Recommender System and Sentimental Analysis

YELP Dataset

IE7275 – Data Mining in Engineering

Milestone: Project Report

Group 8

Sparsh Marwah

Sharvari Pravin Deshpande

Medhavi Uday Pande

(857) 225-9142 (Tel of Sparsh)

(857) 202-8669 (Tel of Sharvari)

(617) 595-0724 (Tel of Medhavi)

marwah.sp@northeastern.edu

deshpande.sha@northeastern.edu

pande.me@northeastern.edu

Index

Problem Setting

In the digital age, online review platforms such as Yelp have become integral to consumer decision-making processes, offering insights into the quality of businesses and services. However, this landscape is not without its challenges. The proliferation of fake reviews undermines the credibility of these platforms, potentially misleading consumers and damaging the reputation of businesses. Moreover, the sheer volume of available options can overwhelm users, necessitating the development of personalised recommendation systems to enhance their experience and facilitate informed choices.

Problem Definition

Within this context, our project aims to address two specific challenges:

1. **Sentimental Analysis:** The aim is to perform sentiment analysis to classify the reviews as positive or negative.
2. **Building a Recommender System:** Our objective is to design and implement a recommender system that leverages user preferences and past choices to recommend the best items or places. By tailoring recommendations to individual tastes, we aim to streamline the decision-making process for users.

By tackling these challenges, our research endeavours to enhance the reliability and credibility of online review platforms like Yelp. Additionally, we aim to empower consumers with personalised recommendations, fostering informed decision-making and bolstering trust in the digital marketplace.

Data Source & Data Description

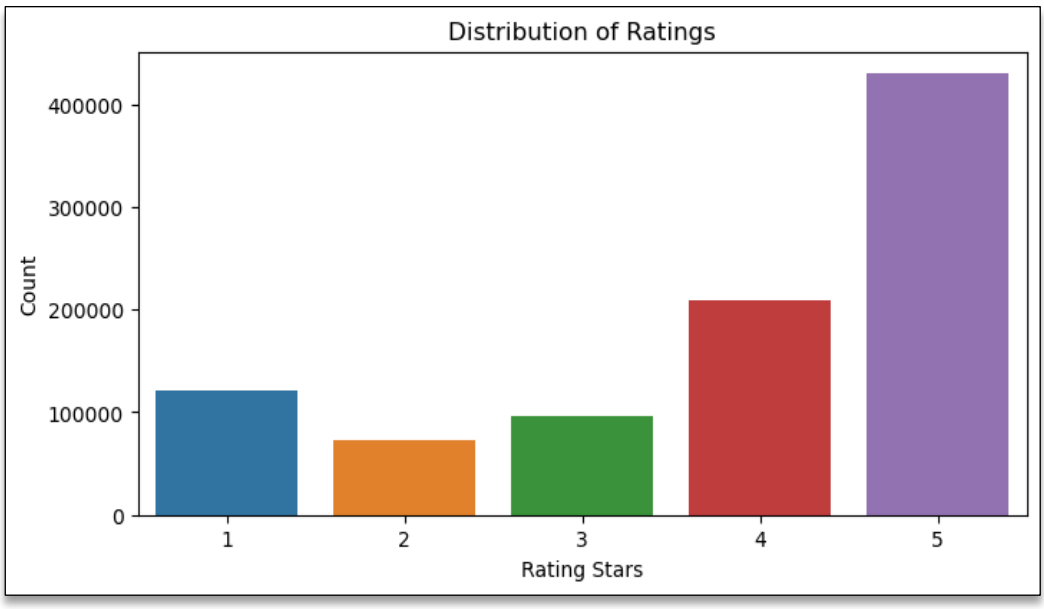
https://www.yelp.com/dataset

This collection includes information about a variety of businesses. Each entry contains information such as the business ID, name, address, location (latitude and longitude), star rating, review count, date, review_id, user_id, business_id, stars, useful, funny, cool, text, complement_count, and attributes (such as whether they accept credit cards or have specific amenities), categories (business types), and hours of operation. The companies vary from medical services and retail outlets to restaurants and beauty salons.

