universität ulm

Amazon Reviews Sentiment Analysis using logistic Regression

Bachelor Thesis

Universität Ulm — 89069 Ulm — Germany

Faculty of Engineering, Computer science and Psychology

Institute of Neural Information Processing

Author:

Michael Yassa

michael.yassa@uni-ulm.de

1170779

Supervisor:

Prof. Dr. Friedhelm Schwenker

friedhelm.schwenker@uni-ulm.de

This is to certify that:

(i) The thesis comprises only my original work toward the Bachelor's Degree

(ii) due acknowledgment has been made in the text to all other material used

---

Michael Milad Wadiee Yassa

15th August, 2023

Acknowledgments

Firstly, I would like to express my sincere gratitude to God for granting me strength, inspiration, and guidance throughout this journey. Your blessings have been instrumental in my success.

I extend my heartfelt appreciation to Prof. Dr. Friedhelm Schwenker, my thesis advisor, for his invaluable support. his guidance and mentorship have shaped my research and enriched my academic experience.

I am incredibly grateful to my family for their unconditional love, constant encouragement, and unwavering belief in my abilities. Their unwavering support has been the foundation of my achievements, and I am forever indebted to them.

I would also like to thank my colleagues and friends for their collaboration, insightful discussions, and the stimulating academic environment we shared.

Finally, I want to acknowledge the collective efforts of all the individuals and institutions who have directly or indirectly contributed to the completion of this thesis. Your support, whether through resources, feedback, or encouragement, has been invaluable.

I am truly grateful for the immense support I have received, and I extend my deepest appreciation to everyone who has been a part of my academic journey.

ABSTRACT

This research project aims to develop a machine learning model for sentiment analysis of product reviews from a large Amazon dataset. Utilizing Logistic Regression, the study focuses on accurately predicting whether reviews are positive or negative. The research question driving this investigation is: Can machine learning models effectively classify sentiment in various product reviews? By training and evaluating the model on a substantial dataset, the study achieves exceptional accuracy in sentiment prediction, providing valuable insights for businesses seeking to understand customer sentiment. The findings underscore the potential of Logistic Regression in sentiment analysis, demonstrating their efficacy in processing vast amounts of textual data and aiding decision-making processes. This research contributes to the field by showcasing the capabilities of advanced machine learning techniques in extracting meaningful insights from user-generated content and driving improvements in customer experience.

To the memory of my father, Milad Wadie Yassa.

To the memory of my brother, Maged Milad Wadiee.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure Page

1. INTRODUCTION

1.1 Motivation

In today's world of online shopping, where platforms like Amazon and E-pay, product reviews have become crucial. Shoppers rely on reviews to decide if a product is good or not. Positive reviews highlight the good things, while negative ones point out problems. People care about reviews because they want to choose wisely. Positive reviews boost confidence, while negative ones prevent bad choices. With countless options online, reviews are like trustworthy guides.

This is why we're into sentiment analysis, a way to understand feelings in reviews. By studying people's words, we determine if they're happy, neutral, or unhappy about a product. This helps us learn what customers like and don't like, making shopping easier and smarter for everyone. So, let's dig into sentiment analysis and uncover the emotions behind reviews. Our goal is to improve online shopping by grasping what people think and feel while writing those reviews.

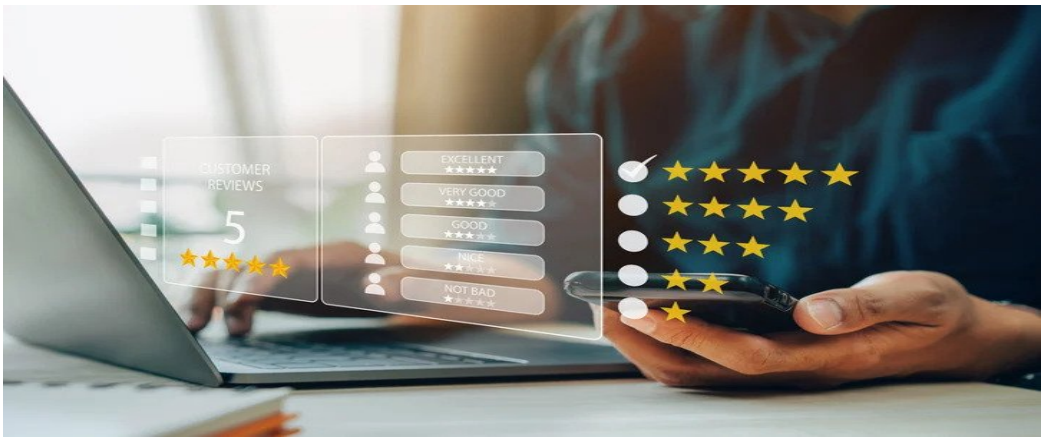

Fig. 1.1. Rating & Reviews Analysis: Unveiling Customer Sentiments. Source: Obtained from: `https://get.nicejob.com`

1.2 Problem Statement

This bachelor thesis project aims to develop an effective sentiment analysis model designed for product review analysis. Sentiment analysis, also known as opinion mining, is a sub-field of natural language processing (NLP) that focuses on identifying and categorizing the sentiment expressed in text. By automating the sentiment analysis process, businesses can gain valuable insights into customer opinions, improve their products and services, and enhance overall customer satisfaction.

The main problem addressed in this project is the lack of an accurate and reliable sentiment analysis model tailored specifically for product reviews. Existing sentiment analysis models often struggle to handle the nuances and complexities of product-related sentiments, leading to sub-optimal performance. Thus, this project aims to fill this gap by developing a model that can accurately predict sentiment in product reviews, enabling businesses to gain a deeper understanding of customer feedback and make informed decisions.

