmap displays the distribution of table booking availability across various locations. Most restaurants across all locations do **not offer** table booking, as shown by the dominance of

darker shades in the "No" row. Locations like **BTM**, **Koramangala**, and **Indiranagar** have a relatively higher count of restaurants that allow table booking compared to other areas.



This bar chart illustrates the distribution of meal types across different restaurant types. **Quick Bites** and **Casual Dining** dominate in **Delivery** and **Dine-out** categories. **Cafes** are mainly associated with **Cafes** and **Dine-out** experiences, while **Dessert Parlors** lead in **Desserts**. **Buffets** and **Drinks & nightlife** is relatively limited to **Casual Dining** and **Bars**.



The scatter plot shows that restaurants with **online ordering** generally receive more votes, especially at moderate costs. **Dine-out** types dominate across both groups, with strong representation in high-vote areas. **Delivery** and **Dessert** types are more prevalent among restaurants offering online orders. Restaurants without online orders tend to cluster in the lower vote and cost ranges.

Milestone 4: Model deployment

Here we will be using a declared constructor to route to the HTML page which we have created earlier.

In the above example, '/' URL is bound with the index.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:

Here we are routing our app to recommend () function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.recommnd() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the web.html page earlier.

Main Function:

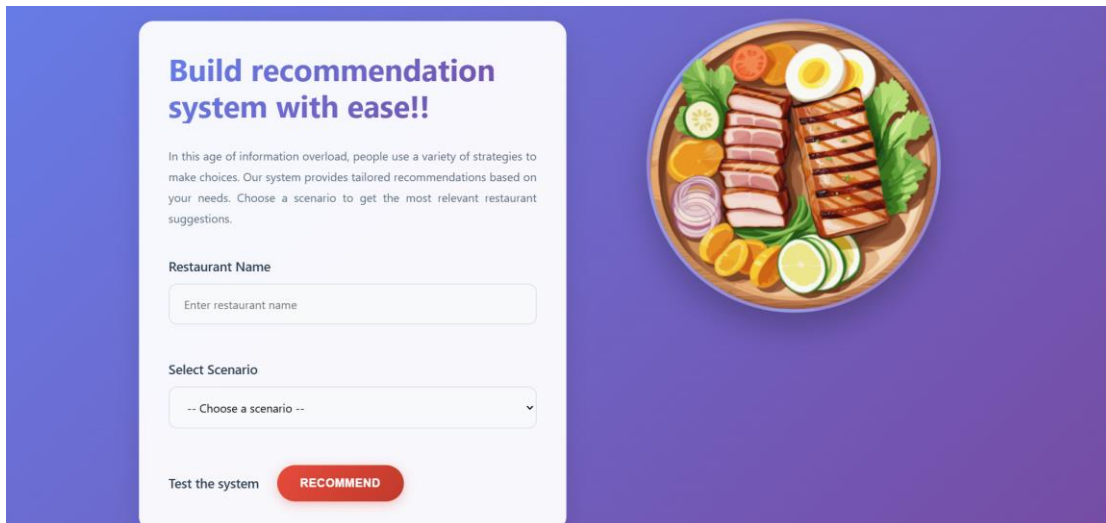
```
if __name__ == '__main__':  
    app.run(debug=True)
```

Activity 2.3: Run the web application

- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type “python app1.py” command
- Navigate to the localhost where you can view your web page.
- Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

```
(base) C:\Users\thanm\Documents\Restaurant Recommendation system\Flask>python app1.py
✅ Model data loaded successfully!
Total restaurants: 41237
* Serving Flask app 'app1'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.1.38:5000
Press CTRL+C to quit
* Restarting with watchdog (windowsapi)
```

Now, Go to the web browser and write the localhost url (http://127.0.0.1:5000) to get the below result



Build recommendation system with ease!!

In this age of information overload, people use a variety of strategies to make choices. Our system provides tailored recommendations based on your needs. Choose a scenario to get the most relevant restaurant suggestions.


Restaurant Name

Enter restaurant name

Select Scenario

-- Choose a scenario --

Test the system **RECOMMEND**



Restaurant Recommendations
Home

Top 10 Dining Recommendations Similar to "Spice Up"

Based on your taste preferences and similar customer reviews

| Restaurant | Cuisines | Rating | Average Cost |
|---------------------|----------|--------|--------------|
| Little Chef | Chinese | 4.0 | 550.0 |
| Mandarin Box | Chinese | 3.9 | 500.0 |
| Gourmet Food Truck | Chinese | 3.8 | 200.0 |
| The Three Dragons | Chinese | 3.3 | 400.0 |
| R R Noodles Express | Chinese | 3.6 | 300.0 |
| Chination | Chinese | 3.5 | 600.0 |

1. Owner Scenario

Restaurant Recommendations
Home

Best Delivery Options Matching "Spice Elephant"


Top-rated restaurants with similar cuisine available for delivery in your area

| Restaurant | Cuisines | Rating | Average Cost |
|---------------------|---------------------------|--------|--------------|
| Soho Bar & Grill | North Indian Chinese Thai | 2.7 | 1.1 |
| Tap House Resto Bar | Chinese North Indian Thai | 3.4 | 900.0 |
| Spice Elephant | Chinese North Indian Thai | 4.1 | 800.0 |
| Tap House Resto Bar | Chinese North Indian Thai | 3.4 | 900.0 |
| Tap House Resto Bar | Chinese North Indian Thai | 3.4 | 900.0 |
| Zaika Take Away | North Indian Chinese Thai | 2.4 | 500.0 |

2. Delivery Scenario

Restaurant Recommendations

Home

 **Best Delivery Options Matching "Spice Elephant"**

Top-rated restaurants with similar cuisine available for delivery in your area

| Restaurant | Cuisines | Rating | Average Cost |
|---------------------|---------------------------|--------|--------------|
| Soho Bar & Grill | North Indian Chinese Thai | 2.7 | 1.1 |
| Tap House Resto Bar | Chinese North Indian Thai | 3.4 | 900.0 |
| Spice Elephant | Chinese North Indian Thai | 4.1 | 800.0 |
| Tap House Resto Bar | Chinese North Indian Thai | 3.4 | 900.0 |
| Tap House Resto Bar | Chinese North Indian Thai | 3.4 | 900.0 |
| Zaika Take Away | North Indian Chinese Thai | 2.4 | 500.0 |