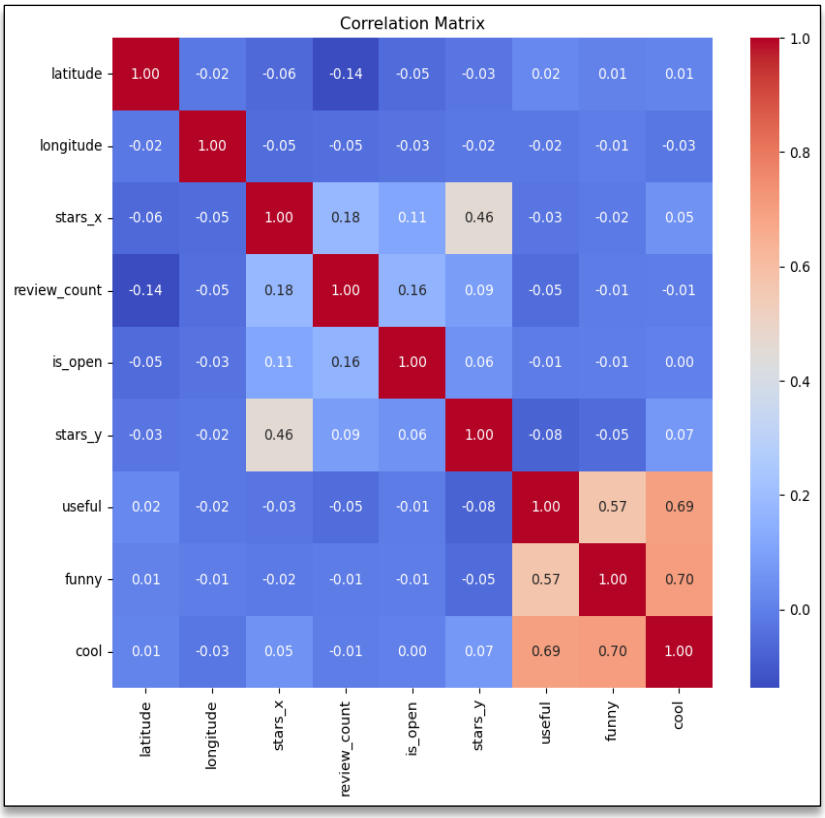
Data Exploration

We started with the statistical summary of numeric features. For exploring the data, we generated some visualisations.

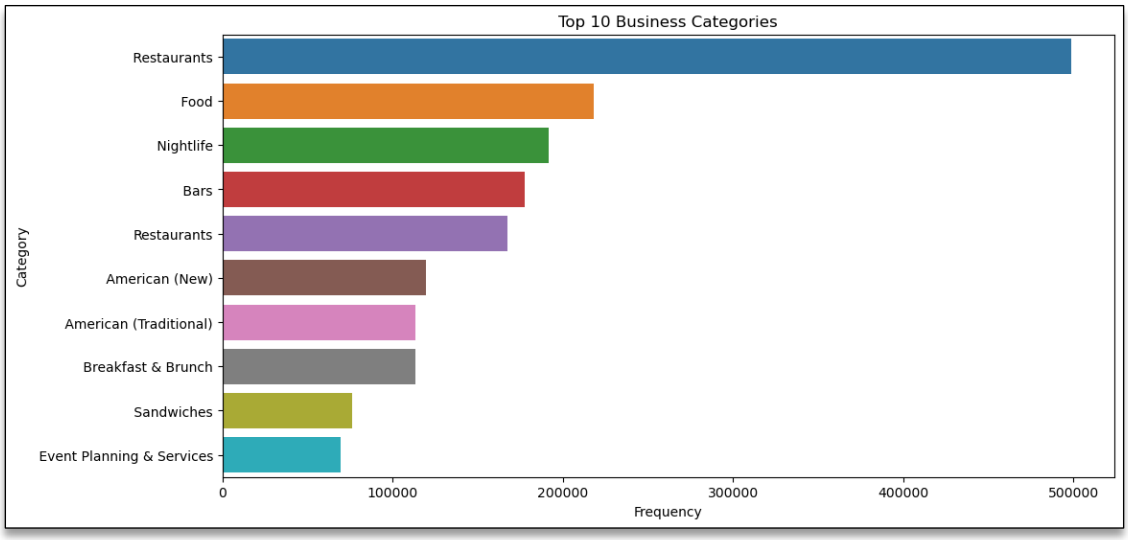
1. **Histogram of ratings:** This visualisation provides insight into the distribution of ratings across different levels, from 1 to 5 stars. By analysing this graph, we can discern trends such as whether there is a skew towards positive or negative reviews, whether there's a prevalence of extreme ratings, or if there's a balanced distribution of ratings. Understanding these patterns can offer valuable insights into customer sentiments and preferences.