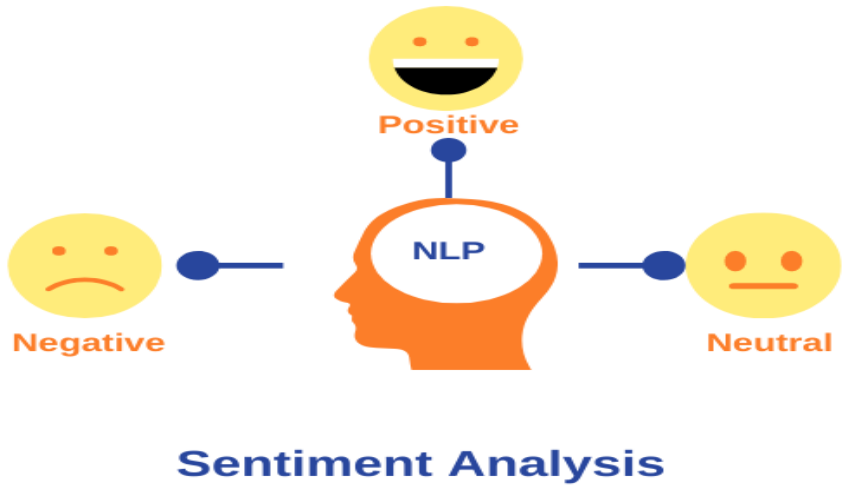

Fig. 1.2. Analysis in Natural Language Processing: Decoding Customer Opinions. Source: Obtained from: `https://medium.com`

1.3 Thesis Organization

1. **Chapter 1 : Introduction**
    - Motivate sentiment analysis's significance in public opinion.
    - Problem: Performing sentiment analysis on Amazon reviews via logistic regression.
    - Outline the organization of the thesis.
2. **Chapter 2 : Background and Literature Review**
    - Discuss machine learning, NLP, and sentiment analysis.
    - Explore feature extraction and logistic regression.
    - Review some previous related work.
3. **Chapter 3 : Methodology**
    - Describe sentiment analysis approach.
    - Explain dataset, preprocessing, detail cleaning, stop words, stemming, TF-IDF.
    - Explain logistic regression for sentiment analysis.
4. **Chapter 4 : Analysis Results**
    - Present sentiment analysis results on Amazon reviews.
    - Display performance metrics.
    - Evaluate the model on an alternative dataset and report the results.
5. **Chapter 5 : Conclusion and Future Work**
    - Summarize findings and contributions.
    - Restate objectives and achievements.
    - Conclude significance in sentiment analysis and Future work.

2. BACKGROUND AND LITERATURE REVIEW

2.1 Natural Language Processing

Natural Language Processing (NLP) is a field of artificial intelligence that focuses on the interaction between computers and human language. It involves the development of algorithms and models that enable computers to understand, interpret, and generate human language in a way that is both meaningful and useful. NLP encompasses a wide range of tasks, including language understanding, sentiment analysis, machine translation, text generation, speech recognition, and more. The goal of NLP is to bridge the gap between human communication and computer understanding, enabling applications that can process and generate natural language text or speech. [1]

2.1.1 Importance of Natural Language Processing

The importance of NLP lies in its wide-ranging applications across various domains. It facilitates sentiment analysis to gauge public opinion, powers virtual assistants like Siri and Alexa, improves language translation services, enables chatbots for customer support, and assists in information retrieval from vast amounts of text data. NLP also plays a significant role in social media analysis, healthcare, finance, and education, revolutionizing how we interact with technology and access information.

2.1.2 How Natural Language Processing works

Natural Language Processing (NLP) encompasses a wide range of tasks aimed at enabling computers to understand, analyze, and generate human language. As elaborated in [2] NLP involves techniques such as tokenization, part-of-speech tagging,

syntactic parsing, and semantic analysis. These techniques allow machines to extract information, uncover linguistic patterns, and derive meaningful representations from text, facilitating applications in machine translation, information retrieval, and sentiment analysis."

2.2 Machine Learning

Machine learning is a subset of artificial intelligence that empowers computers to learn from data and improve performance on specific tasks without explicit programming. Its algorithms enable the identification of patterns, making predictions, and discovering insights from vast datasets. There are several types of machine learning: supervised learning (using labeled data for training), unsupervised learning (extracting patterns from unlabeled data), and reinforcement learning (agents learn through trial and error based on rewards and penalties) [3]. These diverse approaches find applications in various fields, from natural language processing and image recognition to recommendation systems and autonomous vehicles. Machine learning continues to drive technological innovation, transforming industries and our daily lives.

2.2.1 Supervised learning

Supervised learning is a machine learning approach where the algorithm is trained on labeled data, consisting of input-output pairs. It aims to learn a mapping function that can accurately predict outputs for new, unseen inputs. As described by Bishop, this process involves the optimization of model parameters through the minimization of an appropriate loss function, enabling the algorithm to generalize from the training data to make predictions on new instances. [3]. The central problems it can solve include

1. classification, where the model assigns inputs to specific categories (e.g., sentiment analysis, spam detection),

2. regression, where it predicts continuous numerical values (e.g.,prices, temperature).

   The supervised learning process involves minimizing a predefined error function, such as mean squared error (MSE), using techniques like gradient descent to optimize the model's parameters and improve its predictive performance. However, obtaining sufficient labeled data and avoiding overfitting are common challenges in supervised learning.

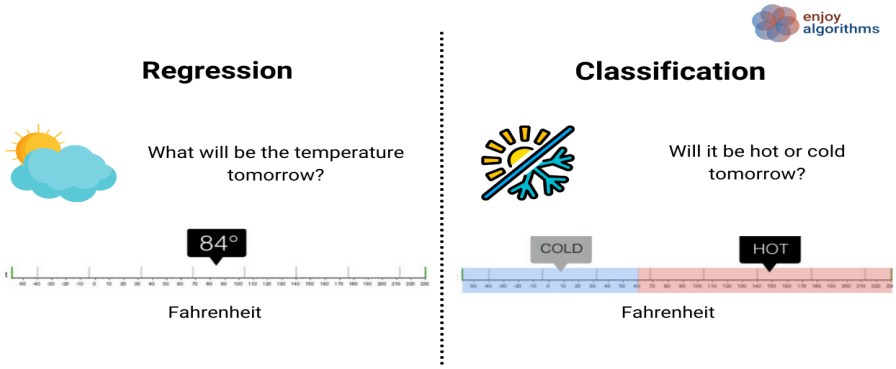

Fig. 2.1. Regression and Classification: Comparing Regression (Temperature Number) with Classification (Hot or Cold). Source: Obtained from: `https://anubrain.com/`

