# PROJECT REPORT

# Sentiment Analysis on TripAdvisor Reviews

**CS6120 – Natural Language Processing**

**Submitted by: Group 7**

**Team Members**
Manasvi Tallapalli (002192133)
Pranavi Katikaneni (002776411)
Suprajasai Konegari (002209421)
Supriya Konegari (002209470)

Term and Year: Spring 2024
Submitted to: Prof. Uzair Ahmad

Submission Date: April 14, 2024

# Sentiment Analysis on TripAdvisor Reviews

## Introduction:

In the realm of travel, hotels play a pivotal role, and with the abundance of information available, new avenues for selecting the best accommodations have emerged. Reviews, often considered the bedrock for evaluating product quality, serve as a crucial determinant in the decision-making process. While platforms like Amazon and YouTube employ scaled rating systems, Tripadvisor takes a nuanced approach. In this project, we aim to perform sentiment analysis on Tripadvisor hotel reviews for the city of New Delhi. This analysis seeks to provide valuable insights to both consumers and business owners, offering a nuanced understanding of the sentiments expressed in the reviews.

## Project Context:

The hospitality and tourism industry heavily relies on customer feedback to gauge satisfaction levels and identify areas for improvement. TripAdvisor is one of the largest platforms for travel-related reviews, hosting millions of reviews covering a vast array of destinations and establishments worldwide. Analyzing sentiments expressed in these reviews can provide valuable information to businesses, enabling them to enhance customer experience, address issues promptly, and make informed decisions based on customer feedback.

## Problem Statement:

The primary objective of this project is to develop a robust sentiment analysis model capable of categorizing TripAdvisor reviews into predefined sentiment categories (Poor, Good, Excellent). The model will be trained using a labeled dataset of TripAdvisor reviews sourced from Kaggle, where each review is associated with a sentiment label. The key challenges and goals of this project include:

- **Text Preprocessing:** Cleaning and preparing textual data to ensure optimal model performance.
- **Exploratory Data Analysis:** Conducting an exploratory data analysis (EDA) to uncover interesting insights within the dataset.
- **Feature Extraction:** Extracting meaningful features from text for sentiment analysis.
- **Model Development:** Implementing and fine-tuning machine learning models to accurately classify reviews based on sentiment.
- **Performance Evaluation:** Evaluating model performance using appropriate metrics like accuracy, precision, recall, and F1-score.
- **Interpretation and Insights:** Gaining actionable insights from sentiment analysis results to assist businesses in making data-driven decisions.

## Data Source and Data Description:

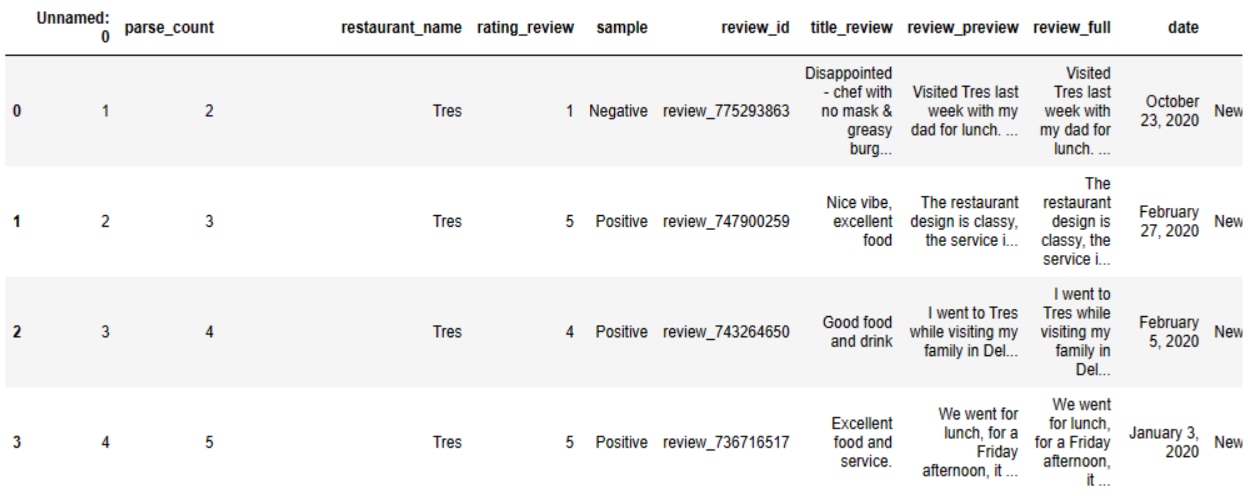
The dataset from kaggle contains 9 attributes and 148448 entries. The attributes are as below:

- **parse_count:** Auto incremental index of the web scraper.
- **restaurant_name:** The name of the restaurant associated with the review.
- **rating_review:** Integer review score within the range [1-5].
- **sample:** String representing "positive" samples [4-5] and "negative" ones [1-3].
- **review_id:** A unique identifier for each review.
- **title_review:** The title of the review.
- **review_preview:** A truncated preview of the review, often displayed on the website when the full text is very long.
- **review_full:** The complete review text.
- **date:** Timestamp indicating the publication date of the review in the format (day, month, year).

### Key Features:

- **Review Text (review_text):** Contains the main content of the review provided by the user.
- **Rating (rating):** Represents the numerical rating given by the user, reflecting their experience.

Below is a snippet of the data:

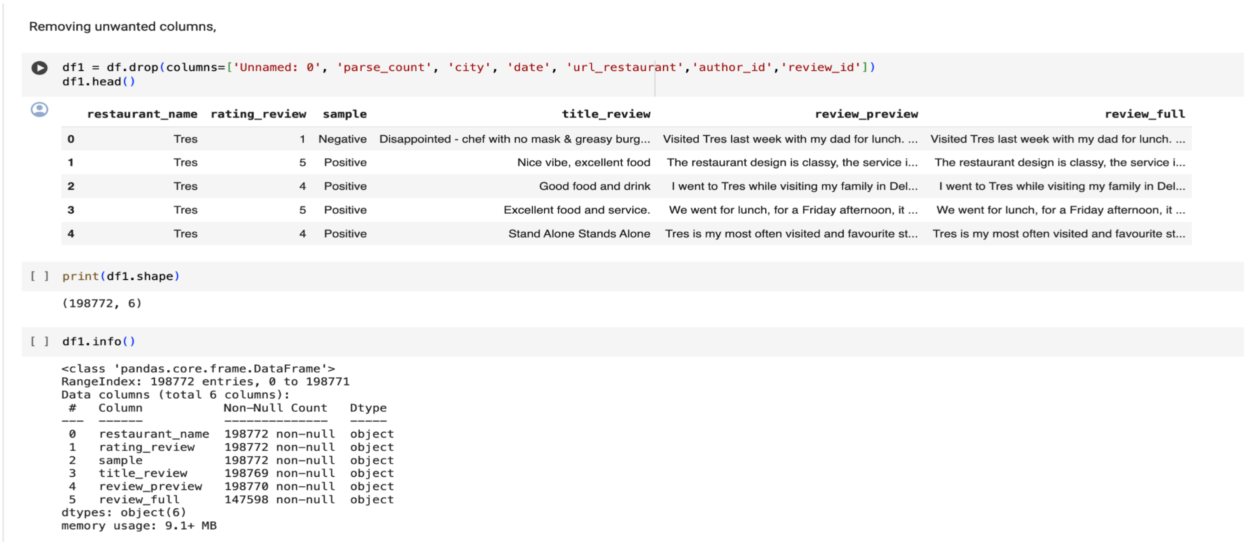


| | Unnamed: 0 | parse_count | restaurant_name | rating_review | sample | review_id | title_review | review_preview | review_full | date | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 2 | Tres | 1 | Negative | review_775293863 | Disappointed - chef with no mask & greasy burg... | Visited Tres last week with my dad for lunch. ... | Visited Tres last week with my dad for lunch. ... | October 23, 2020 | New |
| **1** | 2 | 3 | Tres | 5 | Positive | review_747900259 | Nice vibe, excellent food | The restaurant design is classy, the service i... | The restaurant design is classy, the service i... | February 27, 2020 | New |
| **2** | 3 | 4 | Tres | 4 | Positive | review_743264650 | Good food and drink | I went to Tres while visiting my family in Del... | I went to Tres while visiting my family in Del... | February 5, 2020 | New |
| **3** | 4 | 5 | Tres | 5 | Positive | review_736716517 | Excellent food and service. | We went for lunch, for a Friday afternoon, it ... | We went for lunch, for a Friday afternoon, it ... | January 3, 2020 | New |

## Methodology:

## 1. Exploratory Data Analysis (EDA):

- Data Collection and Cleaning:
    - We streamlined the DataFrame df1 to include only essential attributes pertinent to sentiment analysis, such as review text and ratings.
    - Removing extraneous columns, such as metadata or identifiers, ensures that our analysis remains focused on the most relevant data for extracting sentiment insights.



