Comsats University Islamabad, Sahiwal Campus
Comsats Road, off GT Road, Sahiwal, Pakistan

Sentiment Analysis

By

MIRZA AHMAD AWAIS	**CIIT/SP20-BCS-016/SWL**
HUSNAIN JAVED	**CIIT/SP20-BCS-012/SWL**

Submitted To:

Mr Asif Imran

Table of Contents

1. Introduction

The rise of social media, online reviews, and digital communication platforms has led to an explosion of textual data available for analysis. Extracting valuable insights from this vast amount of text has become a significant challenge. Sentiment analysis, also known as opinion mining, is a field of natural language processing that focuses on automatically determining the sentiment expressed in textual data. Sentiment analysis also plays a vital role in social media monitoring, allowing companies to gauge public sentiment towards their brand and manage online reputation. Moreover, sentiment analysis has applications in political analysis, public opinion polling, and customer support analysis, making it an invaluable tool in today's data-driven world.

2. Problem Statement

The problem we are addressing is the need to efficiently analyse and classify sentiment in large volumes of textual data. By accurately determining the sentiment expressed in text, we can gain insights into public sentiment. Our goal is to build a sentiment analysis system that can process and classify text data accurately and efficiently. By accurately classifying sentiment, can gain insights into public opinion, and overall public perception. This, in turn, enables them to make informed decisions, identify emerging trends, and tailor their strategies to better meet people needs. These texts can vary in length, language style, and grammatical structure, making sentiment analysis a complex task.

3. Data Collection and Preprocessing

3.1 Data Sources

To collect data for our sentiment analysis project, we utilized the Kaggle platform as one of our data sources. When collecting data from Kaggle, we followed a systematic approach to ensure data relevance, quality, and suitability for our sentiment analysis project. Here is an outline of the data collection process:

Search for Relevant Datasets: We started by searching for datasets on Kaggle that specifically focused on sentiment analysis. We used relevant keywords such as "sentiment analysis," "opinion mining," or "text classification" to narrow down the search results.

Evaluate Dataset Descriptions and Metadata: After obtaining search results, we carefully reviewed the dataset descriptions and metadata to understand the nature of the data. This included checking the dataset size, the number of samples, the format of the text data, and the sentiment labelling scheme.

3.2 Data Preprocessing

We performed several preprocessing steps on the raw text data:

1. Removal of Punctuation Marks and Special Characters:

Punctuation marks and special characters, such as commas, periods, and exclamation marks, were removed from the text. These symbols do not contribute to sentiment analysis and can be safely eliminated.

2. Tokenization of Text:

Tokenization involves breaking down the text into individual words or sub word units. This step helps in analysing the sentiment of each word independently. We used various techniques such as whitespace tokenization or more advanced tokenizers like the Word Piece tokenizer for sub word unit tokenization.

3. Removal of Stop Words:

Stop words are common words that do not carry much meaning in sentiment analysis, such as "a," "the," "is," etc. We removed these stop words from the text to reduce noise and improve computational efficiency.

4. Lemmatization:

Lemmatization and stemming techniques were applied to normalize word forms. Lemmatization reduces words to their base or dictionary form (lemma), while stemming reduces words to their root form. These techniques help in reducing word variations and improving consistency in sentiment analysis.

5. Handling of Spelling Errors:

Spelling errors and abbreviations can hinder sentiment analysis accuracy. We implemented techniques to handle spelling errors, including spell checking and correction. We also expanded abbreviations to their full forms for better sentiment interpretation.

6. Handling of Capitalization:

To ensure consistency, we converted all text to lowercase. This step prevents the model from treating the same word with different capitalizations as different entities.

7. Removing HTML Tags:

In cases where the text was sourced from web pages or online forums, we removed any HTML tags, URLs, or other markup that may be present. This step ensures that only the actual textual content is considered for sentiment analysis.

8. Handling of Emotions or Emojis:

Emoticons and emojis can convey sentiment and meaning in text. We developed methods to handle emoticons and emojis, such as mapping them to sentiment-related terms or removing them if they were not relevant to the analysis.

9. Handling of Numerical Values:

Numerical values and symbols were either removed or converted to their textual representation to ensure consistency in sentiment analysis.

10. Handling of Rare or Infrequent Words:

Rare or infrequent words that do not occur frequently in the dataset may not contribute significantly to sentiment analysis. We applied techniques to identify and handle these rare words, such as excluding them from analysis or replacing them with a special token.

4. Methodology

4.1 Feature Extraction:

Feature extraction is a crucial step in sentiment analysis that involves transforming the textual data into a numerical representation suitable for machine learning algorithms. I have used one popular technique for feature extraction.