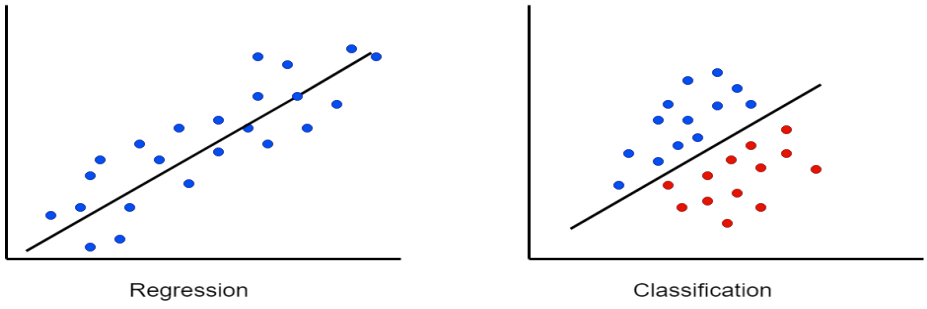

Fig. 2.2. Regression and Classification: Drawing Decision Boundaries to Predict Values and Classify Data. Source: Obtained from: `https://www.pycodemates.com/`

2.2.2 Unsupervised Learning

In [4], unsupervised learning is described as a fundamental machine learning paradigm aimed at extracting meaningful patterns and structures from unlabeled data. Unlike supervised learning, which relies on labeled examples, unsupervised learning focuses on uncovering inherent relationships and representations within the data itself. This approach encompasses techniques such as clustering and dimensionality reduction, facilitating the discovery of latent features that can aid in subsequent tasks like classification or visualization. The central problems it solves include:

1. Clustering: Grouping similar data points to reveal patterns and segments.
2. Dimensionality Reduction: Simplifying data by retaining vital information.
3. Anomaly Detection: Identifying outliers deviating from normal patterns.

Unsupervised learning finds applications in fields such as customer segmentation, anomaly detection, and data compression, enabling valuable insights and simplification in complex datasets without the need for labeled data.

2.2.3 Reinforcement learning

Reinforcement learning is a type of machine learning in which an agent learns how to behave in an environment by performing actions and receiving rewards or penalties in return. As explained in [5] the agent's goal is to maximize the cumulative reward over time by discovering optimal strategies and making informed decisions. The central problems it solves include:

1. Decision-Making in Uncertain Environments: Reinforcement learning addresses situations where the outcome of an action is uncertain, and the agent must learn the best actions through trial and error.
2. Reward Optimization: The agent learns to optimize its actions to maximize the cumulative rewards received from the environment.

3. Exploration-Exploitation Tradeoff: Balancing between exploring new actions to discover better strategies and exploiting known actions for immediate rewards.

Reinforcement learning is employed in various applications like game playing, robotics, and autonomous vehicles, where the agent learns to make intelligent decisions in dynamic and complex environments to achieve specific goals.

2.3 Sentiment Analysis

Sentiment analysis, also referred to as opinion mining, is a branch of natural language processing (NLP) that seeks to discern the emotional inclination or sentiment conveyed in textual content. As elucidated in [6], the objective of sentiment analysis is to categorize the sentiment as positive, negative, neutral, or in some cases, encompass more nuanced emotions such as happiness, sadness, anger, and others.

This process involves several steps, including text preprocessing, feature extraction, and classification. Various NLP tools and machine learning algorithms, such as Support Vector Machines (SVM), Logistic regression, and deep learning models like Recurrent Neural Networks (RNNs) and Transformers, are commonly used for sentiment analysis tasks.

Sentiment analysis finds applications in social media monitoring, customer feedback analysis, market research, and brand reputation management. While it has shown great potential, challenges remain in handling sarcasm, context-dependent sentiment, and language nuances, especially with short and informal texts. Continuous advancements in NLP and machine learning techniques continue to enhance the accuracy and reliability of sentiment analysis systems.

2.3.1 Importance of Sentiment Analysis

1. Customer Insights: Sentiment analysis provides valuable insights into customer sentiments, preferences, and feedback for data-driven improvements [7].

2. Brand Reputation: Monitoring sentiment helps companies track brand reputation and address negative sentiments to safeguard brand image [8].

3. Market Research: Sentiment analysis aids in understanding market trends, competitor analysis, and identifying opportunities or threats [9].

4. Customer Service: Analyzing sentiments in customer interactions leads to personalized and empathetic responses, enhancing satisfaction [10].

5. Product Launches: Sentiment analysis gauges public interest, identifies issues, and refines strategies for successful product launches [11].

6. Stock Market Predictions: In finance, sentiment analysis predicts stock trends based on public sentiments about companies or industries [12].

7. Social Media Marketing: It measures campaign success, identifies influencers, and optimizes social media marketing strategies [13].

2.4 Feature Extraction

Feature extraction is a crucial step in many data analysis, machine learning, and computer vision tasks. It refers to the process of transforming raw data (e.g., images, text, audio) into a set of relevant and informative features that can be used as input for further analysis or modeling. The goal of feature extraction is to represent the data in a more compact and meaningful way, capturing important patterns or characteristics that are relevant to the specific task at hand [14].

In the context of Natural Language Processing (NLP), feature extraction is a critical process aimed at converting raw text data into numerical representations that can be effectively processed by machine learning algorithms. The challenge in NLP lies in converting unstructured text data into a structured format that can be utilized for various language-based tasks.

two common methods for feature extraction are Bag-of-Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF).

1. Bag-of-Words (BoW): The Bag of Words model is a simple and effective approach for representing text data, where each document is represented by a vector that counts the occurrences of words in a fixed vocabulary [14]. BoW ignores word order and syntax, focusing only on the presence and frequency of words. While BoW is simple and interpretable, it may not capture the sequential information and context of words.