Removing unwanted columns,

```
df1 = df.drop(columns=['Unnamed: 0', 'parse_count', 'city', 'date', 'url_restaurant','author_id','review_id'])
df1.head()
```

| | restaurant_name | rating_review | sample | title_review | review_preview | review_full |
|---|---|---|---|---|---|---|
| 0 | Tres | 1 | Negative | Disappointed - chef with no mask & greasy burg... | Visited Tres last week with my dad for lunch. ... | Visited Tres last week with my dad for lunch. ... |
| 1 | Tres | 5 | Positive | Nice vibe, excellent food | The restaurant design is classy, the service i... | The restaurant design is classy, the service i... |
| 2 | Tres | 4 | Positive | Good food and drink | I went to Tres while visiting my family in Del... | I went to Tres while visiting my family in Del... |
| 3 | Tres | 5 | Positive | Excellent food and service. | We went for lunch, for a Friday afternoon, it ... | We went for lunch, for a Friday afternoon, it ... |
| 4 | Tres | 4 | Positive | Stand Alone Stands Alone | Tres is my most often visited and favourite st... | Tres is my most often visited and favourite st... |

```
print(df1.shape)
```

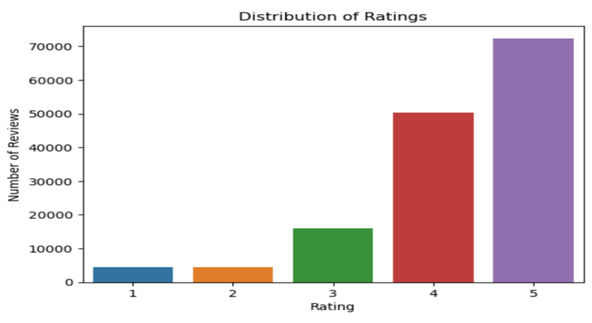
```
(198772, 6)
```

```
df1.info()
```

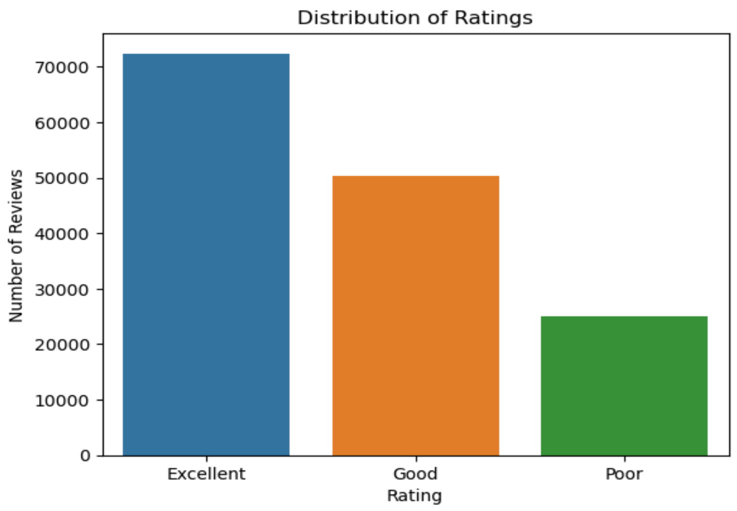
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 198772 entries, 0 to 198771
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype 
---  ------           --------------   ----- 
 0   restaurant_name  198772 non-null  object
 1   rating_review    198772 non-null  object
 2   sample           198772 non-null  object
 3   title_review     198769 non-null  object
 4   review_preview   198770 non-null  object
 5   review_full      147598 non-null  object
dtypes: object(6)
memory usage: 9.1+ MB
```

- Exploring Review Data:
    - Analyzed the distribution of ratings to understand sentiment distributions.

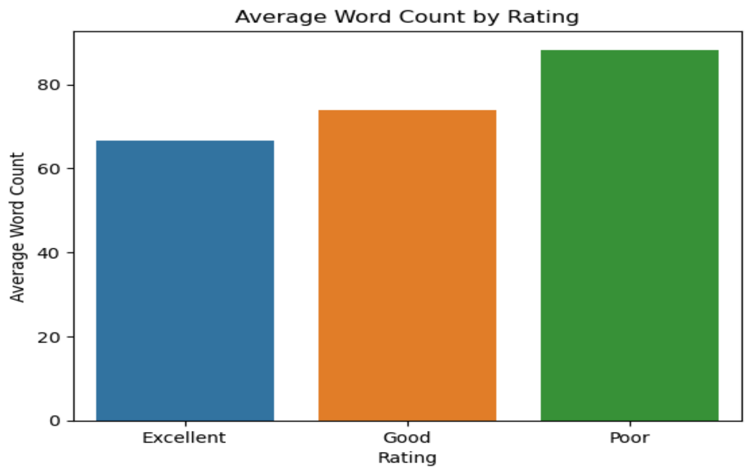
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.barplot(x=rating_counts.index, y=rating_counts.values)
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Number of Reviews')
plt.show()
```



From the above plot , there is clearly class imbalance .So, we merged ratings 1,2,3 as ‘poor’ , 4 as ‘good 'and 5 as ‘excellent’ and derived another attribute ‘rating_category’.



- Analyzed the Average Word Count by Rating

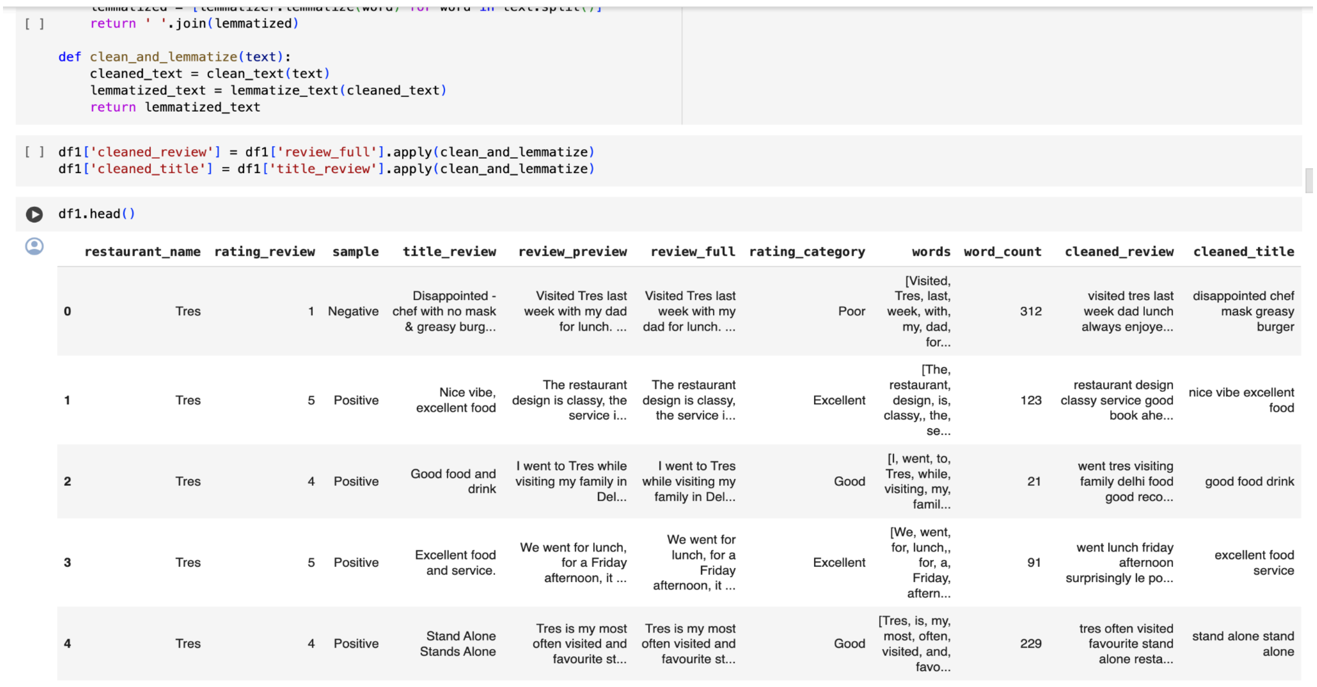


## 2. Data Preprocessing:

- <u>Text Analysis:</u>
  1) **Text Cleaning:**
  - Stopwords Removal**:** Removed common non-informative words.
  - Lowercasing**:** Converted text to lowercase for consistency.
  - Removing Numbers**:** Eliminated numerical digits from the text.
  2) **Noise Reduction:**
  - URLs and HTML Tags Removal: Stripped out URLs and HTML tags.