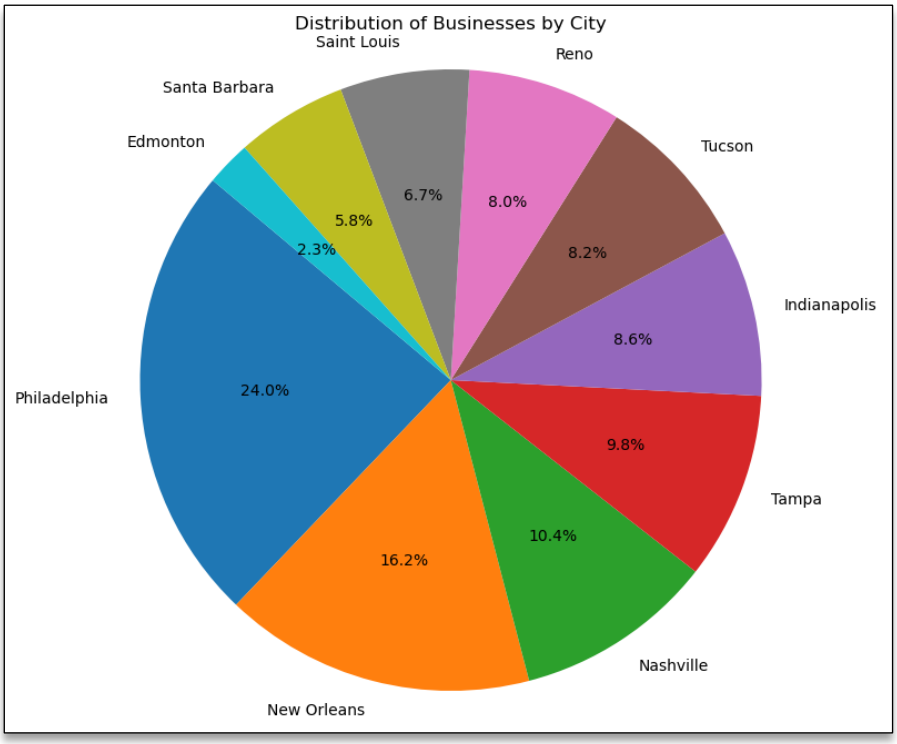
2. **Correlation Matrix:** The correlation matrix offers a comprehensive view of the relationships between various variables in the dataset. It helps identify whether there are linear relationships between different features, the strength and direction of these relationships (positive or negative correlation), and potential patterns or dependencies within the data. This analysis aids in understanding how different factors influence each other and guides further exploration or modelling.



3. **<u>Histogram for top categories of businesses:</u>** This visualisation delves into the distribution of reviews across different business categories, highlighting which categories attract the most attention from reviewers. By identifying the most-reviewed categories, such as restaurants and food establishments, we gain insights into consumer preferences and behaviours. This analysis also suggests potential strategies for expanding the dataset by focusing on categories with fewer reviews, thus enhancing the breadth and depth of recommendations over time.

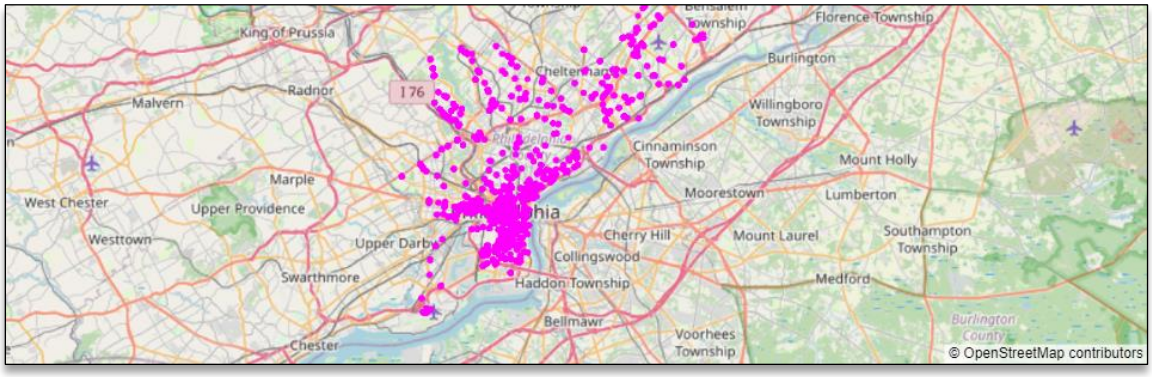


4. **<u>Distribution of Business by City:</u>** The pie chart illustrates the geographical distribution of reviews by city, shedding light on regional variations in reviewer activity. By pinpointing cities with a lower volume of reviews, businesses can prioritise marketing efforts in these areas to bolster platform popularity and increase dataset diversity. This strategic approach not only fosters business expansion but also improves the accuracy of recommendations and facilitates the detection of fake reviews by capturing a broader spectrum of consumer experiences.



5. **<u>Plot restaurant names with location data, star rating, and food category:</u>** This geospatial visualisation overlays restaurant locations with additional information such as star ratings and food categories, providing a spatial perspective on review distribution and preferences. By mapping out areas with high and low review densities, businesses can identify geographical patterns in consumer behaviour and tailor marketing strategies accordingly. This visual exploration aids in understanding regional variations in culinary preferences

and enables targeted interventions to enhance customer engagement and satisfaction.