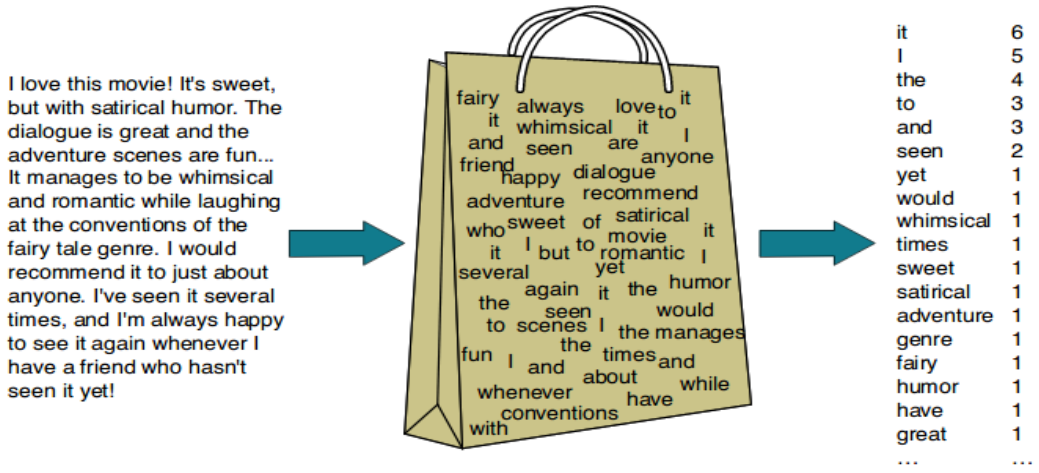

Fig. 2.3. Illustration of the Bag of Words Technique for Text Analysis. Source: Obtained from: `https://medium.com/`

2. TF-IDF (term frequency-inverse document frequency): is a numerical representation of a term's importance in a document within a collection [15], TF-IDF is an enhancement to BoW, which takes into account both the term frequency (TF) and the inverse document frequency (IDF). TF-IDF assigns higher weights to words that are frequent in the document but rare across the entire corpus, emphasizing more informative words.

Word	TF		IDF	TF*IDF	
	A	B		A	B
The	1/7	1/7	log(2/2) = 0	0	0
Car	1/7	0	log(2/1) = 0.3	0.043	0
Truck	0	1/7	log(2/1) = 0.3	0	0.043
Is	1/7	1/7	log(2/2) = 0	0	0
Driven	1/7	1/7	log(2/2) = 0	0	0
On	1/7	1/7	log(2/2) = 0	0	0
The	1/7	1/7	log(2/2) = 0	0	0
Road	1/7	0	log(2/1) = 0.3	0.043	0
Highway	0	1/7	log(2/1) = 0.3	0	0.043

Fig. 2.4. Visualizing TF-IDF: Term Frequency-Inverse Document Frequency Representation. Source: Obtained from: `https://www.freecodecamp.org/`

2.5 Logistic Regression

Logistic regression is a statistical method used for modeling the probability of a binary outcome by fitting a logistic curve to a set of independent variables. It is commonly used in situations where the dependent variable is categorical and represents one of two possible outcomes (such as yes/no, success/failure, or true/false). [16] Logistic Regression is often compared to Linear Regression, another popular statistical method used for predicting continuous numerical values. While both models use linear relationships between input features and coefficients, they differ in their output and purpose. Linear Regression predicts a continuous output, making it suitable for regression tasks, while Logistic Regression is designed for binary classification, producing probabilities that represent the likelihood of an instance belonging to a specific class. Additionally, Logistic Regression employs a sigmoid function to constrain its output between 0 and 1, allowing for probabilistic interpretations, while Linear Regression outputs unbounded continuous values.

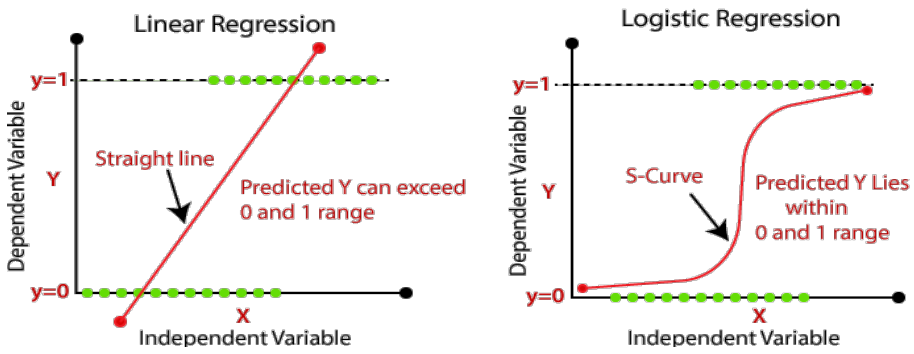

Fig. 2.5. Two Sides of the Regression Coin: A Comparative Analysis of Linear and Logistic Models. Source: Obtained from: `https://www.javatpoint.com/`

2.5.1 Model Overview and Sigmoid Function

The basic idea behind logistic regression is to model the probability that an instance belongs to a particular class as a function of its features. The logistic regression model applies the logistic function (also known as the sigmoid function) to the linear combination of the input features and their associated weights, and then maps the result to a probability value between 0 and 1. [17]

The sigmoid function, denoted as $\sigma(z)$, is a key component of logistic regression. It has an S-shaped curve that maps any input value to an output value between 0 and 1. The sigmoid function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where e is the base of the natural logarithm and z is the linear combination of input features and their corresponding weights, i.e., $z = w_1x_1 + w_2x_2 + \ldots + w_nx_n$.

The sigmoid function is used to squash the output of the linear combination into a range that can be interpreted as a probability. Values close to 0 or 1 are interpreted as

high probabilities of belonging to one of the classes, while values close to 0.5 indicate uncertainty or a balanced probability.