- Removing Emojis, Punctuation, and Non-ASCII Characters: Cleaned text of emojis, punctuation, and non-ASCII characters.

3) **Enhanced Standardization:**
   - Removing Extra White Spaces: Eliminated extra white spaces for uniformity.
   - Lemmatization: Reduced words to base forms for standardization and accuracy.
   - Modular Integration: Encapsulated cleaning and lemmatization into separate functions for readability and reuse.



```
    lemmatized = [lemmatizer.lemmatize(word) for word in text.split()]
    return ' '.join(lemmatized)

def clean_and_lemmatize(text):
    cleaned_text = clean_text(text)
    lemmatized_text = lemmatize_text(cleaned_text)
    return lemmatized_text
```

```
df1['cleaned_review'] = df1['review_full'].apply(clean_and_lemmatize)
df1['cleaned_title'] = df1['title_review'].apply(clean_and_lemmatize)
```

```
df1.head()
```

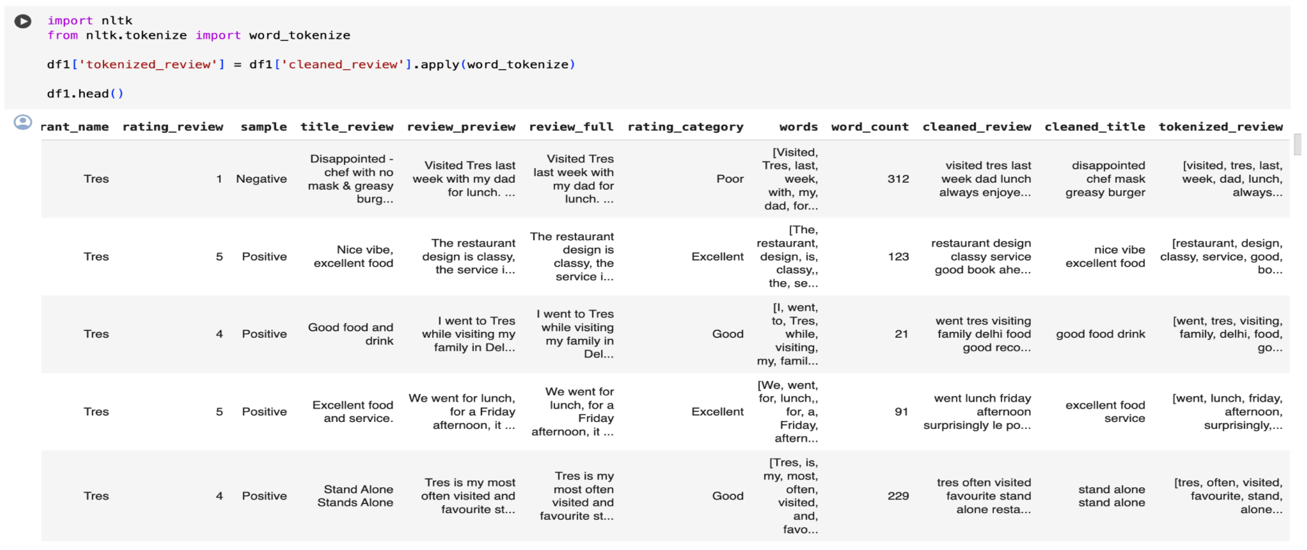
| | restaurant_name | rating_review | sample | title_review | review_preview | review_full | rating_category | words | word_count | cleaned_review | cleaned_title |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Tres | 1 | Negative | Disappointed - chef with no mask & greasy burg... | Visited Tres last week with my dad for lunch. ... | Visited Tres last week with my dad for lunch. ... | Poor | [Visited, Tres, last, week, with, my, dad, for... | 312 | visited tres last week dad lunch always enjoye... | disappointed chef mask greasy burger |
| 1 | Tres | 5 | Positive | Nice vibe, excellent food | The restaurant design is classy, the service i... | The restaurant design is classy, the service i... | Excellent | [The, restaurant, design, is, classy,, the, se... | 123 | restaurant design classy service good book ahe... | nice vibe excellent food |
| 2 | Tres | 4 | Positive | Good food and drink | I went to Tres while visiting my family in Del... | I went to Tres while visiting my family in Del... | Good | [I, went, to, Tres, while, visiting, my, famil... | 21 | went tres visiting family delhi food good reco... | good food drink |
| 3 | Tres | 5 | Positive | Excellent food and service. | We went for lunch, for a Friday afternoon, it ... | We went for lunch, for a Friday afternoon, it ... | Excellent | [We, went, for, lunch,, for, a, Friday, aftern... | 91 | went lunch friday afternoon surprisingly le po... | excellent food service |
| 4 | Tres | 4 | Positive | Stand Alone Stands Alone | Tres is my most often visited and favourite st... | Tres is my most often visited and favourite st... | Good | [Tres, is, my, most, often, visited, and, favo... | 229 | tres often visited favourite stand alone resta... | stand alone stand alone |

- <u>Tokenization:</u>

1) **Tokenization Process:**
   - NLTK Library Import: Imported NLTK for text processing capabilities.
   - Word Tokenization: Utilized NLTK's word_tokenize function for breaking down text into words.
   - Application to DataFrame: Applied word tokenization to the 'cleaned_review' column in DataFrame df1, generating tokenized lists for each entry.

2) **DataFrame Enhancement:**
   - New Column Creation: Created a new column named 'tokenized_review' in df1 to store tokenized text.
   - Tokenized Text Storage: Stored tokenized results as lists of words, corresponding to each 'cleaned_review' entry.

3) **Integration for Analysis:**
   - Preprocessing Integration: Integrated tokenization into the DataFrame to facilitate subsequent analysis tasks.
   - Enhanced Data Structure: Improved data structure by adding tokenized text, enabling more efficient text processing and analysis.

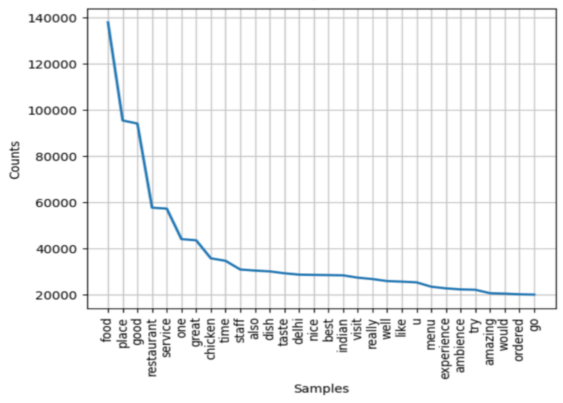


```
import nltk
from nltk.tokenize import word_tokenize

df1['tokenized_review'] = df1['cleaned_review'].apply(word_tokenize)

df1.head()
```