Data Mining Tasks

In our process, we undertook several data cleaning operations to prepare the dataset for further analysis.

1. **Importing and Reading Data:** The initial step involved importing the dataset into our environment and reading its contents. This allowed us to gain a preliminary understanding of the data structure and format.
2. **Handling Missing Values:** Addressing missing values is essential to prevent biassed analysis and ensure the accuracy of results. We implemented techniques to fill in missing values appropriately, thereby maintaining the integrity of the dataset.
3. **Handling Nested Attributes:** The dataset contained nested attributes within the "attributes" column, which required special treatment for effective analysis. To facilitate this, we developed a method to extract these nested attributes and transform them into their own individual columns. This process involved two key functions:a. Extracting Keys from Nested Dictionary (extract_keys): This function was designed to extract keys from the nested dictionary structure within the "attributes" column. By identifying and isolating these keys, we were able to segregate the nested attributes for further processing.b. Converting String to Dictionary (str_to_dct): The purpose of this function was to convert string representations of dictionaries into actual dictionary objects. This transformation was23 necessary to access and manipulate the nested attributes effectively during the data processing stage.
4. **Handling Categories:** The dataset contained a “categories” column. We had to encode it to 0s and 1s. This we did by making use of pd.series() and get_dummies() functions.

By employing these functions, we were able to systematically parse and separate the nested attributes, ultimately creating a structured feature table that facilitated comprehensive analysis and exploration of the dataset. This preprocessing stage laid the foundation for subsequent data mining tasks, enabling us to derive meaningful insights and patterns from the data with accuracy and efficiency.

Data Mining Models/Methods

We divided the modelling part into 2. The first one being the recommender system and the second one is the sentimental analysis.

RECOMMENDER SYSTEM

For the recommender system, we used two approaches, one was based on content-based filtering while the other was based on collaborative filtering.

1. **Content-based Filtering:** Content-based filtering uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback. Its limitation is that it does not capture any information about user's preferences and considers only restaurant features only.

   Under content-based filtering, we deployed 1 supervised machine learning model K-Nearest Neighbors (KNN).
2. **Collaborative Filtering:** Uses similarities between users and items simultaneously to provide recommendations.

   For this type of filtering, we trained the model using 2 supervised machine learning algorithms, being Singular Value Decomposition model (SVD), Neural Network - Keras.

Lastly, we compared the models on performance evaluation parameters.

K-Nearest Neighbors (KNN)

1. **Data Splitting:** The dataset is divided into a training set and a test set, with 80% of the data allocated for training and 20% for testing.
2. **Model Instantiation and Fitting:** The KNN model is instantiated and trained using the training set.
3. **Model Testing:** To evaluate the model's performance, the last row of the dataset is designated as a validation set, separate from the training data. This row is not used in the model training process.

4. **Distance Calculation:** After fitting the KNN model to the validation set, distances between the validation set and other restaurants in the dataset are computed based on their similar features.
5. **Recommendation Table Creation:** A result table is generated, displaying restaurants similar to the validation set based on their distances. In this recommendation system, shorter distances indicate greater similarity to the validation set.
6. **Initial Recommendations:** An example recommendation is provided based on the generated result table. For instance, "Adelita Taqueria & Restaurant" is identified as a recommended restaurant, offering authentic Lebanese cuisine, specifically sandwiches. These recommendations are derived from the model's analysis of similarity in menu offerings and other features.

Singular Value Decomposition (SVD)

1. **Identifying Popular Restaurants:** We begin by identifying the restaurants with the highest ratings, which indicate their popularity among users.
2. **Determining the Most Popular Restaurant:** Once we've identified the top-rated restaurants, we pinpoint the specific restaurant that stands out as the most popular based on its ratings.
3. **Constructing the Utility Matrix:** To prepare the data for analysis, we create a Utility Matrix, also known as the User-Restaurant Matrix. This matrix captures the ratings given by each user to each restaurant. As users don't review every restaurant, this matrix may have many empty entries, forming a sparse structure.
4. **Generating the User-Item Matrix:** Next, we transform the Utility Matrix by transposing it, switching the roles of users and restaurants. In this new matrix, users are represented by columns, and restaurants are represented by rows.
5. **Analysing the Utility Matrix and Matrix Decomposition:** We examine the shape and properties of the original Utility Matrix and its transpose. Then, we decompose the matrix to prepare for further analysis.
6. **Compressing the Matrix using TruncatedSVD:** Utilising the Truncated Singular Value Decomposition (SVD) technique from the sklearn library, we compress the transposed matrix into smaller matrices containing only 12 rows.