By applying the sigmoid function to the linear combination of features and weights, logistic regression effectively transforms the linear output into a probability score, making it suitable for binary classification tasks. [18]

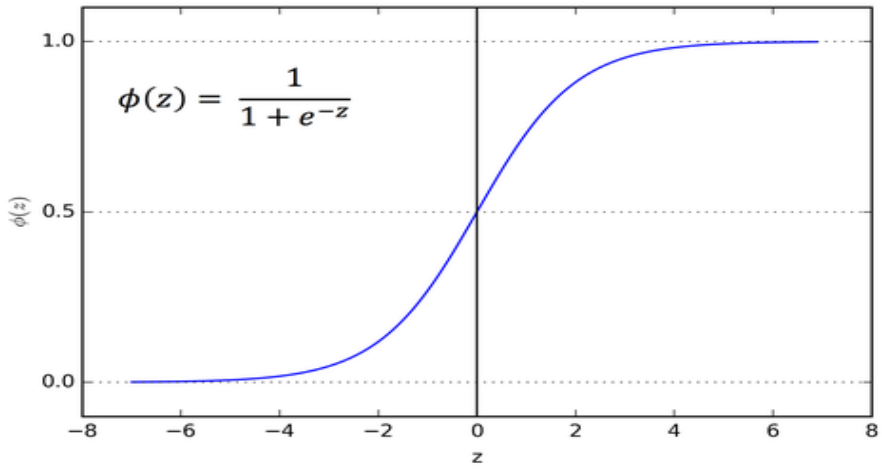

Fig. 2.6. Shaping Predictions: The Role of the Sigmoid Function in Logistic Regression [19].

2.6 Related Work

2.6.1 Sentiment Analysis

Sentiment analysis is a multifaceted field encompassing various aspects, including methodologies, techniques, applications, and assessment metrics. In his study [20], Peter Turney employed an unsupervised approach to classify reviews by calculating their semantic orientation. His method focused on phrases containing adjectives or adverbs that convey positive or negative sentiment based on context. Through this approach, Turney achieved a 75% accuracy in categorizing reviews based on their semantic orientation. Another approach to sentiment classification involves analyzing

each sentence independently to determine its sentiment. It is crucial to differentiate between subjective and objective data in sentiment analysis. Jiani Zhang et al. introduced an innovative technique for sentiment analysis using a hierarchical neural network to categorize sentiment at the aspect level. The network identifies aspect categories and sentiment polarity, employing recurrent neural networks for text data processing. Notably, Vandana Jagtap et al.'s research [21] assumed a single sentiment per sentence and achieved an impressive accuracy of 82.9%. Bo Pang et al. [22] adopted a multiclass approach for text categorization, assessing the overall sentiment of entire text documents rather than individual sentences. This approach considers star ratings (ranging from 1 to 5) from various e-commerce websites [23]. Initially classifying reviews based on ratings, the multi-class method categorizes each review into multiple classes, contributing to the classification of the entire dataset.

2.6.2 Evaluation of Sentiment Analysis

Many research studies have examined various methods for Sentiment analysis, which involve using computer programs to understand if the text expresses positive, negative, or neutral emotions. These methods include using different types of computer algorithms, looking at lists of words associated with emotions, and advanced techniques like deep learning. The goal is to make these methods better at accurately identifying emotions in text. In a study mentioned in [24], researchers examined how to understand emotions in Amazon reviews. They used special computer programs like Multinomial Naive Bayesian (MNB) and support vector machine (SVM) to help them. They also used some techniques to get the text ready for analysis, like TF-IDF and Part of Speech tagging. In their work, they used a different way to count words than we did. Instead of using Bag-of-Words, they used something called N-grams. They found that the SVM method was able to guess feelings with about 82.27% accuracy when they used the TF-IDF technique on the Amazon product data. In the context of the study [25], a comparison was conducted between the outcomes derived

from decision trees and naive Bayes algorithms for sentiment analysis, employing Amazon review datasets. The training process for the classifiers took place using the Kindle dataset. Our approaches encompass a wider spectrum of techniques and models when contrasted with those presented earlier, and the highest accuracy they achieved is 88%. In another research [26], scientists looked into how choosing certain features and using them together affects sentiment analysis of dialectal Arabic. They studied how different methods of giving importance to words, making words simpler, removing common words, and using different ways to represent features affected the model's performance, similar to our suggested approach. What they found was that the SVM classifier had the best results in terms of accuracy, achieving a high accuracy rate of 90.7%. Similarly, in a different research paper [27], the authors attempted to predict sentiments by considering ratings in written reviews. They identified words that had positive or negative effects specifically related to the "Health & Personal Care" category. Unlike our approach, they evaluated their results using a metric called Root Mean Squared Error (RMSE). For instance, when using the SVM model, they achieved an RMSE of 1.02.

3. METHODOLOGY

The methodology involves data collection from a suitable dataset, followed by text preprocessing steps such as lowercasing, removal of non-word characters/punctuation, removal of accented characters, and word stemming. The preprocessed text will then be represented using TF-IDF vectorization to convert it into numerical feature vectors. The logistic regression model will be implemented and trained on the TF-IDF vectors to perform sentiment analysis. The model's performance will be evaluated using appropriate metrics, and the results will be discussed, providing insights into the most influential features contributing to sentiment classification. Ethical considerations related to the dataset and analysis will also be addressed.

3.1 Dataset

In this research, we utilized a dataset comprising 4 million Amazon reviews, which was obtained from the Kaggle website. The dataset is divided into two separate files: a training file containing 3,600,000 comments and a testing file with 400,000 comments. Notably, both files exhibit a balanced distribution of positive and negative reviews, where

´__label__<X>´ is assigned the value of 1, representing negative reviews,

´__label__<Y>´ is assigned the value of 2, indicating positive reviews.

Each review in the dataset follows a specific format, with two labels denoted as ´__label__<X>´ and ´__label__<Y>´, preceding the actual review text. Prior to utilization, no additional data pre-processing was conducted, and the raw labels were used for training our machine learning models. The dataset's dual-label format, clarifying positive and negative sentiments, played a pivotal role in the success of our research. Proper ethical considerations were ensured during the study, safeguarding

data privacy and confidentiality. The full citation for the Kaggle dataset has been provided in the references section to duly acknowledge the original creators.