| rant_name | rating_review | sample | title_review | review_preview | review_full | rating_category | words | word_count | cleaned_review | cleaned_title | tokenized_review |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Tres | 1 | Negative | Disappointed - chef with no mask & greasy burg... | Visited Tres last week with my dad for lunch. ... | Visited Tres last week with my dad for lunch. ... | Poor | [Visited, Tres, last, week, with, my, dad, for... | 312 | visited tres last week dad lunch always enjoye... | disappointed chef mask greasy burger | [visited, tres, last, week, dad, lunch, always... |
| Tres | 5 | Positive | Nice vibe, excellent food | The restaurant design is classy, the service i... | The restaurant design is classy, the service i... | Excellent | [The, restaurant, design, is, classy,, the, se... | 123 | restaurant design classy service good book ahe... | nice vibe excellent food | [restaurant, design, classy, service, good, bo... |
| Tres | 4 | Positive | Good food and drink | I went to Tres while visiting my family in Del... | I went to Tres while visiting my family in Del... | Good | [I, went, to, Tres, while, visiting, my, famil... | 21 | went tres visiting family delhi food good reco... | good food drink | [went, tres, visiting, family, delhi, food, go... |
| Tres | 5 | Positive | Excellent food and service. | We went for lunch, for a Friday afternoon, it ... | We went for lunch, for a Friday afternoon, it ... | Excellent | [We, went, for, lunch,, for, a, Friday, aftern... | 91 | went lunch friday afternoon surprisingly le po... | excellent food service | [went, lunch, friday, afternoon, surprisingly,... |
| Tres | 4 | Positive | Stand Alone Stands Alone | Tres is my most often visited and favourite st... | Tres is my most often visited and favourite st... | Good | [Tres, is, my, most, often, visited, and, favo... | 229 | tres often visited favourite stand alone resta... | stand alone stand alone | [tres, often, visited, favourite, stand, alone... |

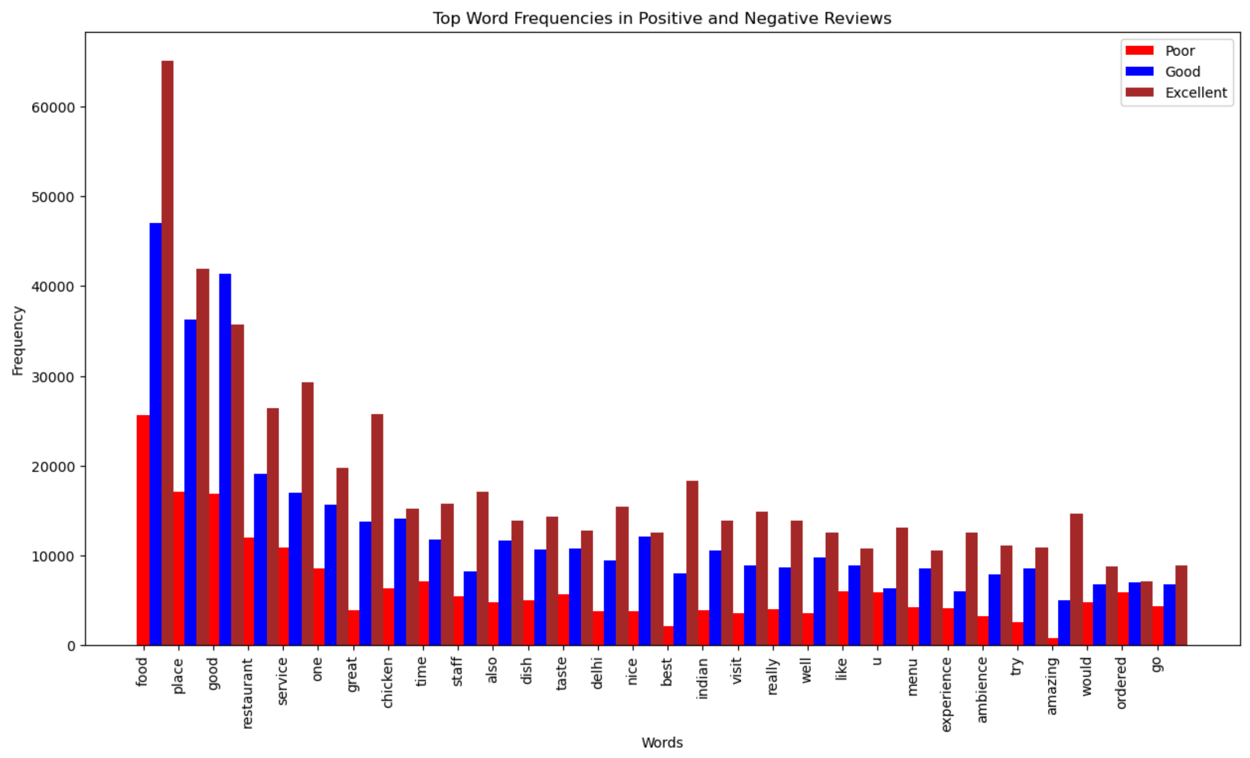
- <u>Word Frequency:</u>
  1) **Text Data Analysis:**
     - Tokenization: Employed NLTK's word_tokenize function to tokenize cleaned reviews.
     - Frequency Distribution: Calculated word frequency distribution across all reviews.
     - Top Words Identification: Identified and printed the 20 most common words with their frequencies.
  2) **Visualization:**
     - Frequency Distribution Plotting: Visualized frequency distribution by plotting the 30 most frequent words.

```
[('food', 137781), ('place', 95281), ('good', 93971), ('restaurant', 57515), ('service', 57112), ('one', 43898), ('great', 43402), ('chicken', 35580), ('time',
```



```
<Axes: xlabel='Samples', ylabel='Counts'>
```

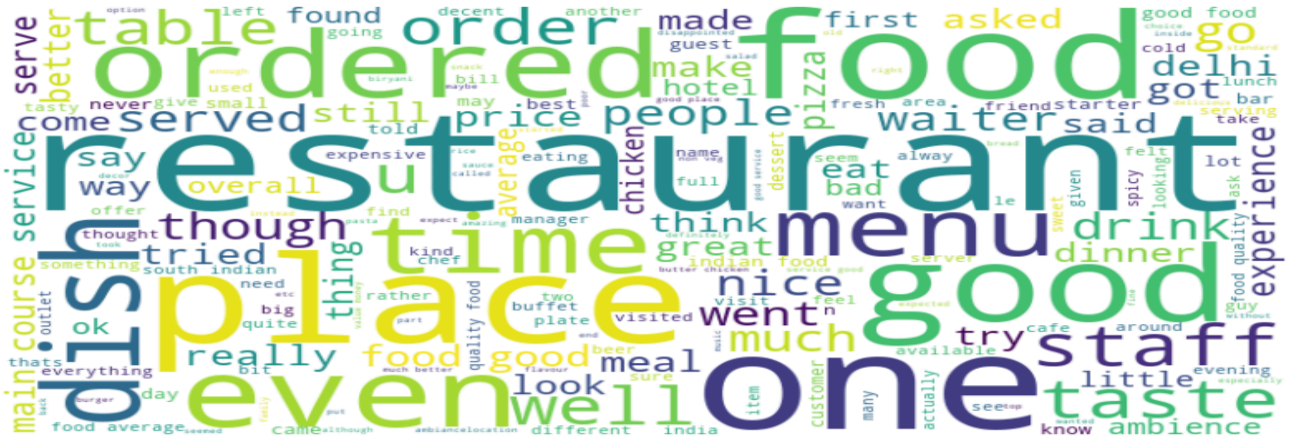
This is the frequency distribution of the most used words in the reviews.



