Artificial Intelligence
CS F407

Lab Project:

Restaurant Recommender System

Report based on CRISP-DM Methodology

Group Members:

1. Aman Agarwal - 2020B4AA2328H
2. Gauri Tewari - 2020B4A32314H
3. Kartik Chitoor - 2020B4A81617H

In accordance with the CRISP-DM (Cross-Industry Process for Data Mining) model, this report is divided into **6 phases** of presentation which highlight the entire working of the project.
It includes everything from the formulation and explanation of the data pipeline to the execution of the mentioned algorithms.

STAGE ONE: Business Understanding

1.1 Determine Business Objectives

1.1.1 Background

In most cities today, there are thousands of restaurants that serve multiple kinds of cuisines, cater to different groups of the population, have their unique selling points and offer different ambiences. Amidst this, the most common way for people to try new places is recommendation by word of mouth, from friends or family. But as each person and their tastes are different, there is a need for a system which takes a user's preferences into account and suggests places which they could try out.

1.1.2 Business Objectives

The most significant objective of this project from the point of view of a business is to provide useful recommendations to a user, who provides his inputs for the attributes he desires. This is also useful to online food delivery services which hope to have customer acquisition by suggesting meaningful recommendations.

1.1.3 Business Success Criteria

The most relevant criteria for success would be a good return rate of customers, and supportive feedback (when collected) about the quality of recommendation. In this project, the criteria is taken as getting the most similar recommendations considering the input preferences of the user.

1.2 Assess Situation

1.2.1 Inventory of Resources

- Data - Several datasets of restaurants in a city/state (along with customer reviews, ratings, cuisines served, location, address, phone number and most liked dishes etc.) from websites on the internet were available

- Computing - Laptops with strong internet connectivity and a decent amount of RAM were needed to run the required tools for this project

- Software - Google Colaboratory with Python language

1.2.2 Requirements, Assumptions, and Constraints

One of the primary requirements of this project was good quality data, and it was assumed that to a certain extent, that data would be enough to build a working recommendation system which can cater to very specific needs as well. There was an upper bound on the amount of data which could be used due to the increase of RAM requirement in proportion to the size of the data.

1.2.3 Risks and Contingencies

The project might fail under the condition that a user provides a certain set of input preferences that do not match the values of any of the feature vectors in the dataset used, which poses the problem of not being able to generate any/accurate recommendations.

1.2.4 Costs and Benefits

The cost of implementation is the main expense along with updating the database as and when a user provides a review for a particular place/dish which has either previously been recommended by the system or not. This, in turn, becomes a benefit for the business in terms of gaining more users who contribute to the database and employ the system to obtain recommendations.

1.3 Determine Data Mining Goals

1.3.1 Data Mining Goals

By using content-based filtering, we aim to take into account the preferences of the user, who will enter their choice of Restaurant type, cuisine, cost, and location of the restaurant.

1.3.2 Data Mining Success Criteria

The most apt success criteria would be to come up with the most accurate predictions for a user's input, where accuracy indicates the degree of similarity between the information provided by the user and the output which the system generates.

1.4 Produce Project Plan

1.4.1 Project Plan

- Finding and incorporating the necessary data (dataset)
- Find the most appropriate techniques to perform data cleaning and preprocessing as required for the implementation of algorithms
- Assess which features are the most important for our goal and decide how to handle the ones which are not very useful/junk
- Decide on the techniques and algorithms to be used to build the system
- Provide the fields for user input
- Generate recommendations as expected

1.4.2 Initial Assessment of Tools and Techniques

For a preliminary analysis, we could guess that using a technique which lets us quantify the "similarity" between two entries in the database will prove quite useful. Then, if we could somehow arrange the predictions in decreasing order of how much their attribute values match the user's input preferences, we could consider the job well done.

STAGE TWO: Data Understanding

2.1 Initial Data Collection Report

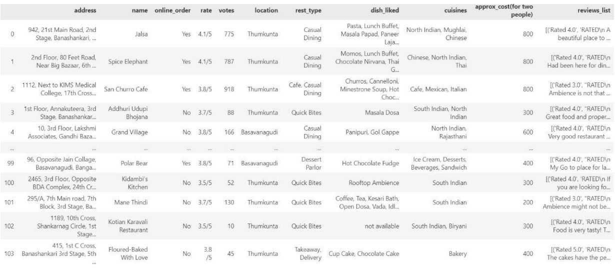
- For our project, we have taken the dataset of ZOMATO "Reviews of restaurants in Hyderabad". This dataset was available on a website called Kaggle.