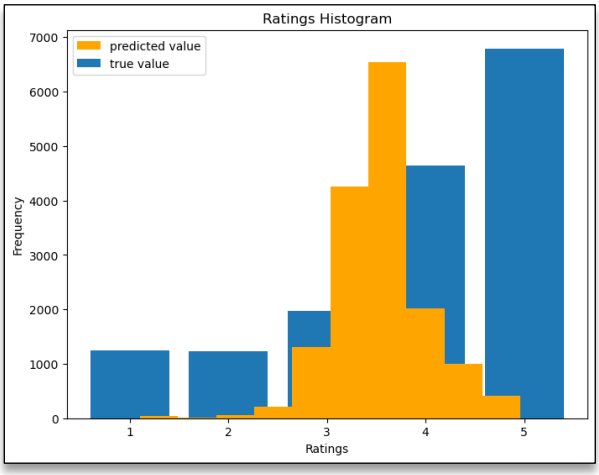
This compression condenses the representation of users' tastes into 12 components while retaining essential information. Subsequently, we generate a Correlation Matrix.

7. **Computing Correlation Coefficients:** We compute Pearson correlation coefficients for each pair of restaurants in the resulting Correlation Matrix. These coefficients measure the similarity in taste preferences between users.
8. **Identifying the Most Popular Restaurant:** From the Correlation Matrix, we isolate the most popular restaurant, such as "Village Whiskey." We then extract correlation values between this restaurant and all others.
9. **Recommending Highly Correlated Restaurants:** Finally, we recommend additional restaurants that exhibit high correlation with the most popular restaurant, "Village Whiskey." By applying specific conditions based on correlation values, we establish another recommender system, enhancing the range of recommendations for users.

Neural Network Model - Keras

1. **Data Preparation:** Initially, we duplicate the combined_business_data table and encode user and business IDs using LabelEncoder from sklearn. Unique users, restaurants, minimum and maximum ratings are stored in variables.
2. **Data Splitting:** The data is split into training and test sets to facilitate model evaluation.
3. **Model Configuration:** We define the number of factors per user/restaurant for the model. These factors determine the size of the embeddings representing users and restaurants.
4. **Embedding Representation:** Embeddings are utilized to represent each user and restaurant in the data. The dot product between user and restaurant vectors generates vectors of specified size to capture weights related to each user per restaurant.
5. **Model Training:** To enhance model performance, bias is added to each embedding. The output of the dot product is passed through a sigmoid layer and scaled using the minimum and maximum ratings in the data. TensorFlow backend is employed for model training.

6. **Prediction:** The trained model is utilized to predict ratings for the test dataset. Performance is assessed by comparing actual ratings with predictions.



7. **Cosine Similarity:** Cosine Similarity is computed to measure the similarity between restaurants. Embedding layers from the Keras model are extracted to compute cosine similarity via dot product.
8. **Normalisation and Representation:** Restaurant embeddings are normalised to ensure cosine similarity calculation. Unique restaurant IDs are extracted and embedded into 50 dimensions.
9. **Recommendation Generation:** A function is created to calculate cosine similarity between a target restaurant (e.g., "Village Whiskey") and other restaurants, resulting in a recommendation table. This table is sorted based on cosine similarity to generate restaurant recommendations.

SENTIMENT ANALYSIS

Sentiment analysis is a pivotal process conducted on the Yelp review dataset to discern the polarity of sentiments expressed within the reviews. The sentiment polarity, whether positive, negative, or neutral, serves as a key indicator for businesses to gauge the overall sentiment of their customers towards their products or services. By analysing these sentiments, businesses can gain invaluable insights into the perception of their brand, identify patterns in customer feedback, and pinpoint areas of strength or weakness in their offerings. Understanding the sentiment polarity allows businesses to not only assess customer satisfaction levels but also to detect emerging trends, anticipate customer preferences, and make informed decisions to enhance

their products or services. Moreover, by leveraging sentiment analysis, businesses can proactively address negative feedback, resolve customer concerns, and foster positive relationships with their clients, ultimately leading to improved customer retention and loyalty.

In the sentiment analysis process, we embark on a series of steps to ensure the robustness and accuracy of our analysis. Initially, we preprocess the raw text data by eliminating various forms of noise, including URLs, mentions, hashtags, and numerical digits, which may obscure the true sentiment of the text. Subsequently, we tokenize the text, a process of breaking down the text into individual words or tokens, facilitating further analysis. To enhance the quality of our text data, we employ techniques like stop word removal and lemmatization, which help standardise the text by reducing it to its base or root form.