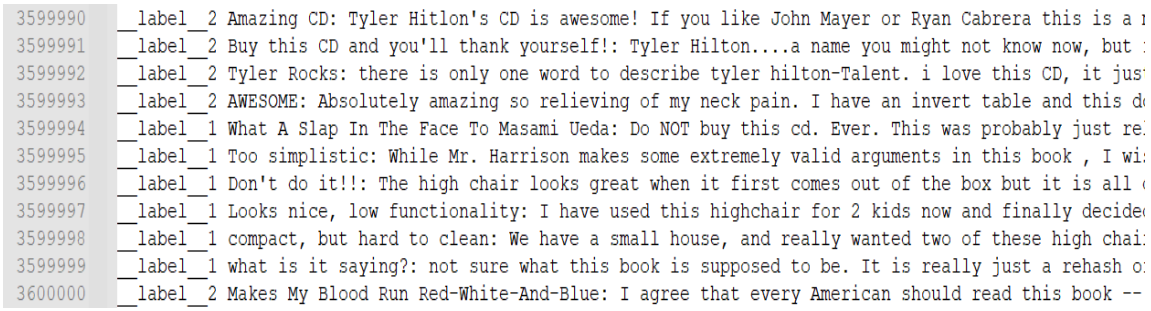

Fig. 3.1. Dataset Samples: A snapshot of reviews showcasing different sentiment expressions. Source: Obtained from: `https://www.kaggle.com/`

3.2 Text Preprocessing

3.2.1 Reading the data

the initial step of the text preprocessing process involved segregating the textual data into two distinct arrays: ”labels” and ”comments.” This segregation was accomplished using a custom function named ”assign_labels_and_comments.” Within the function, the text data was read from a specified file, which appeared to contain sentiment labels (representing positive or negative sentiment) and the corresponding review text. The ”labels” array was utilized to store the sentiment labels, where the value 0 indicated a positive sentiment and the value 1 indicated a negative sentiment. On the other hand, the ”comments” array was employed to store the associated review text. The function extracted the sentiment labels and the review text from each line of the file, following a specific format, and compiled them into the respective arrays.

3.2.2 Cleaning the data

Data cleaning plays a vital role in the data preparation process, ensuring the dataset's accuracy, completeness, and reliability. By eliminating errors, inconsistencies, and irrelevant information, data cleaning enhances the quality of the data, leading to more robust and accurate analyses, better decision-making, and reliable insights for various applications.

1. The first step involves converting all text to lowercase. This process is essential to eliminate any discrepancies in letter case and ensure uniformity throughout the dataset. **For Example**, consider two reviews: "The product is AMAZING!" and "I didn't like the product." After applying lowercase conversion, both reviews would read: "the product is amazing!" and "i didn't like the product."

2. Next, we perform the removal of non-word characters and punctuation. This step is aimed at eliminating special characters, symbols, and punctuation, which are not significant for our analysis and may introduce noise in the data. **For Example**, the review "The product is great! Highly recommended!!!" would be transformed to "The product is great Highly recommended".

3. The third and final step involves removing accented characters. Accents, though valuable for expressing nuances in language, may create inconsistencies when processing textual data. By removing accents, we ensure that words with or without accents are treated as the same entity. **For Example**, the review "Café au lait was fantastique!" would be converted to "Cafe au lait was fantastique!" after the removal of the accent from the 'e' in 'Café'.

By executing these three data-cleaning steps on the product review dataset, we create a cleaner, more manageable corpus that retains the essential textual information while discarding extraneous elements. This cleaned data will serve as a solid foundation for further analysis and modeling in our research.

3.2.3 Removal of stop words

Removal of stop words is a critical step in text preprocessing for natural language processing tasks. Stop words are common words like "the," "and," and "is" that add little value to the meaning of the text. By eliminating these words, the text becomes more focused and meaningful, which improves the efficiency and effectiveness of text analysis. The process involves creating a set of stop words and then comparing each word in the text to this set. Any matches are removed

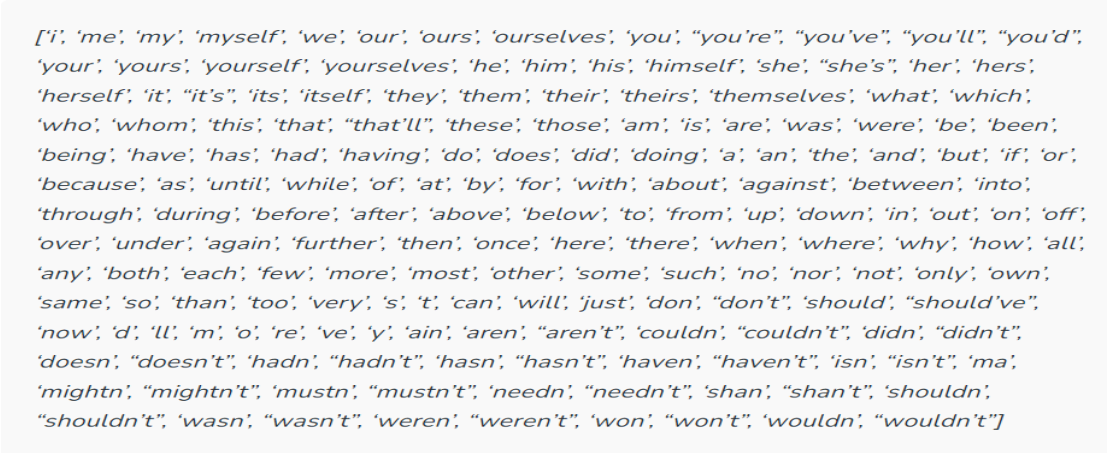

Fig. 3.2. Optimizing Text Analysis: Excluding Stop Words from the Corpus - The Array of Excluded Words.

I employed the Python library "nltk.corpus" to download a collection of stopwords in the English language. These stopwords are common words with limited semantic meaning, as represented in the provided image. Subsequently, I developed a method that splits the comments into individual tokens or words, facilitating further processing. Within the method, I created a string variable called `filtered_text` to store the words from the comments that are not found in the stopword array. If a word is part of the array, it is removed from the `filtered_text`. Consequently, I applied the stopword removal method to all comments in the dataset. Additionally, it's worth noting that the number of stopwords available in the NLTK library for the English

language is 179.