4.1.1 Term Frequency and Inverse Frequency Document (TFIFD):

TFIDF works by proportionally increasing the number of times a word appears in the document but is counterbalanced by the number of documents in which it is present. Hence, words like ‘this’, ’are’ etc., that are commonly present in all the documents are not given a very high rank. However, a word that is present too many times in a few of the documents will be given a higher rank as it might be indicative of the context of the document the text. Term Frequency measures the frequency of each term in a document. It is calculated as the number of occurrences of a term divided by the total number of terms in the document. IDF assigns higher weights to terms that are less common across the corpus, indicating their significance. TF-IDF is computed by multiplying the term frequency (TF) of a term in a document by its inverse document frequency (IDF). The resulting TF-IDF score represents the importance of the term in the document within the context of the entire corpus.

TF-IDF scores serve as features for sentiment analysis. Each document is represented as a vector of TF-IDF scores, with each dimension corresponding to a unique term in the corpus. These TF-IDF vectors capture the importance and relevance of different terms in the document, providing valuable information for sentiment classification.

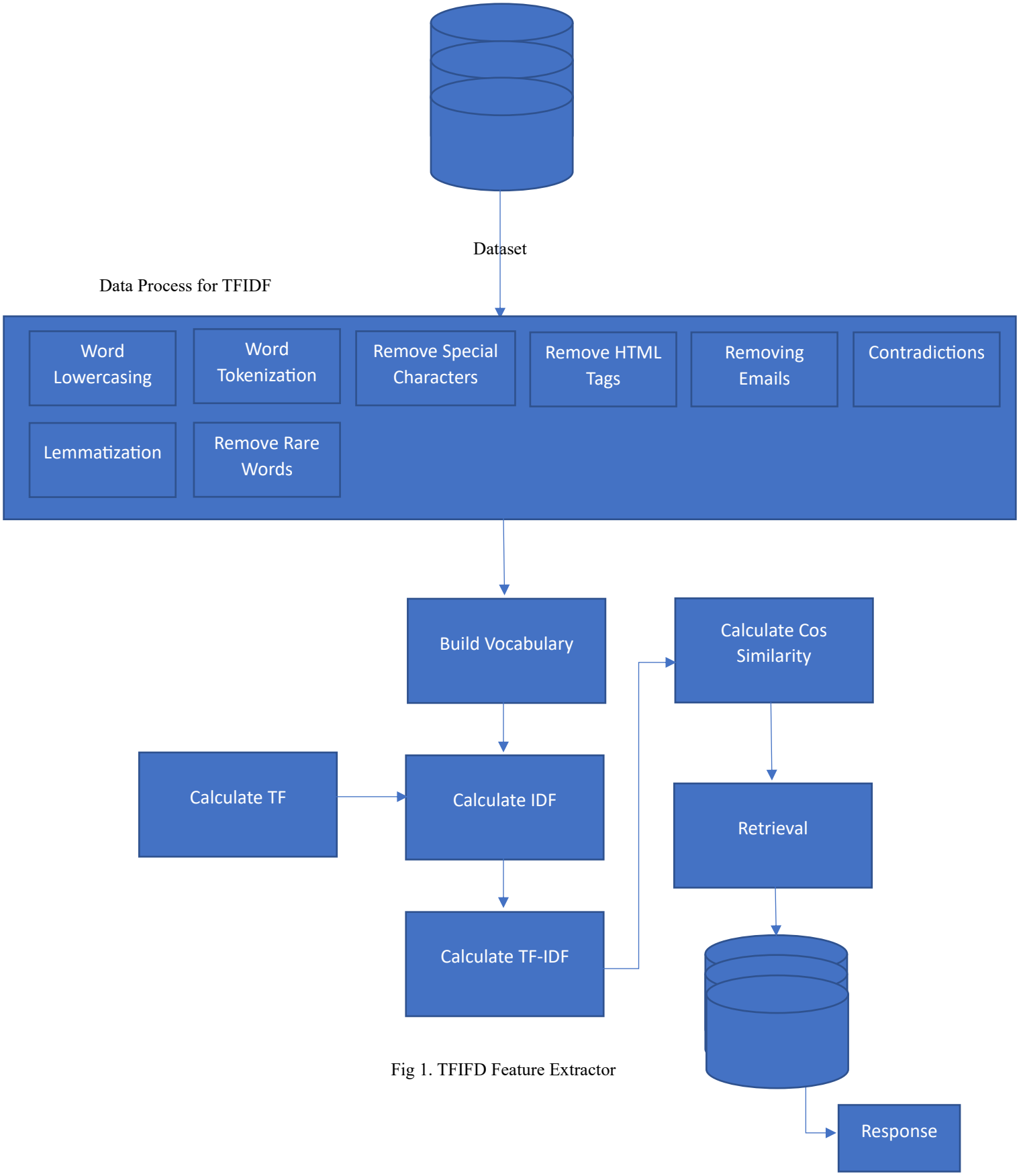

Fig 1. TFIFD Feature Extractor

4.2 Classification Algorithms

We experimented with several machine learning algorithms for sentiment classification, including:

- **Logistic Regression**
- **Support Vector Machines (SVM)**

4.2.1 Logistic Regression

Logistic regression is another commonly used classification algorithm in sentiment analysis. Despite its name, logistic regression is a binary classification algorithm that estimates the probability of an instance belonging to a particular class. Like SVM, the textual data is transformed into numerical feature vectors using techniques like TF-IDF or word embeddings. Logistic regression estimates the probability of an instance belonging to a specific class using the logistic function. It calculates the weighted sum of the feature values and applies the logistic function to obtain the probability. Logistic regression uses a decision boundary to classify instances based on the predicted probabilities. By selecting an appropriate threshold (e.g., 0.5), instances with probabilities above the threshold are assigned to one class, while those below the threshold are assigned to the other class. Logistic regression is trained by optimizing the parameters (weights) of the model to minimize the difference between the predicted probabilities and the actual class labels. This optimization is typically done using techniques like maximum likelihood estimation or gradient descent.

4.2.2 Support Vector Machine (SVM):

Support Vector Machines (SVM) is a popular supervised machine learning algorithm used for classification tasks, including sentiment analysis. Before applying SVM, the textual data is typically transformed into numerical feature vectors using techniques like TF-IDF or word embeddings. SVM aims to find a hyperplane that best separates the data points of different classes in the feature space. The hyperplane is defined by a set of weights and biases. SVM seeks to maximize the margin between the hyperplane and the nearest data points of each class. The margin represents the distance between the hyperplane and the closest data points. By maximizing the margin, SVM aims to achieve better generalization and robustness.

For kernel, SVM can be extended to handle nonlinear data by applying the kernel trick. The kernel function transforms the input space into a higher-dimensional space, allowing SVM to find nonlinear decision boundaries. SVM solves an optimization problem to find the optimal hyperplane that maximizes the margin. Various optimization algorithms, such as quadratic programming or gradient descent, can be used to find the optimal solution.

4.3 Model Training and Evaluation

We divided the dataset into training and testing sets, using a 50:50 split. We trained each model using the training set and evaluated its performance using various evaluation metrics, such as accuracy, precision, recall, and F1 score. We also employed k-fold cross-validation to assess the models' robustness. We also employed k-fold cross-validation to the model could performed accurately and efficiently on unseen data.

5. Implementation and Results

5.1 Model Selection

Based on our evaluation results, we selected the Logistic Regression Algorithm as our final model. It demonstrated the highest accuracy and performed well across all evaluation metrics. Logistic regression is a simpler algorithm compared to SVM, making it easier to interpret and implement. It performs well when there is a moderate number of features and enough labelled training data. While SVM can handle complex decision boundaries and potentially capture non-linear relationships through the kernel trick, logistic regression can still provide reliable results for sentiment analysis tasks. The logistic regression cross validation score is 87.06 and SVM is 84.05. That’s why we selected Logistic regression instead of SVM.

5.2 System Implementation

We implemented the sentiment analysis system using Python, scikit-learn library and the Natural Language Pre-processing (NLP). The system takes input text data and collecting a suitable dataset for sentiment analysis. This can be obtained from various sources, such as online review platforms, social media platforms, or custom data collection methods. Then ensure that the dataset is labelled with sentiment classes (positive, negative) to train and evaluate the sentiment analysis model.

In third step, NLP Clean and pre-process the collected data to prepare it for analysis. This typically involves steps like removing noise, handling special characters, tokenizing the text, removing stop words, and normalizing the text (e.g., lowercase conversion). Apply any additional data cleaning techniques specific to your dataset. Further, we extract relevant features from the pre-processed data. One common technique is TF-IDF (Term Frequency-Inverse Document Frequency), which assigns weights to words based on their importance in the documents. Train the selected model using the training dataset. Adjust the model's parameters to minimize the loss function and optimize the sentiment predictions. Apply techniques like cross-validation and grid search to fine-tune hyperparameters for better performance. We Choose the appropriate machine learning or deep learning model for sentiment analysis run different algorithms like Logistic regression, Support Vector Machines (SVM). We Selected the model based on factors such as accuracy, computational requirements, and interpretability. In second last step, evaluate the trained model's performance using the validation dataset. Metrics such as accuracy, precision, recall, F1-score, and confusion matrix can be used to assess the model's effectiveness in predicting sentiment. At the end, predictions on the new data and check the different sentiments.

5.3 Performance Results

Our final sentiment analysis system achieved an accuracy of 86% on the test set. The precision, recall, and F1 score for each sentiment category were as follows:

Table 1. Classification Report

precision	recall	f1-score	support	
negative	0.90	0.79	0.84	12474
positive	0.81	0.91	0.86	12526
accuracy			0.85	25000
macro avg	0.85	0.85	0.85	25000
weighted avg	0.85	0.85	0.85	25000

6. Conclusion and Future Work

Our sentiment analysis project successfully developed a system capable of accurately classifying sentiment in text data. The Logistic Regression model demonstrated strong performance, achieving an accuracy of 86%. The system can be deployed for sentiment analysis in various domains, such as social media monitoring, customer feedback analysis, and market research.