Following data preprocessing, we harness the power of TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. This method converts the text into numerical features while accounting for the significance of words within the corpus. TF-IDF not only transforms the text into a format suitable for machine learning algorithms but also preserves the importance of words relative to the entire dataset.

Once the data is preprocessed and vectorized, we partition it into training and testing sets to facilitate model training and evaluation. Leveraging machine learning algorithms such as Logistic Regression, Random Forest, and XGBoost, we train models to classify the sentiment of the reviews into positive or negative categories. These models learn from the patterns inherent in the text data and make predictions based on learned features.

To assess the efficacy of our sentiment analysis models, we evaluate their performance using a range of metrics, including accuracy, precision, recall, and F1-score. These metrics provide valuable insights into the models' ability to correctly classify sentiments and their performance across different categories. By meticulously evaluating the models against these metrics, we ensure the reliability and effectiveness of our sentiment analysis process, empowering businesses to make informed decisions based on customer feedback and sentiment.

The below models have been used for our sentiment analysis part:

1) **Logistic Regression :** Logistic Regression is a fundamental algorithm in the realm of binary classification. It operates by estimating the probability that a given input belongs to a particular class using a logistic function. Despite its simplicity, Logistic regression is highly effective for binary classification tasks, making it a popular choice for sentiment analysis.

   Logistic Regression plays a crucial role in interpreting the textual features extracted from reviews to discern the underlying sentiment conveyed within the text. This process involves several key steps:

   1. **Textual Feature Extraction:** Before applying Logistic Regression, the textual content of reviews undergoes preprocessing, which includes tasks such as removing noise (e.g., URLs, mentions, hashtags), tokenization, removing stop words, and lemmatization. This preprocessing step ensures that the text data is clean and standardised for analysis. Once preprocessed, the text is transformed into numerical features using techniques like TF-IDF vectorization, which captures the importance of words in the text corpus.

   2. **Model Training:** Logistic Regression is then applied to the transformed textual features. During the training phase, the algorithm learns from labelled examples of reviews, where each review is associated with a sentiment label (e.g., positive or negative). Logistic Regression fits a logistic curve to the training data, which models the probability that a given review belongs to a particular sentiment class (e.g., positive sentiment). This curve represents the relationship between the input features (textual features extracted from reviews) and the likelihood of each sentiment class.

   3. **Prediction:** Once trained, the Logistic Regression model can be used to predict the sentiment of unseen reviews. For a given review, the model calculates the probability that it belongs to each sentiment class (e.g., positive or negative). The review is then assigned to the sentiment class with the highest probability. In the case of binary sentiment classification (positive or negative), the model categorises reviews as either positive or negative based on a predefined threshold probability.

4. **Evaluation:** After making predictions on a test set of reviews, the performance of the Logistic Regression model is evaluated using metrics such as accuracy, precision, recall, and F1-score. These metrics provide insights into how well the model generalises to unseen data and its ability to correctly classify reviews into the respective sentiment categories.

Score on training set: 92.620%
Score on test set: 91.248%

2) **Random Forest Classifier :** Random forest serves as a powerful and versatile classifier in sentiment analysis, leveraging the collective knowledge of multiple decision trees to accurately predict the sentiment expressed in reviews. Here's a detailed description of how Random Forest has been utilised.
In the context of the provided code, Random Forest serves as a pivotal component in the ;sentiment analysis pipeline. Here's a detailed description of how Random Forest is utilised:

1. **Model Implementation:** The code instantiates a Random Forest classifier using the `RandomForestClassifier` from the scikit-learn library. This classifier is capable of constructing an ensemble of decision trees during the training phase.

2. **Training Phase:** Once the classifier is initialised, it is trained on the preprocessed text data. Specifically, the textual features extracted from reviews are used as input, while the corresponding sentiment labels (positive or negative) serve as the target variable. During training, the Random Forest algorithm constructs multiple decision trees by randomly selecting subsets of features and data samples. Each decision tree is trained independently on a bootstrapped subset of the training data.

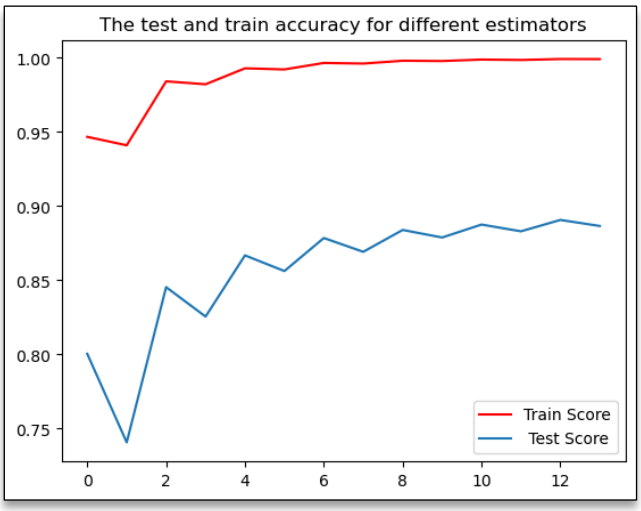
3. **Aggregation of Predictions:** After training, the Random Forest aggregates the predictions of individual decision trees to make a final prediction for each review. In the context of sentiment analysis, this involves combining the