Below are the first two comments in their original form, followed by the processed versions after eliminating all identified stop words.

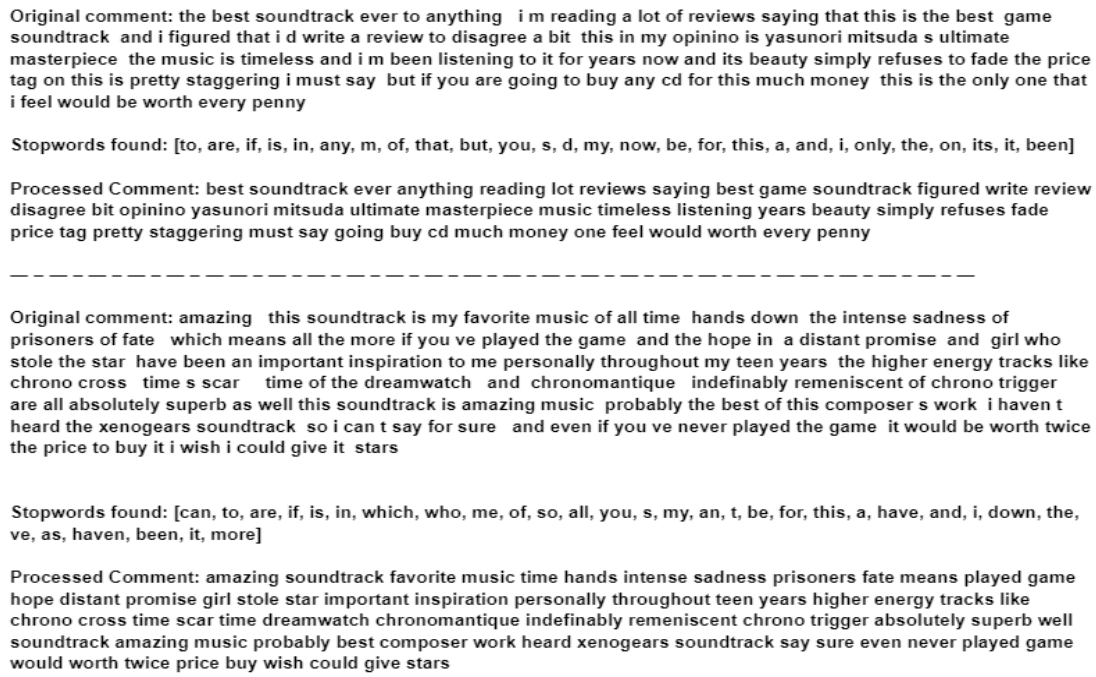

Fig. 3.3. Stop Words Removal: Visualizing comments before and after eliminating common words for enhanced text analysis.

In the examples provided in the image, we can see that the first comment had 508 characters before using the method to remove stop words. After applying this method, the comment became shorter and contained only 309 characters. Similarly, the second comment originally had 759 characters, but after applying the method, it was reduced to 489 characters.

So, when we attempt to calculate the average number of characters per comment using the formula:

$$\text{Average number of characters} = \frac{\text{Total number of characters}}{\text{Total number of comments}}$$

we find that the average number of characters per comment before using the method is 430.7302291, and after applying the method, we observe that the average number of characters per comment becomes 267.71549583.

3.2.4 Word Stemming

Word stemming is a natural language processing technique that reduces words to their base or root form. It helps in normalizing variations of words, improving text analysis and information retrieval. Stemming algorithms apply rules to remove suffixes and return the simplified form of a word Let us discuss the methodology employed in word stemming.

In this study, the word stemming process utilizes the "PorterStemmer" algorithm from the Natural Language Toolkit (NLTK) library in Python. The algorithm effectively reduces words to their base or root forms through a series of systematic steps. First, the text is tokenized into individual words, typically by splitting the text based on whitespace or punctuation marks.

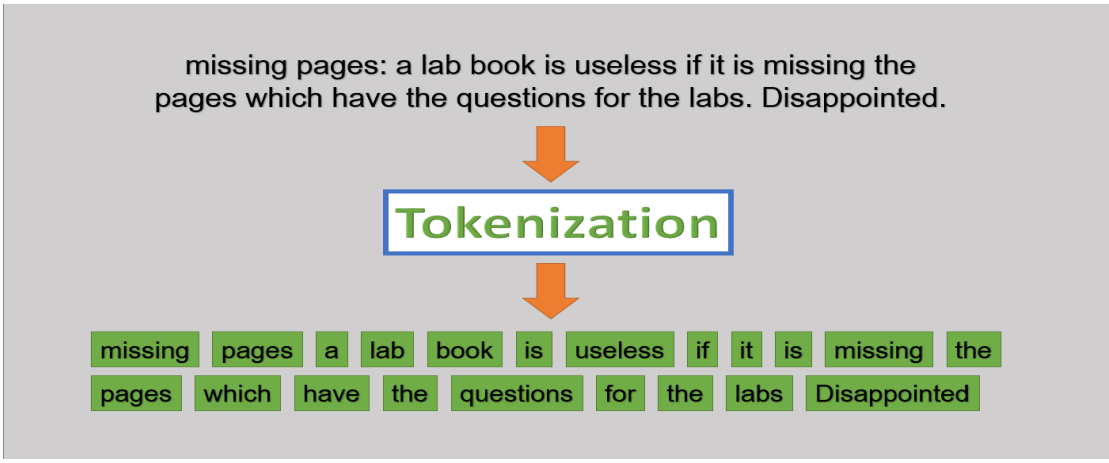

Fig. 3.4. Word Stemming First Step: Tokenization breaks text into individual words for analysis.