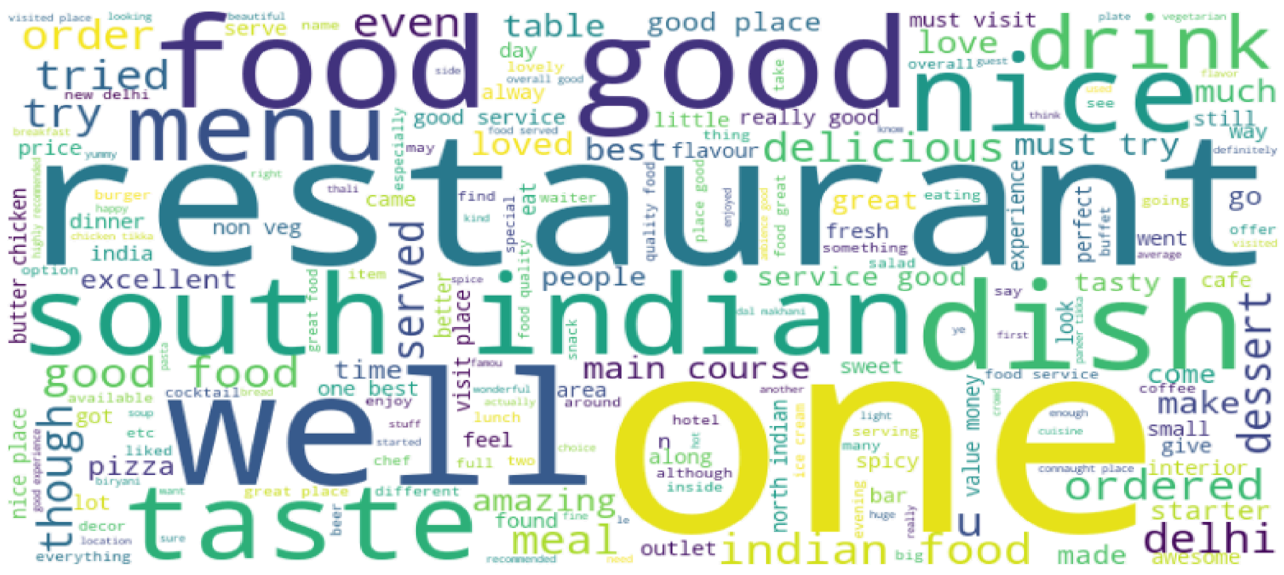
From the above graph, we can see the top words associated with different review categories.

- <u>Word Clouds:</u>
  1) **Word Cloud Analysis:**
  - ○ Categorization by Rating: Segmented reviews into 'Poor', 'Good', and 'Excellent' categories based on ratings.
  - ○ Word Cloud Generation: Utilized word clouds to visually represent the most frequent terms in each category.
  2) **Visualization Impact:**
  - ○ Understanding Customer Sentiment: Enhanced understanding of customer sentiments across varying satisfaction levels.
  - ○ Facilitated Analysis: Aided in effective analysis and interpretation of review data.

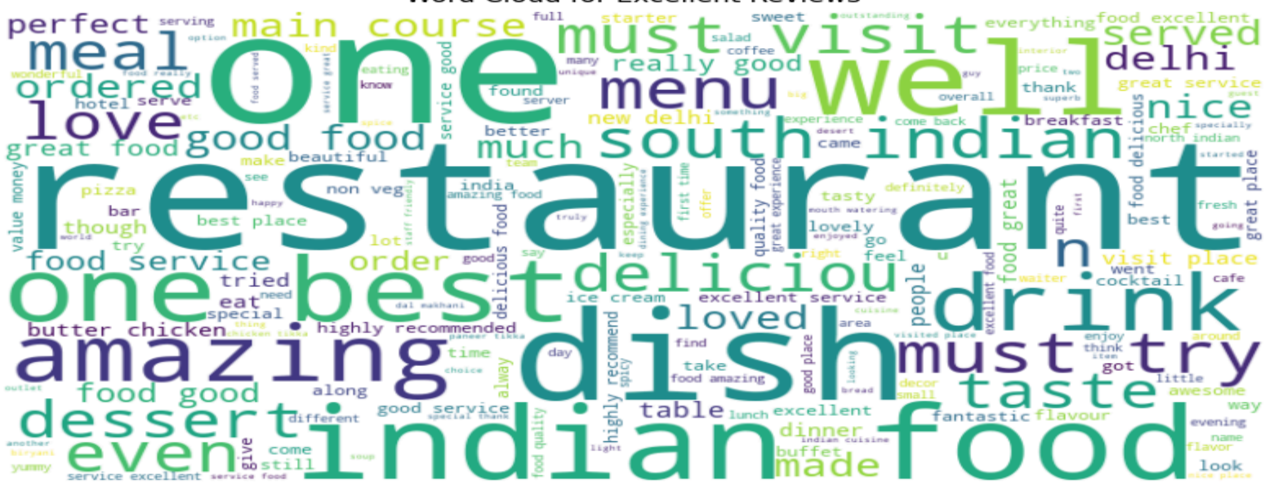
Word Cloud for Poor Reviews



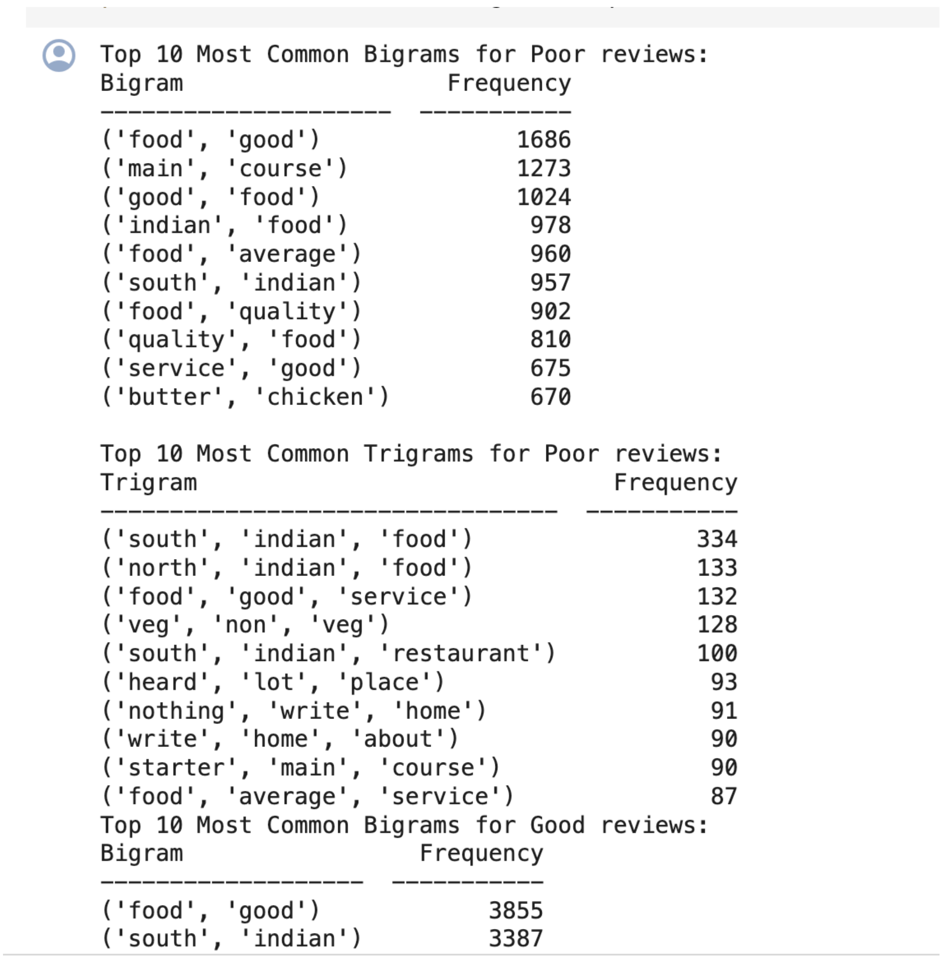
Word Cloud for Good Reviews



Word Cloud for Excellent Reviews



- Bigrams and Trigrams:
    1) **N-gram Frequency Analysis:**
    - Categorization by Rating: Segmented reviews into 'Poor', 'Good', and 'Excellent' categories based on ratings.
    - Bigram and Trigram Calculation: Generated and calculated frequency distributions of bigrams and trigrams for each category.
    2) **Insight Extraction:**
    - Common Phrase Identification: Identified top 10 most frequent bigrams and trigrams for each rating category.
    - Language Trends Overview: Provided concise overview of language trends associated with different satisfaction levels.