2.2 Data Description Collection Report

- In the dataset that we acquired, the data was categorized into different columns (attributes) such as:
    1. URL
    2. Address
    3. Name
    4. Online_order
    5. Book_table
    6. Rate
    7. Votes
    8. Phone
    9. Location
    10. Rest_type
    11. Dish_liked
    12. Cuisines
    13. Approx_cost(for two people)
    14. Reviews_list
    15. Menu_item
    16. Listed_in(type)
    17. Listed_in(city)

- There are a total of 51717 entries (rows) and 17 attributes (columns) in the data that we have collected.
- This data is more than sufficient for our analysis as it contains many factors that help us determine the different metrics of a restaurant and also helps ease the task of recommending similar restaurants based on the user's preference.

- The following is a snapshot of how the data looks like.



	address	name	online_order	rate	votes	location	rest_type	dish_liked	cuisines	approx_cost(for two people)	reviews_list
0	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	4.1/5	775	Thumkunta	Casual Dining	Pasta, Lunch Buffet, Masala Papad, Paneer Laja...	North Indian, Mughlai, Chinese	800	[('Rated 4.0', 'RATED\n A beautiful place to ...
1	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	4.1/5	787	Thumkunta	Casual Dining	Momos, Lunch Buffet, Chocolate Nirvana, Thai G...	Chinese, North Indian, Thai	800	[('Rated 4.0', 'RATED\n Had been here for din...
2	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	3.8/5	918	Thumkunta	Cafe, Casual Dining	Churros, Cannelloni, Minestrone Soup, Hot Choc...	Cafe, Mexican, Italian	800	[('Rated 3.0', "RATED\n Ambience is not that ...
3	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	3.7/5	88	Thumkunta	Quick Bites	Masala Dosa	South Indian, North Indian	300	[('Rated 4.0', "RATED\n Great food and proper...
4	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	3.8/5	166	Basavanagudi	Casual Dining	Panipuri, Gol Gappe	North Indian, Rajasthani	600	[('Rated 4.0', 'RATED\n Very good restaurant ...
...	...	...	...	...	...	...	...	...	...	...	...
99	96, Opposite Jain Collage, Basavanagudi, Banga...	Polar Bear	Yes	3.8/5	71	Basavanagudi	Dessert Parlor	Hot Chocolate Fudge	Ice Cream, Desserts, Beverages, Sandwich	400	[('Rated 4.0', "RATED\n My Go to place for la...
100	2465, 3rd Floor, Opposite BDA Complex, 24th Cr...	Kidambi's Kitchen	No	3.5/5	52	Thumkunta	Quick Bites	Rooftop Ambience	South Indian	300	[('Rated 4.0', 'RATED\n If you are looking fo...
101	295/A, 7th Main road, 7th Block, 3rd Stage, Ba...	Mane Thindi	No	3.7/5	130	Thumkunta	Quick Bites	Coffee, Tea, Kesari Bath, Open Dosa, Vada, Idl...	South Indian	200	[('Rated 3.0', "RATED\n Ambience might not be...
102	1189, 10th Cross, Shankarnag Circle, 1st Stage...	Kotian Karavali Restaurant	No	3.5/5	10	Thumkunta	Quick Bites	not available	South Indian, Biryani	300	[('Rated 4.0', 'RATED\n Food is very tasty! T...
103	415, 1st C Cross, Banashankari 3rd Stage, 5th ...	Floured-Baked With Love	No	3.8 /5	45	Thumkunta	Takeaway, Delivery	Cup Cake, Chocolate Cake	Bakery	400	[('Rated 5.0', 'RATED\n The cakes have the pe...

2.3 Data Exploration Report

- In our findings, we noticed that there were two types of data available i.e., continuous as well as discretized. The columns such as rating, approx cost for two people, which should be integers, were strings. Almost all the restaurants had many different cuisines and also the restaurant types were also more than one for most of the restaurants. Our code includes visualizations to analyze the number of restaurants in each location, the count of each cuisine, and the top-rated restaurants.
- We have also visually represented the type of restaurants in each location, the price and also how many restaurants take online orders in each location and the price of the same.

2.4 Verify Data Quality

- The raw data that we had collected had many issues, but after we have applied the steps such as removing the rows where we thought that the missing data wouldn't matter much as well as encoding a few columns to get less number of variables while calculating the accuracy have ensured that the data is much cleaner now.The steps that we have taken to clean the data should be enough for users to get much more accurate recommendations.

STAGE THREE: Data Preparation

3.1 Rationale for inclusion/ exclusion

- In this section, we describe how we have selected the necessary columns for further processing.
- As discussed previously, we have 17 attributes (columns) in our data. In order to reduce the number of variables, we decided to remove the following columns:

1. URL
2. book_table
3. Phone
4. Menu_item
5. Listed_in(type)
6. Listed_in(city)

We removed these variables from data because:

For values like Phone number, URL and Menu_item there were a lot of missing items. Also we couldn't substitute the empty items for some other item as these attributes are unique for each restaurant.