predictions of multiple decision trees to determine the overall sentiment expressed in a review. The final prediction is typically based on a majority voting scheme, where the sentiment with the most votes across the ensemble of trees is selected as the predicted sentiment for the review.

4. **Robustness and Accuracy:** Random Forest is known for its robustness against overfitting and its ability to handle high-dimensional datasets effectively. By constructing multiple decision trees and aggregating their predictions, Random Forest can capture complex patterns within the text data while mitigating the risk of overfitting. This leads to enhanced predictive accuracy and generalisation performance, making Random Forest a suitable choice for sentiment analysis tasks, particularly when dealing with diverse and intricate textual data.

5. **Evaluation and Performance:** Following prediction, the performance of the Random Forest model is evaluated using metrics such as accuracy, precision, recall, and F1-score. These metrics provide insights into the model's ability to correctly classify reviews into the respective sentiment categories and assess its overall predictive performance on unseen data.

Score on training set: 99.917%
Score on test set: 89.069%



3) **XGBoost :** Extreme Gradient Boosting plays a crucial role as a high-performance classifier for sentiment analysis.

1. **Model Implementation:** The code employs the XGBoost classifier using the `XGBClassifier` from the `xgboost` library. XGBoost is chosen for its exceptional performance, scalability, and efficiency in handling large-scale datasets.

2. **Gradient Boosting Algorithm:** XGBoost sequentially builds an ensemble of decision trees, where each subsequent tree learns from the errors of its predecessors. During training, XGBoost optimises a loss function by adding new trees that minimise the residual errors from the previous iterations, resulting in a highly accurate predictive model.

3. **Optimization Techniques:** XGBoost incorporates various optimization techniques such as gradient descent optimization, parallel computing, and tree pruning to enhance its performance and speed. These techniques enable XGBoost to efficiently handle complex relationships and subtle nuances present in the text data, leading to superior predictive accuracy.

4. **Regularization Strategies:** XGBoost implements regularisation techniques such as shrinkage (learning rate) and tree depth constraints to prevent overfitting and improve generalisation performance. By controlling the complexity of individual trees and the overall model, XGBoost achieves a balance between bias and variance, resulting in robust and reliable predictions.

5. **Feature Importance:** XGBoost provides valuable insights into feature importance, allowing users to identify the most influential features contributing to the predictive performance. This feature enables practitioners to interpret the model's decision-making process and prioritise relevant features for further analysis.

6. **Scalability and Speed:** XGBoost is highly scalable and efficient, capable of handling large datasets with millions of samples and features. Its parallelized

implementation leverages multi-core processing and distributed computing frameworks, enabling rapid training and inference on massive datasets.

7. **Evaluation and Performance:** Similar to other classifiers, the performance of the XGBoost model is evaluated using standard evaluation metrics such as accuracy, precision, recall, and F1-score. These metrics provide insights into the model's ability to accurately classify reviews into positive or negative sentiments and assess its overall predictive performance on unseen data.

Score on training set: 94.794%
Score on test set: 90.775%

Since logistic regression directly models the probability of a binary outcome based on the input features, it provides clear insights into how each feature contributes to the sentiment classification. Additionally, logistic regression tends to perform well when the relationship between features and the target variable is relatively linear or when the dataset is not excessively complex. Given that sentiment analysis often involves analysing the presence or absence of certain words or phrases in the text, logistic regression's linear decision boundary may effectively capture the sentiment patterns in the data. However, the performance of each model can vary depending on the specific characteristics of the dataset, and it's essential to empirically evaluate and compare the models to determine the most suitable one for a given task.

Performance Evaluation

Below are the performance metrics for sentiment analysis models:

1. **Logistic Regression:**
   - Precision: 0.924
     - Precision indicates the accuracy of the positive predictions made by the model. A higher precision suggests that the model correctly identifies positive reviews with greater confidence.
   - Recall: 0.914
     - Recall measures the ability of the model to capture all positive instances in the dataset. A higher recall indicates that the model can effectively identify a larger proportion of positive reviews.
   - F1 Score: 0.917
     - The F1 score is the harmonic mean of precision and recall, providing a balanced measure of the model's performance. A higher F1 score suggests that the model achieves a good balance between precision and recall.