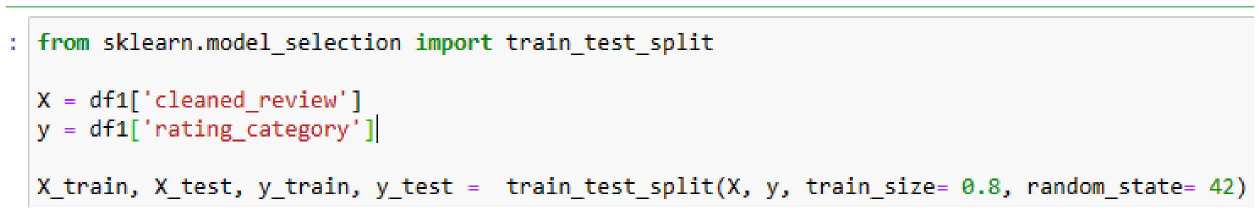


```
Top 10 Most Common Bigrams for Poor reviews:
Bigram                    Frequency
---------------------  -----------
('food', 'good')              1686
('main', 'course')            1273
('good', 'food')              1024
('indian', 'food')             978
('food', 'average')            960
('south', 'indian')            957
('food', 'quality')            902
('quality', 'food')            810
('service', 'good')            675
('butter', 'chicken')          670

Top 10 Most Common Trigrams for Poor reviews:
Trigram                              Frequency
---------------------------------  -----------
('south', 'indian', 'food')                334
('north', 'indian', 'food')                133
('food', 'good', 'service')                132
('veg', 'non', 'veg')                      128
('south', 'indian', 'restaurant')          100
('heard', 'lot', 'place')                   93
('nothing', 'write', 'home')                91
('write', 'home', 'about')                  90
('starter', 'main', 'course')               90
('food', 'average', 'service')              87
Top 10 Most Common Bigrams for Good reviews:
Bigram                  Frequency
-------------------  -----------
('food', 'good')            3855
('south', 'indian')         3387
```

- Train-Test Split:
    1) **Data Splitting for Model Training:**
    - Function Utilization: Utilized scikit-learn's train_test_split function to split dataset into training and testing sets.
    - Input-Output Consistency: Assigned cleaned reviews to X and corresponding rating categories to y, ensuring alignment between input features and target labels.

2) **Partitioning Parameters:**

- Training Size Specification: Specified training size of 80% to allocate majority of data for model training.
- Random State Assignment: Set random state to 42 for reproducibility, ensuring consistent results across different runs.



```
: from sklearn.model_selection import train_test_split

X = df1['cleaned_review']
y = df1['rating_category']

X_train, X_test, y_train, y_test =  train_test_split(X, y, train_size= 0.8, random_state= 42)
```

## 3. Feature Extraction:

- TF-IDF Vectorization:

1) **TF-IDF Vectorization:**

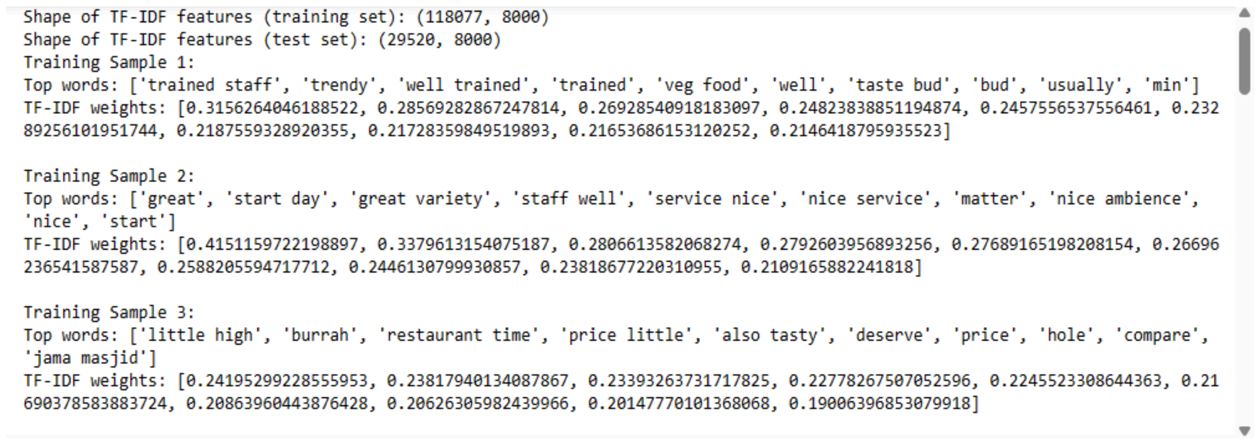
- Feature Extraction: Transformed textual data into numerical feature vectors using TfidfVectorizer.
- Weight Assignment: Assigned weights based on word frequency in each document (TF) and inverse frequency across all documents (IDF).
- Customization: Configured parameters like max_features and ngram_range to tailor the vectorization process.

2) **Data Transformation and Interpretation:**

- Consistent Transformation: Applied TF-IDF transformation consistently to both training and test sets for model compatibility.
- Feature Names Retrieval: Retrieved feature names to interpret TF-IDF weights and identify significant terms.
- Top Words Extraction: Extracted top words with highest TF-IDF weights to understand informative terms for classification.

3) **Post-Transformation Enhancements:**

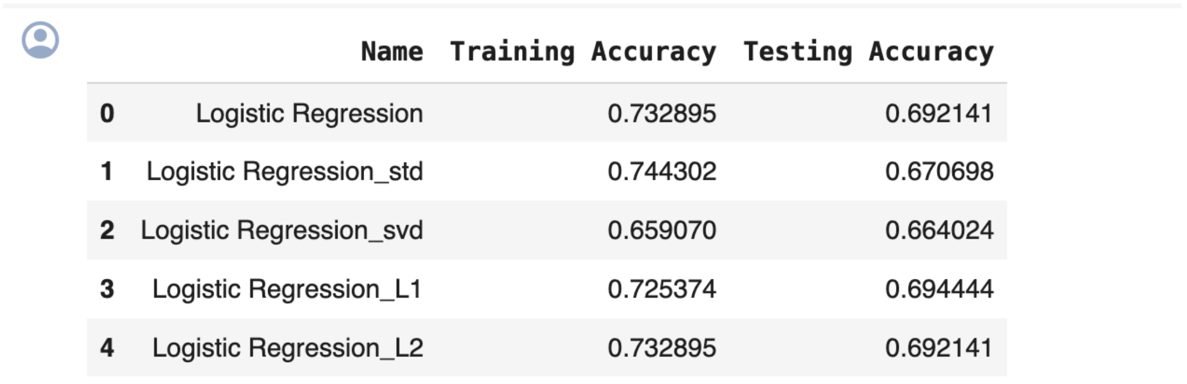
- Standardization: Ensured feature scaling uniformity post-TF-IDF vectorization to prevent dominance by particular features.
- Dimensionality Reduction: Utilized TruncatedSVD for dimensionality reduction, improving computational efficiency while preserving essential information for analysis.



```
Shape of TF-IDF features (training set): (118077, 8000)
Shape of TF-IDF features (test set): (29520, 8000)
Training Sample 1:
Top words: ['trained staff', 'trendy', 'well trained', 'trained', 'veg food', 'well', 'taste bud', 'bud', 'usually', 'min']
TF-IDF weights: [0.3156264046188522, 0.28569282867247814, 0.26928540918183097, 0.24823838851194874, 0.2457556537556461, 0.232
89256101951744, 0.2187559328920355, 0.21728359849519893, 0.21653686153120252, 0.2146418795935523]

Training Sample 2:
Top words: ['great', 'start day', 'great variety', 'staff well', 'service nice', 'nice service', 'matter', 'nice ambience',
'nice', 'start']
TF-IDF weights: [0.4151159722198897, 0.3379613154075187, 0.2806613582068274, 0.2792603956893256, 0.2768916519820154, 0.26696
236541587587, 0.2588205594717712, 0.2446130799930857, 0.23818677220310955, 0.2109165882241818]

Training Sample 3:
Top words: ['little high', 'burrah', 'restaurant time', 'price little', 'also tasty', 'deserve', 'price', 'hole', 'compare',
'jama masjid']
TF-IDF weights: [0.24195299228555953, 0.23817940134087867, 0.23393263731717825, 0.22778267507052596, 0.2245523308644363, 0.21
690378583883724, 0.20863960443876428, 0.20626305982439966, 0.20147770101368068, 0.19006396853079918]
```