- The reason for removal of values like Listed_in(type),Listed_in(city) and book_table is that these don't hold much significance for the user while recommending restaurants. Listed_in(type),Listed_in(city) are very similar to rest_type and location respectively. This is also one of the reasons to exclude them.

- With the exclusion of these attributes, our data looks cleaner as well as concise but we have yet to clean the data further which is explained in the next stages.

3.2 Data Cleaning Report

- The data still has a few missing values in the following columns.

  rate - 7775 - 15.03%

  location - 21 - 0.04%

  rest_type - 227 - 0.44%

  dish_liked - 28078 - 54.29%

  cuisines - 45 - 0.09%

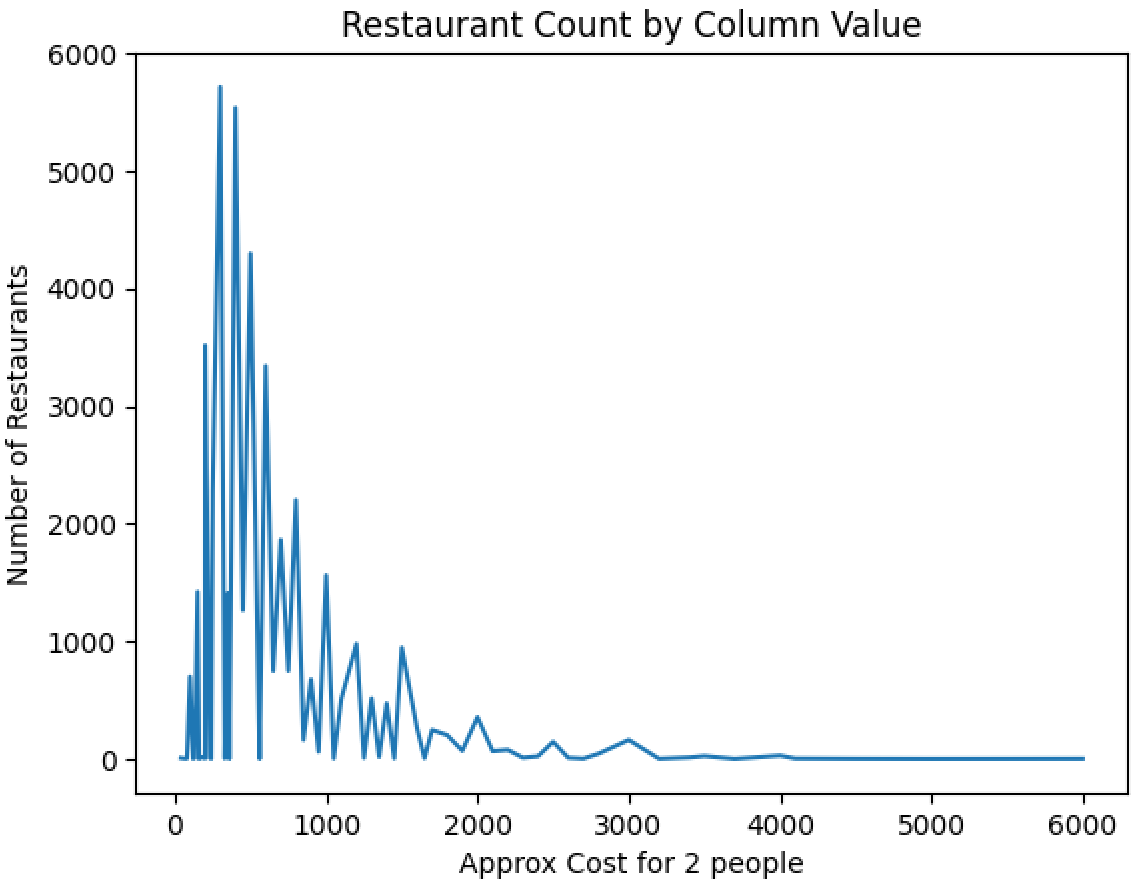
  approx_cost(for two people) - 346 - 0.67%

- Since rate, location, rest_type, cuisines, approx_cost(for two people) have negligible missing values, we dropped those rows.

- The column of dish_liked has many missing values, so we can't disregard them. So we replaced them with "Not Available". The size of the data is 43533 and 11 columns. Now by removing the commas and formatting the data to float type, we process the rate column as well.

- We observed that a few columns have too many variables, so we encoded them such that the columns' data is categorized in a few distinct categories. This is done so that when the data is modeled we get better accuracy.