2. **Random Forest:**
   - Precision: 0.889
   - Recall: 0.891
   - F1 Score: 0.890
   - Random Forest achieves slightly lower precision, recall, and F1 score compared to logistic regression. While it still performs well, it may not be as precise in identifying positive reviews and capturing all positive instances as effectively as logistic regression.

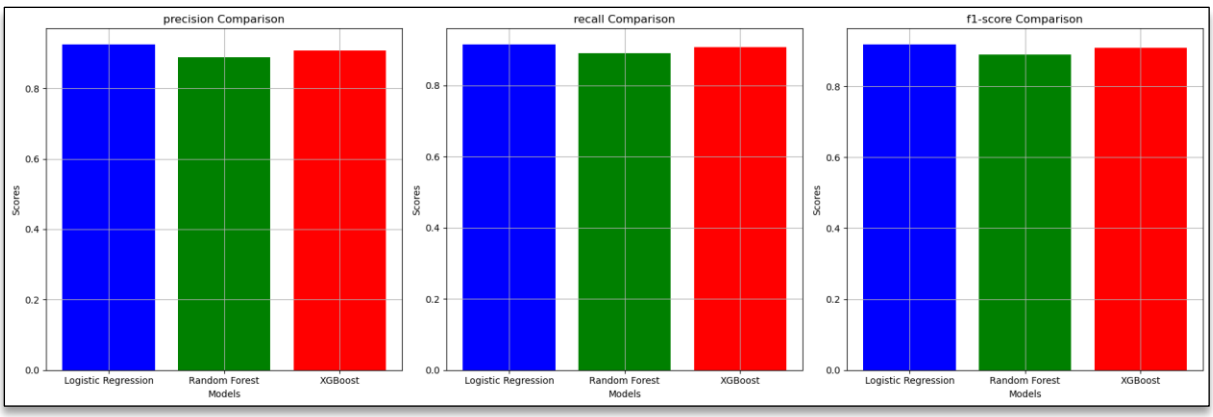
3. **XGBoost:**
   - Precision: 0.909
   - Recall: 0.908
   - F1 Score: 0.908
   - XGBoost performs slightly better than Random Forest but falls slightly behind logistic regression in terms of precision, recall, and F1 score. It demonstrates a

comparable performance to Random Forest, indicating that both ensemble methods can be effective for sentiment analysis tasks.

- **Logistic Regression:** With the highest precision, recall, and F1 score among the three models, logistic regression demonstrates its effectiveness in accurately identifying positive reviews and capturing a significant portion of positive instances in the dataset. Its balanced performance suggests that it can reliably distinguish between positive and negative sentiments.

- **Random Forest and XGBoost**: While Random Forest and XGBoost perform well, they fall slightly short compared to logistic regression in terms of precision, recall, and F1 score. However, the differences in performance between these models are relatively small, indicating that all three models are viable options for sentiment analysis tasks on this dataset.

Overall, logistic regression emerges as the best-performing model based on the metrics, showcasing its effectiveness in sentiment analysis tasks on the given Yelp review dataset.



Project Results

K-Nearest Neighbor:

	distance	index	name	stars
0	4.000000	3082	Los Taquitos de Puebla	4
1	4.123106	1762	The Pizza Place	3
2	4.123106	3061	Yummy Sushi	4
3	4.242641	1181	The Flavor Spot	4
4	4.358899	921	Artigiano Pizza	3

Singular Value Decomposition (SVD):

```
['Guavaberry Foods & Drinks ', 'Halal Food Special', 'Prince Pizza II']
```

Neural Network- Keras:

	similar_rest	cos
5	Village Whiskey	1.000000
378	Murphy's Tavern	0.523315
338	Poke Burri	0.441958
529	El Sarape Restaurant	0.428551
224	Green Basil Thai Kitchen	0.419238
300	Craft Hall	0.401216
603	Irwin's	0.399666
60	Federal Donuts	0.399084
194	Maria's Ristorante on Summit	0.386649
150	México Lindo	0.358753

Sentimental Analysis:

```
Prediction on an input string:  people that works here for sure is friendly! :)I do love that big menu book and see
ms like there are a lot of items to choose from. This is always nice as Vietnamese food is definitely more than pho
and more spring rolls.
Logistic Regression model: [1]
Random Forest model      : [1]
XGboost model            : [1]
```

```
Prediction on an input string: The waiting time was really long.
Logistic Regression model: [0]
Random Forest model      : [1]
XGboost model            : [0]
```

Impact of Project Outcomes

- We can extend the use of positive and negative reviews in sentimental analysis to detect fake reviews.
- The recommender system can be used by third party apps for their users to get close suggestions of restaurants that they have previously visited.