## 4. Model Implementation:

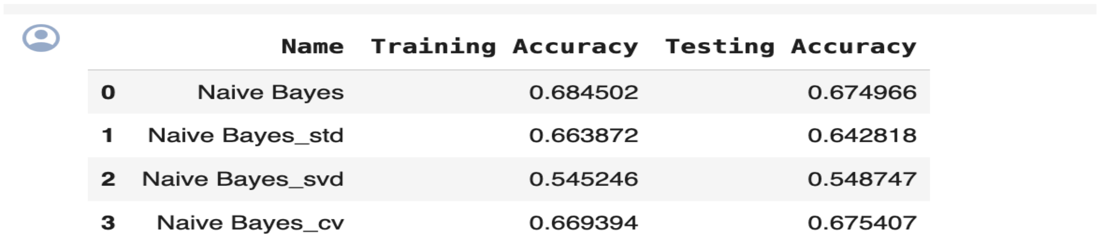
- **Logistic Regression:**
    - Implemented logistic regression for sentiment classification.
    - First, we applied logistic regression directly to the TF-IDF features.
    - Next, we applied logistic regression to the standardized features followed by the dimensionality reduced TF-IDF features.
    - We also used L1 and L2 regularization techniques to handle feature selection and reduce overfitting.
    - We trained the logistic regression model on the training dataset and evaluated its performance on the test dataset using standard metrics.



| | Name | Training Accuracy | Testing Accuracy |
|---|---|---|---|
| 0 | Logistic Regression | 0.732895 | 0.692141 |
| 1 | Logistic Regression_std | 0.744302 | 0.670698 |
| 2 | Logistic Regression_svd | 0.659070 | 0.664024 |
| 3 | Logistic Regression_L1 | 0.725374 | 0.694444 |
| 4 | Logistic Regression_L2 | 0.732895 | 0.692141 |

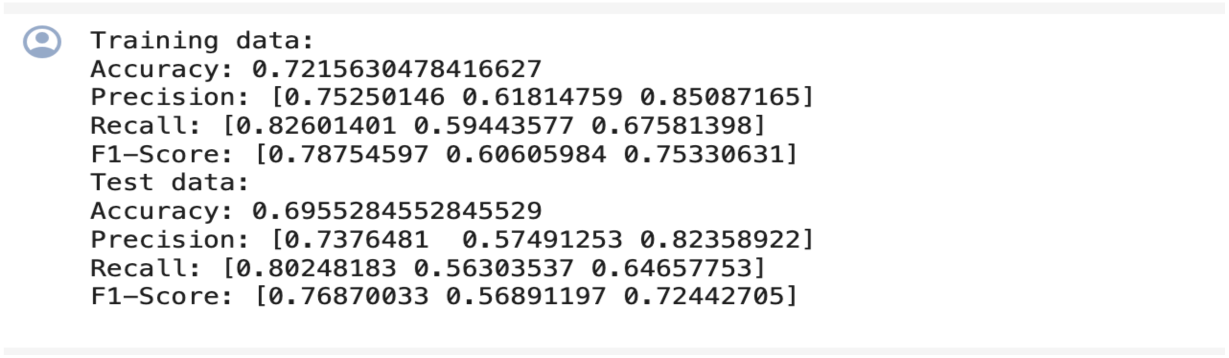
- **Naive Bayes Classifier:**
    - Implemented a Naive Bayes classifier for sentiment analysis.
    - First, we applied Naive Bayes classifier directly to the TF-IDF features.
    - Next, we applied Naive Bayes classifier to the standardized features followed by the dimensionality reduced TF-IDF features.
    - We performed cross-validation to assess the generalization performance of the Naive Bayes model.

  - We trained the Naive Bayes model on the training dataset and evaluated its performance on the test dataset using standard metrics.



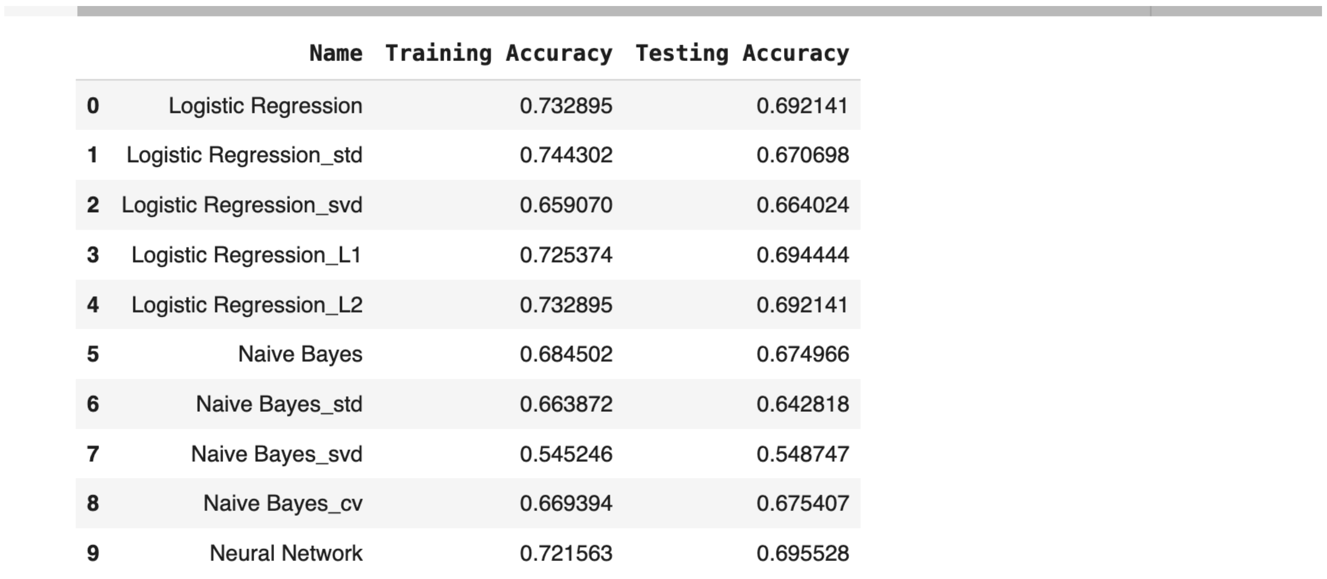
| | Name | Training Accuracy | Testing Accuracy |
|---|---|---|---|
| 0 | Naive Bayes | 0.684502 | 0.674966 |
| 1 | Naive Bayes_std | 0.663872 | 0.642818 |
| 2 | Naive Bayes_svd | 0.545246 | 0.548747 |
| 3 | Naive Bayes_cv | 0.669394 | 0.675407 |

- **Neural Networks:**
  - We explored different neural network architectures for sentiment analysis.
  - We initialized the MLP classifier with a single hidden layer of 100 neurons and trained it for 300 iterations.
  - We then  introduced L2 regularization by setting the alpha parameter to 0.0001 to mitigate overfitting.
  - Later, we enabled early stopping during training to prevent overfitting and improve generalization performance.
  - We trained the Neural Network model on the training dataset and evaluated its performance on the test dataset using standard metrics.



```
Training data:
Accuracy: 0.7215630478416627
Precision: [0.75250146 0.61814759 0.85087165]
Recall: [0.82601401 0.59443577 0.67581398]
F1-Score: [0.78754597 0.60605984 0.75330631]
Test data:
Accuracy: 0.6955284552845529
Precision: [0.7376481  0.57491253 0.82358922]
Recall: [0.80248183 0.56303537 0.64657753]
F1-Score: [0.76870033 0.56891197 0.72442705]
```

## Accuracies Comparison:



| | Name | Training Accuracy | Testing Accuracy |
|---|---|---|---|
| 0 | Logistic Regression | 0.732895 | 0.692141 |
| 1 | Logistic Regression_std | 0.744302 | 0.670698 |
| 2 | Logistic Regression_svd | 0.659070 | 0.664024 |
| 3 | Logistic Regression_L1 | 0.725374 | 0.694444 |
| 4 | Logistic Regression_L2 | 0.732895 | 0.692141 |
| 5 | Naive Bayes | 0.684502 | 0.674966 |
| 6 | Naive Bayes_std | 0.663872 | 0.642818 |
| 7 | Naive Bayes_svd | 0.545246 | 0.548747 |
| 8 | Naive Bayes_cv | 0.669394 | 0.675407 |
| 9 | Neural Network | 0.721563 | 0.695528 |

Picking top 2 models: Logistic Regression_L1 and Neural Network.

1) Performance of Logistic Regression:
    - Standard Logistic Regression and Logistic Regression with L2 regularization exhibit similar accuracies on both training and testing datasets.
    - While Logistic Regression with L1 regularization achieves slightly higher testing accuracy compared to others, it still maintains reasonable training accuracy.
    - Standardization and dimensionality reduction do not significantly improve model performance.
2) Performance of Naive Bayes:
    - Naive Bayes models demonstrate lower accuracy compared to Logistic Regression across all variants.
    - Standardization and dimensionality reduction do not substantially enhance Naive Bayes model performance.
3) Performance of Neural Network:
    - The Neural Network model achieves competitive accuracy on both training and testing datasets, comparable to Logistic Regression.