- For example, the rate column has continuous values which we have categorized into 5 types on the basis of graph we plotted for different ranges as given below:



₹0 - ₹500	—>	Cheap
₹501 - ₹800	—>	Reasonable
₹801 - ₹1200	—>	Affordable
₹1201 - ₹1800	—>	Moderately Expensive
₹1801 - ₹2500	—>	Expensive
₹2501 and above	—>	Lavish

- We have also encoded the rest_type and location by giving the value of "others" to the categories that are less than 100 in number.

3.3 Data Construction Report

3.3.1 Derived attributes

- For our data there weren't any Derived Attributes i.e., all the attributes were independent of each other. But we can say the new attribute cost which is now discretised is derived from the ranges available in the original data.

3.3.2 Generated records

- To implement our model, we created a new column which is a combination of cuisines, rest_type, location and cost. The similarity scores of this column with respect to the user's input helps us to recommend similar restaurants.

3.4 Merged Data

- We generated a new column that merges cuisines, rest_type, location, and rate in order to put our model into practice. We can propose comparable eateries based on the similarity scores of this column relative to the user's input.

3.5 Reformatted Data

- We also process the rate column by eliminating the commas and formatting the data to float type.
- We saw that certain columns contain too many variables, therefore, we encoded the columns' data to group its information into a limited number of separate categories.

STAGE FOUR: MODELING

4.1 Select Modeling Techniques

The goal is to determine the most suitable modelling techniques for developing the restaurant recommender system. We have leveraged the **TF-IDF (Term Frequency-Inverse Document Frequency)** technique in combination with **cosine similarities.**

TF-IDF is a numerical measure that reflects the importance of a term in a document using given metrics, while cosine similarity measures the similarity between two vectors. Using TF-IDF and cosine similarities, the recommender system can identify relevant restaurants based on user preferences and recommend them accordingly.

This is a use case of content-based filtering, which focuses on the various attributes of the restaurants. This helps us select the restaurants which match the criteria given by the user.

4.2 Generate Test Design

Creating a comprehensive test design is essential to evaluate the performance and accuracy of the recommender that is being built. The test design ideally should include multiple scenarios and datasets to assess the recommender's ability to suggest relevant restaurants to users with different preferences. It is also essential to have a balanced mix of positive and negative test cases, representing situations where the recommender system successfully suggests suitable restaurants and cases where it fails to do so. This ensures that we cover the most possible situations which the system may encounter. We should also consider scalability and performance to ensure the system can handle many users, restaurant options and diverse records.

4.3 Build Model

- **About TF-IDF:**
  TF-IDF (Term Frequency-Inverse Document Frequency) is used for information retrieval, essentially to quantify the importance of a term within a document or a collection of documents.

  TF - term frequency represents the frequency of a specific term within a given document. It computes how often a particular term appears in a document.

  IDF - inverse document frequency is used to measure the importance of a term across the entire collection of documents (also known as corpus). It helps us to identify terms that are relatively rare but have a high influence in differentiating one restaurant from another. This quantity is calculated by taking the logarithm of the ratio between the total number of documents and the number of documents containing the specific term for which we found TF.

  Finally,

  TF-IDF score = TF * IDF

  So by calculating TF-IDF scores for all terms (from the reviews, cuisine and the other important textual columns) in each restaurant description, we can represent each restaurant as a numerical vector.

- **About cosine similarities:**
  Given two vectors (say A and B), cosine similarity measures the cosine of the angle between them. The more the value of the cosine similarly, the more similar the direction the vectors lie in. All values of cosine similarities lie between -1 and 1 (including both). The cosine similarity between vectors A and B is calculated as follows:

$$\textbf{Cosine\_Similarity}(A, B) = (A \bullet B) / (\|A\| * \|B\|)$$

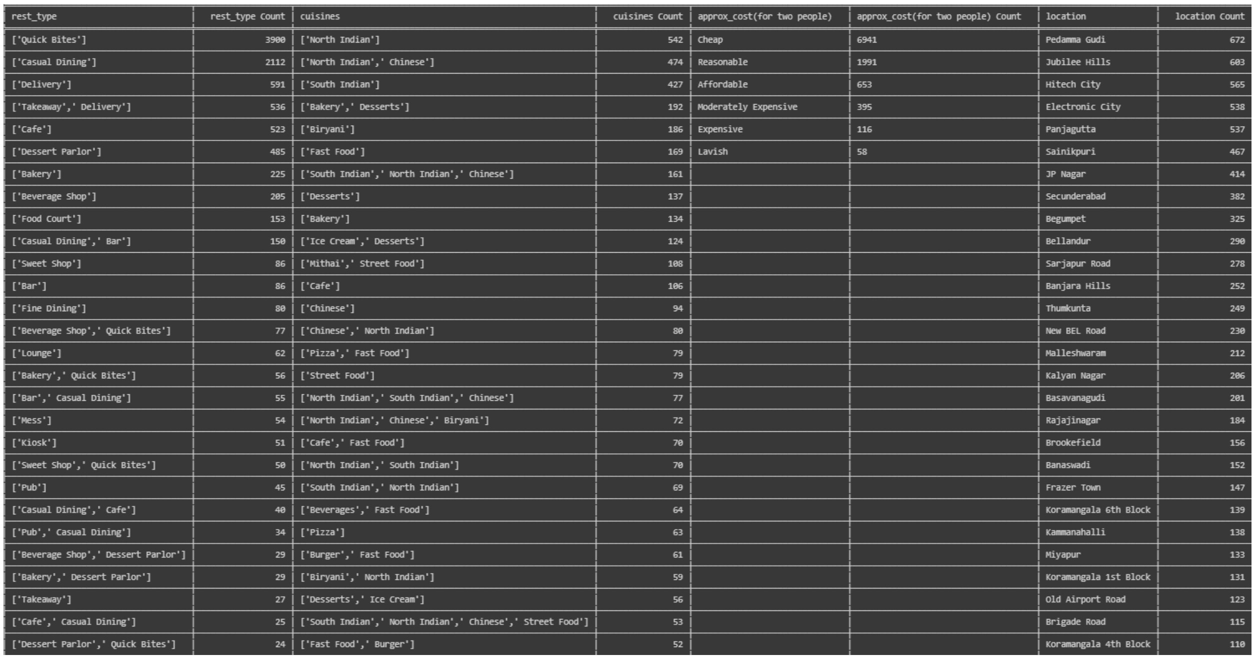
where

$(A \bullet B)$ represents the dot product of vectors A and B,
$\|A\|$ and $\|B\|$ denote the magnitudes of the vectors

So, the vectors obtained through TF-IDF are then used to calculate cosine similarities and find restaurants with similar textual characteristics, thus executing the recommendation process.

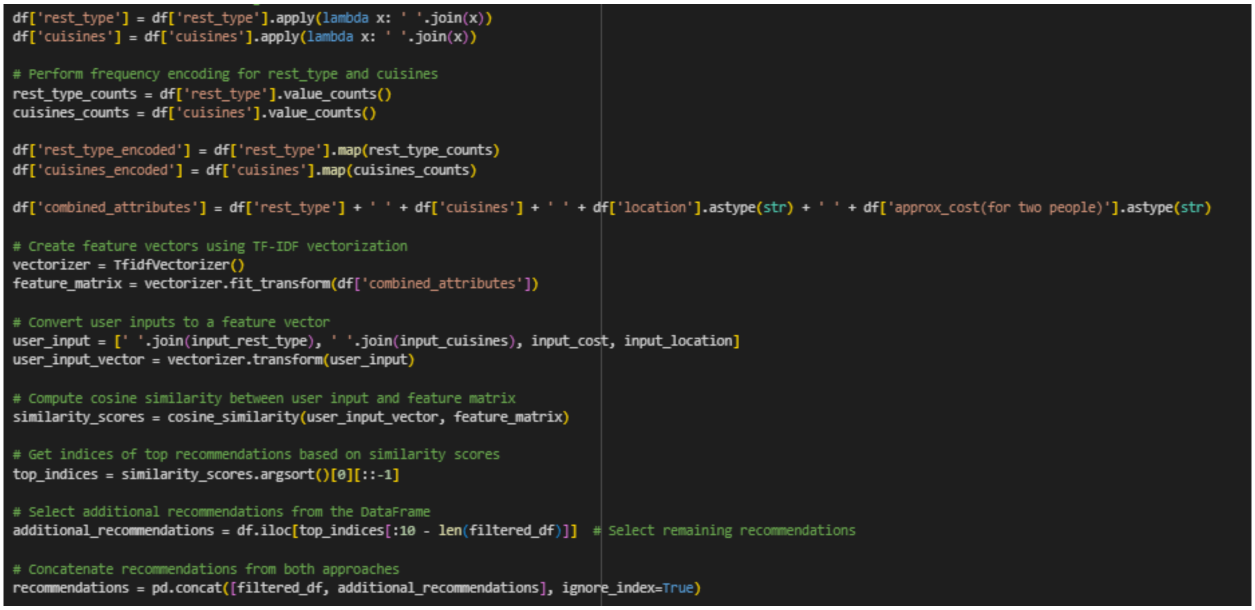
4.4 Assess Model

So first user is asked for input by displaying the following input options in a clean tabular format which shows the options along with the counts of how many restaurants are available for that particular option:



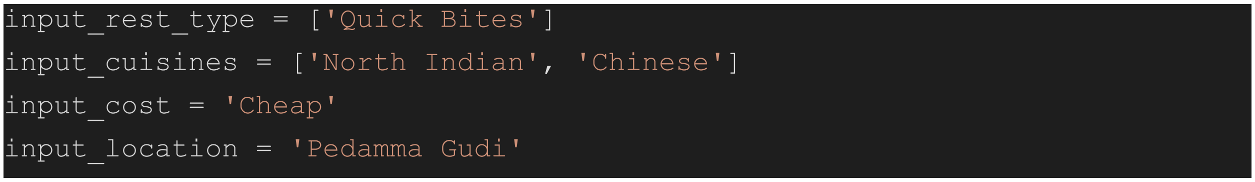
rest_type	rest_type Count	cuisines	cuisines Count	approx_cost(for two people)	approx_cost(for two people) Count	location	location Count
['Quick Bites']	3900	['North Indian']	542	Cheap	6941	Pedamma Gudi	672
['Casual Dining']	2112	['North Indian',' Chinese']	474	Reasonable	1991	Jubilee Hills	603
['Delivery']	591	['South Indian']	427	Affordable	653	Hitech City	565
['Takeaway',' Delivery']	536	['Bakery',' Desserts']	192	Moderately Expensive	395	Electronic City	538
['Cafe']	523	['Biryani']	186	Expensive	116	Panjagutta	537
['Dessert Parlor']	485	['Fast Food']	169	Lavish	58	Sainikpuri	467
['Bakery']	225	['South Indian',' North Indian',' Chinese']	161			JP Nagar	414
['Beverage Shop']	205	['Desserts']	137			Secunderabad	382
['Food Court']	153	['Bakery']	134			Begumpet	325
['Casual Dining',' Bar']	150	['Ice Cream',' Desserts']	124			Bellandur	290
['Sweet Shop']	86	['Mithai',' Street Food']	108			Sarjapur Road	278
['Bar']	86	['Cafe']	106			Banjara Hills	252
['Fine Dining']	80	['Chinese']	94			Thumkunta	249
['Beverage Shop',' Quick Bites']	77	['Chinese',' North Indian']	80			New BEL Road	230
['Lounge']	62	['Pizza',' Fast Food']	79			Malleshwaram	212
['Bakery',' Quick Bites']	56	['Street Food']	79			Kalyan Nagar	206
['Bar',' Casual Dining']	55	['North Indian',' South Indian',' Chinese']	77			Basavanagudi	201
['Mess']	54	['North Indian',' Chinese',' Biryani']	72			Rajajinagar	184
['Kiosk']	51	['Cafe',' Fast Food']	70			Brookefield	156
['Sweet Shop',' Quick Bites']	50	['North Indian',' South Indian']	70			Banaswadi	152
['Pub']	45	['South Indian',' North Indian']	69			Frazer Town	147
['Casual Dining',' Cafe']	40	['Beverages',' Fast Food']	64			Koramangala 6th Block	139
['Pub',' Casual Dining']	34	['Pizza']	63			Kammanahalli	138
['Beverage Shop',' Dessert Parlor']	29	['Burger',' Fast Food']	61			Miyapur	133
['Bakery',' Dessert Parlor']	29	['Biryani',' North Indian']	59			Koramangala 1st Block	131
['Takeaway']	27	['Desserts',' Ice Cream']	56			Old Airport Road	123
['Cafe',' Casual Dining']	25	['South Indian',' North Indian',' Chinese',' Street Food']	53			Brigade Road	115
['Dessert Parlor',' Quick Bites']	24	['Fast Food',' Burger']	52			Koramangala 4th Block	110

After getting the input, the model applies TF-IDF to store the restaurants in a list by using the following piece of code:



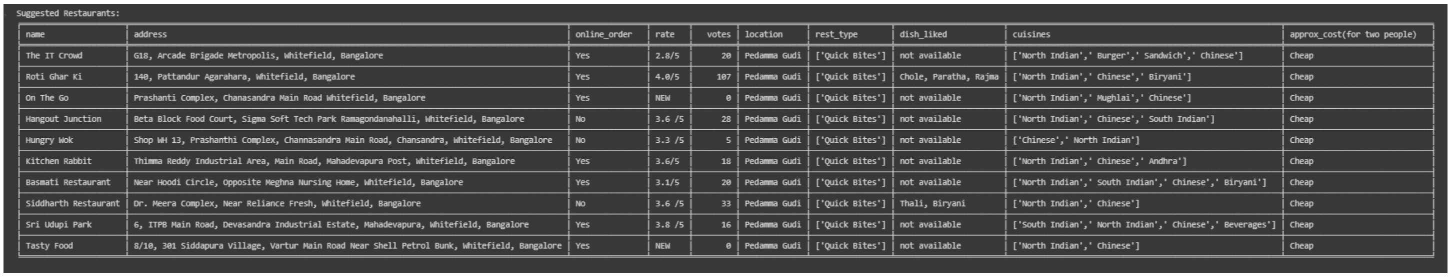
```python
df['rest_type'] = df['rest_type'].apply(lambda x: ' '.join(x))
df['cuisines'] = df['cuisines'].apply(lambda x: ' '.join(x))

# Perform frequency encoding for rest_type and cuisines
rest_type_counts = df['rest_type'].value_counts()
cuisines_counts = df['cuisines'].value_counts()

df['rest_type_encoded'] = df['rest_type'].map(rest_type_counts)
df['cuisines_encoded'] = df['cuisines'].map(cuisines_counts)

df['combined_attributes'] = df['rest_type'] + ' ' + df['cuisines'] + ' ' + df['location'].astype(str) + ' ' + df['approx_cost(for two people)'].astype(str)

# Create feature vectors using TF-IDF vectorization
vectorizer = TfidfVectorizer()
feature_matrix = vectorizer.fit_transform(df['combined_attributes'])

# Convert user inputs to a feature vector
user_input = [' '.join(input_rest_type), ' '.join(input_cuisines), input_cost, input_location]
user_input_vector = vectorizer.transform(user_input)

# Compute cosine similarity between user input and feature matrix
similarity_scores = cosine_similarity(user_input_vector, feature_matrix)

# Get indices of top recommendations based on similarity scores
top_indices = similarity_scores.argsort()[0][::-1]

# Select additional recommendations from the DataFrame
additional_recommendations = df.iloc[top_indices[:10 - len(filtered_df)]]  # Select remaining recommendations

# Concatenate recommendations from both approaches
recommendations = pd.concat([filtered_df, additional_recommendations], ignore_index=True)
```

After storing the top 10 recommendations in the list, it is printed in a neat way, along with all the important attributes. A sample output for the input:



```python
input_rest_type = ['Quick Bites']
input_cuisines = ['North Indian', 'Chinese']
input_cost = 'Cheap'
input_location = 'Pedamma Gudi'
```

Is shown below.



```
Suggested Restaurants:
| name                | address                                                                            | online_order | rate   | votes | location     | rest_type       | dish_liked           | cuisines                                                | approx_cost(for two people) |
|---------------------|------------------------------------------------------------------------------------|--------------|--------|-------|--------------|-----------------|----------------------|---------------------------------------------------------|-----------------------------|
| The IT Crowd        | G18, Arcade Brigade Metropolis, Whitefield, Bangalore                              | Yes          | 2.8/5  |    20 | Pedamma Gudi | ['Quick Bites'] | not available        | ['North Indian',' Burger',' Sandwich',' Chinese']       | Cheap                       |
| Roti Ghar Ki        | 140, Pattandur Agarahara, Whitefield, Bangalore                                    | Yes          | 4.0/5  |   107 | Pedamma Gudi | ['Quick Bites'] | Chole, Paratha, Rajma | ['North Indian',' Chinese',' Biryani']                 | Cheap                       |
| On The Go           | Prashanti Complex, Chanasandra Main Road Whitefield, Bangalore                     | Yes          | NEW    |     0 | Pedamma Gudi | ['Quick Bites'] | not available        | ['North Indian',' Mughlai',' Chinese']                  | Cheap                       |
| Hangout Junction    | Beta Block Food Court, Sigma Soft Tech Park Ramagondanahalli, Whitefield, Bangalore | No          | 3.6 /5 |    28 | Pedamma Gudi | ['Quick Bites'] | not available        | ['North Indian',' Chinese',' South Indian']             | Cheap                       |
| Hungry Wok          | Shop WH 13, Prashanthi Complex, Channasandra Main Road, Chansandra, Whitefield, Bangalore | No     | 3.3 /5 |     5 | Pedamma Gudi | ['Quick Bites'] | not available        | ['Chinese',' North Indian']                             | Cheap                       |
| Kitchen Rabbit      | Thimma Reddy Industrial Area, Main Road, Mahadevapura Post, Whitefield, Bangalore  | Yes          | 3.6/5  |    18 | Pedamma Gudi | ['Quick Bites'] | not available        | ['North Indian',' Chinese',' Andhra']                   | Cheap                       |
| Basmati Restaurant  | Near Hoodi Circle, Opposite Meghna Nursing Home, Whitefield, Bangalore             | Yes          | 3.1/5  |    20 | Pedamma Gudi | ['Quick Bites'] | not available        | ['North Indian',' South Indian',' Chinese',' Biryani']  | Cheap                       |
| Siddharth Restaurant | Dr. Meera Complex, Near Reliance Fresh, Whitefield, Bangalore                     | No           | 3.6 /5 |    33 | Pedamma Gudi | ['Quick Bites'] | Thali, Biryani       | ['North Indian',' Chinese']                             | Cheap                       |
| Sri Udupi Park      | 6, ITPB Main Road, Devasandra Industrial Estate, Mahadevapura, Whitefield, Bangalore | Yes        | 3.8 /5 |    16 | Pedamma Gudi | ['Quick Bites'] | not available        | ['South Indian',' North Indian',' Chinese',' Beverages'] | Cheap                      |
| Tasty Food          | 8/10, 301 Siddapura Village, Vartur Main Road Near Shell Petrol Bunk, Whitefield, Bangalore | Yes  | NEW    |     0 | Pedamma Gudi | ['Quick Bites'] | not available        | ['North Indian',' Chinese']                             | Cheap                       |
```

So as we can see, all the input attributes are satisfied in the output; hence the approach used is working, and the accuracy is great. In case all the attributes are not satisfied, i.e. there do not exist ten restaurants with the attribute values which the user wants, then this technique will find the restaurants which are closest to those values; it will compromise one value but keep all others the same and so on. This is happening because of the decreasing order of similarity scores.

In addition to the above, user feedback and satisfaction surveys can be conducted to gather useful data and insights regarding the effectiveness of the recommender system. Feedback can also help us to identify areas which need improvement.

Hence, we use several resources to fine-tune the system, so it can provide personalised and relevant recommendations to users, enhancing their dining experiences and increasing consumer satisfaction.

STAGE FIVE: EVALUATION

5.1 Evaluate Results

In this phase, we attempt to analyse the recommendations generated by the system, measuring key metrics and evaluating the system's responses and to what extent they satisfy the requirements presented by the user. The evaluation process might also provide insights into the strengths and weaknesses of the system.

- Review the recommendations: To do this; we examine the recommendations generated by the system for a set of users and/or test cases. An important point would be to see how well the system can handle diversified options and is still able to generate valuable results.

- Calculate evaluation metrics: Utilize appropriate evaluation metrics to measure the performance of the recommender system. Common metrics include precision, recall, F1-score, and accuracy. Precision measures the proportion of relevant recommendations among the total recommendations made, while recall calculates the proportion of relevant recommendations identified from the entire set of relevant items. F1-score provides a balanced measure between precision and recall. Accuracy measures the overall correctness of the recommendations.

- Conduct user surveys and feedback analysis: We can also gather user feedback through surveys and feedback forms to understand their satisfaction with the recommendations. This will also help us consider varied perspectives and requirements which can then be integrated into the system.

- Consider business impact: We may evaluate the impact of the recommender system with respect to key business metrics such as consumer engagement, conversion rates, and revenue. This involves analyzing whether the system has positively influenced user behavior as expected, provided customer satisfaction, or improved business outcomes for those which are a part of the database fed to the system.

Based on the evaluation results, further analysis can be performed to identify factors contributing to the system's performance and potential areas for improvement.

5.2 Review Process

To summarise, we obtained a dataset which was appropriately cleaned and preprocessed, and exploratory data analysis was performed on it. Then we drafted our business objectives and prepared a path to build the system. After deciding the techniques, tools, and frameworks to be used, we implemented them in the most effective way possible. Further, we also found metrics and criteria to evaluate and improve the performance of the model.

5.3 Determine The Next Steps

The following steps can be implemented or enhanced in the future:

- Addressing limitations in terms of available data and resources (if found)
- Giving weights to the different attributes available and prioritising one attribute over the other. For example, if a user wants that location has to be strictly Secunderabad but other things can vary then we can assign a higher weight to the ‘location’ attribute so that TF-IDF calculates similarity scores on the basis of those weights and the user’s preference is taken into consideration.
- Enhancing accuracy and quality of recommendations
- Personalisation and adaptation to specific requirements
- Storing a user’s previous preferences and ask for feedback about it to ensure continuous improvement

STAGE SIX: DEPLOYMENT

6.1 Plan Deployment

We have implemented the project using Google Colab as of now, where the user can enter his/her preferences and obtain a list of 10 restaurants which are most similar to the attribute values provided.

The file can be run on any system which has the mentioned dataset already loaded into it. Various runs of the program were carried out to ensure all kinds of test cases (especially the ones with rare values for any/all attributes) were covered.

6.2 Plan Monitoring and Maintenance

In addition to assessing feedback to decide upon improvements (if necessary), it would also be a good idea to consistently check and update the database which is used in order to incorporate newer additions and accommodate more flexibility in the system.

6.3 Produce Final Report

- Executive summary
- Project methodology
- Data collection and preprocessing
- Modeling techniques and implementation
- Evaluation and results
- Implementation of improvements based on feedback
- Conclusive note

6.4 Review Project

Hence, by the methodologies and techniques described above, this project has been implemented and executed successfully. In the future, recommendation systems can prove to be useful for businesses and consumers alike.