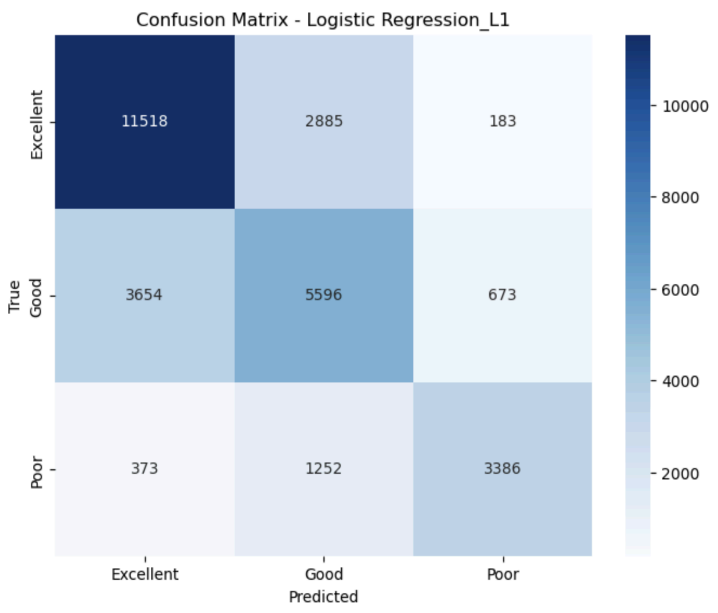
## F1 Scores Comparison:

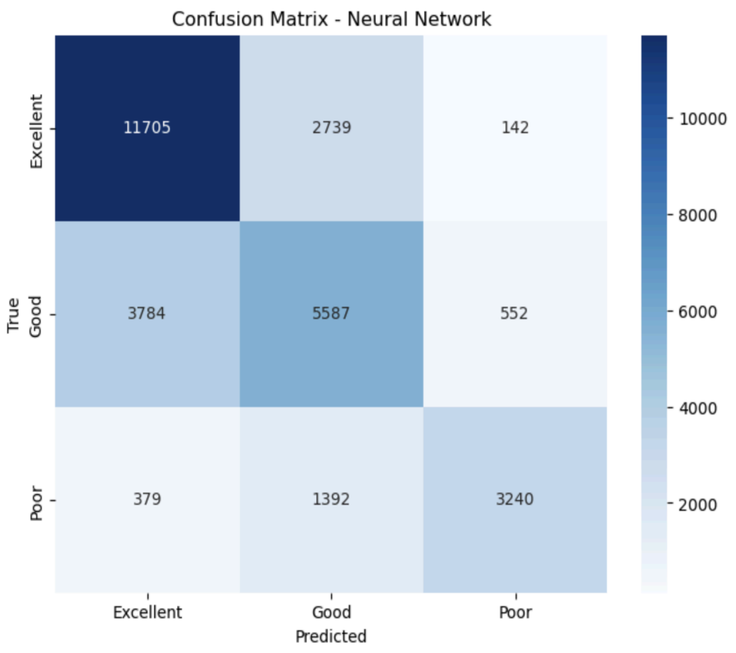
Logistic regression and neural networks show higher F1 scores when compared to Naive Bayes. Within logistic regression variations, both standard LR and LR with L1 regularization typically achieve slightly higher F1 scores, whereas LR with singular value decomposition (SVD) tends to achieve slightly lower F1 scores. Similarly, among Naive Bayes variations, NB without preprocessing generally achieves slightly higher F1 scores compared to NB with standardization, and NB with singular value decomposition (SVD) exhibits the lowest F1 scores.

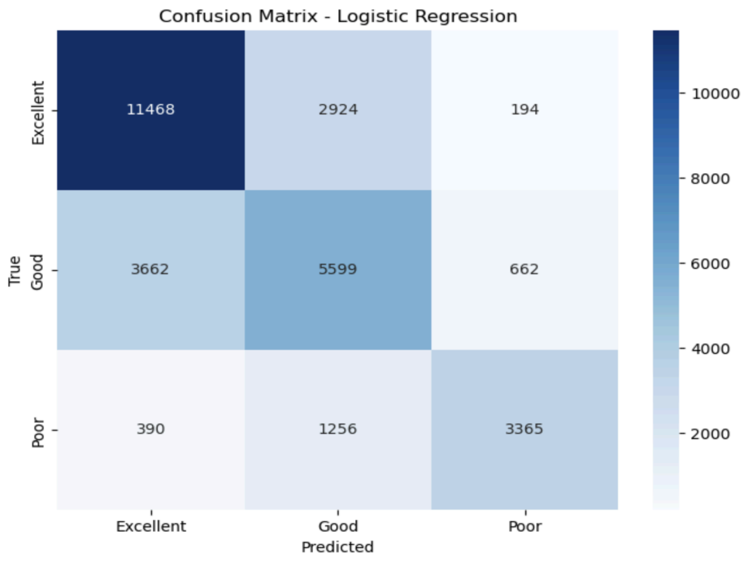
| | Excellent | Good | Poor |
|---|---|---|---|
| **LR training** | 0.79 | 0.63 | 0.77 |
| **LR testing** | 0.76 | 0.57 | 0.73 |
| **LR STD training** | 0.80 | 0.64 | 0.79 |
| **LR STD testing** | 0.75 | 0.54 | 0.69 |
| **LR SVD training** | 0.74 | 0.53 | 0.66 |
| **LR SVD testing** | 0.75 | 0.53 | 0.65 |
| **LR L1 training** | 0.79 | 0.62 | 0.76 |
| **LR L1 testing** | 0.76 | 0.57 | 0.73 |
| **LR L2 training** | 0.79 | 0.63 | 0.77 |
| **LR L2 testing** | 0.76 | 0.57 | 0.73 |
| **NB training** | 0.75 | 0.58 | 0.69 |
| **NB testing** | 0.75 | 0.56 | 0.67 |
| **NB STD training** | 0.72 | 0.59 | 0.68 |
| **NB STD testing** | 0.70 | 0.56 | 0.66 |
| **NB SVD training** | 0.65 | 0.44 | 0.49 |
| **NB SVD testing** | 0.65 | 0.44 | 0.49 |
| **NN training** | 0.79 | 0.60 | 0.75 |
| **NN testing** | 0.76 | 0.57 | 0.72 |

## Confusion Matrices:

Confusion Matrices help quantify model accuracy, precision, recall, and other metrics, enabling informed decision-making and continuous model refinement for effective sentiment analysis on TripAdvisor reviews.







Each confusion matrix above displays the counts of true positives, false positives, and false negative predictions for each class (Excellent, Good, Poor).

## Conclusion:

Based on the performance evaluation across sentiment categories, Logistic Regression with L1 regularization consistently outperforms other models, achieving high accuracies. Logistic Regression variants with standardization and dimensionality reduction show marginal improvement or no significant impact on accuracy. Logistic Regression with L2 regularization methods yield similar results to standard Logistic Regression, confirming its stability and versatility.

For sentiment analysis on TripAdvisor reviews, Logistic Regression emerges as the top choice due to its consistent high accuracy across all sentiment categories. While Naive Bayes and Neural Network models also show potential. Overall, Logistic Regression remains the preferred solution for reliable sentiment analysis in the hospitality domain.

## Future scope:

We will expand the scope of our analysis by incorporating aspect-based sentiment analysis, a technique that categorizes sentiments based on specific aspects or features mentioned in the reviews, such as service quality, cleanliness, and amenities. This approach will provide us with more detailed insights into customer feedback, allowing us to identify areas for improvement with greater granularity.

Additionally, we will implement sentiment trend analysis to monitor changes in customer sentiments over time. By analyzing historical review data, we will be able to identify trends and patterns, enabling us to adapt our strategies and services according to evolving customer preferences and expectations.

## References:

https://ieeexplore.ieee.org/abstract/document/8850982/authors#authors
https://ieeexplore.ieee.org/abstract/document/8908466