Next, stemming is performed on each word using the PorterStemmer, which applies a set of predefined rules to identify and remove common suffixes, For example, the stem() method in the PorterStemmer class may involve internally calling methods like step1a(), step1b(), step1c(), step2(), step3(), and step4() to perform specific transformations on the word. Here's an illustrative example of the rules in action:

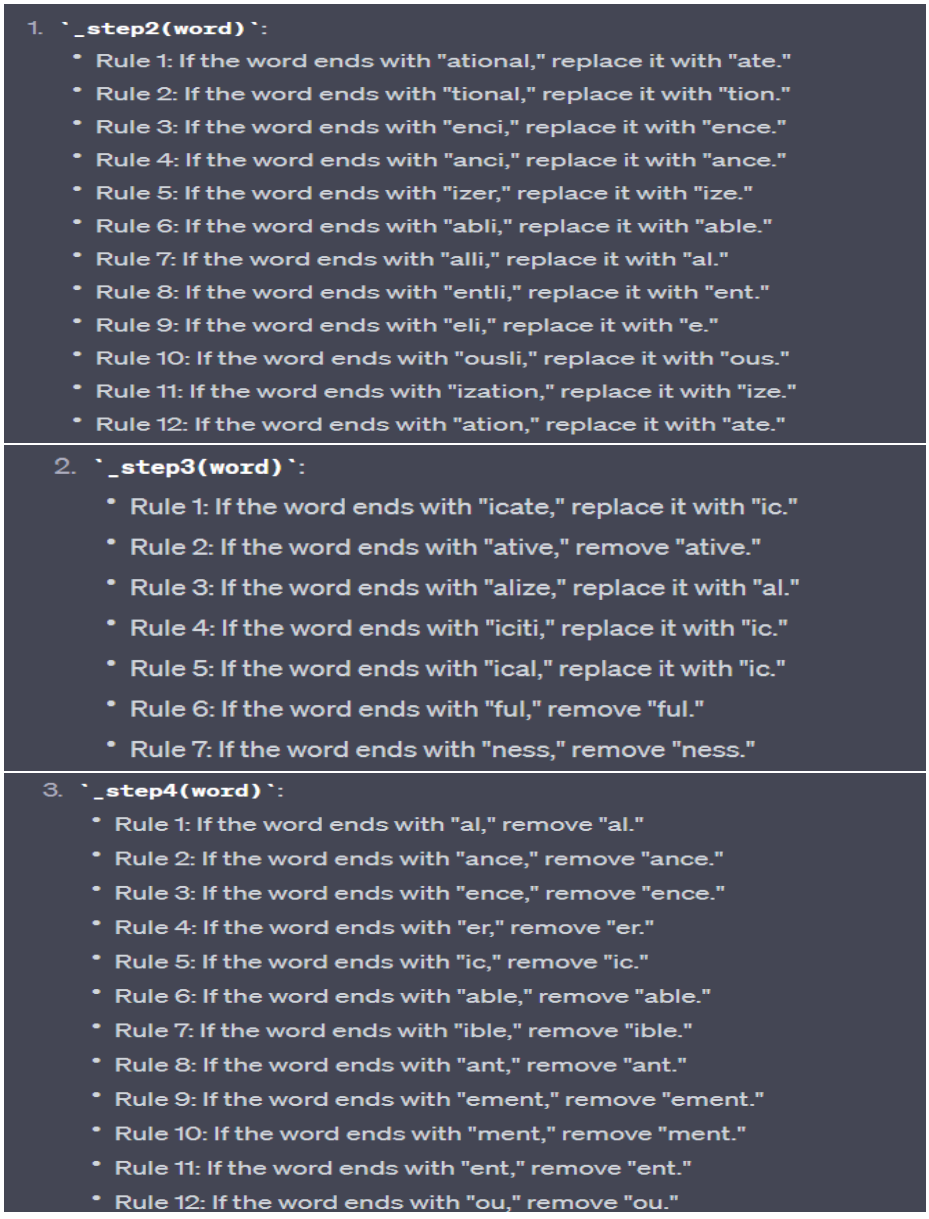

Fig. 3.5. Word Stemming Rules: Removing suffixes and replacing.

It's noteworthy that the Porter stemming algorithm comprises a total of 120 rules that encompass various transformations. In practice, words within reviews are checked against these rules, and if a word matches any rule, the corresponding transformation is applied.

Presented below are examples of word-stemming rules from each method:

- **Replacing "national" with "ate"**

  Rule example from Step 2 - _step2(word):

  Word: "educational"

  Stemmed: "educate"

- **Removal of the 'ful' suffix from words.**

  Rule example from Step 3 - _step3(word):

  Word: "beautiful"

  Stemmed: "beauti"

- **Removal of the 'ment' suffix from words.**

  Rule example from Step 4 - _step4(word):

  Word: "development"

  Stemmed: "develop"

3.3 Feature Extraction : TF-IDF Vectorizer

TF-IDF (Term Frequency-Inverse Document Frequency) is a widely used technique in text analysis tasks, particularly in natural language processing and information retrieval. It is a statistical measure that plays a crucial role in converting unstructured text data into a numerical representation, which is essential for applying machine learning algorithms to text-based datasets.

In text analysis, one of the challenges is how to represent text data in a way that machine learning algorithms can process effectively. Text data is inherently unstructured, composed of words and sentences, making it unsuitable for direct numerical analysis. TF-IDF addresses this problem by quantifying the importance of words in a document corpus and transforming text data into a numeric format [28].

The TF-IDF score of a word in a document is calculated by combining two components:

1. TF (Term Frequency) measures the frequency of a term (word) in a document relative to the total number of words in that document. It quantifies how often a specific word appears in a document and is computed as follows:

$$\text{TF}(t, d) = \frac{\text{Number of occurrences of term } t \text{ in document } d}{\text{Total number of terms in document } d}$$

   In this study, Term Frequency (TF) is calculated by counting how many times a specific "word" or "token" appears in an individual "comment" or "Review," then dividing it by the total count of "words" or "tokens" in that same "comment." This gives a TF value between 0 and 1, showing how often the word is used in the comment. A high TF value (close to 1) means the word is used frequently, possibly showing its importance in the main topic. Conversely, a low TF value (close to 0) means the word is used rarely, possibly having little effect on the overall meaning or context of the text.

2. IDF (Inverse Document Frequency) measures the rarity of a term in a document corpus. It quantifies the importance of a word by dividing the total number of documents in the corpus by the number of documents containing the term. The IDF value represents the logarithm of this ratio, and it can be used to weigh the significance of words in distinguishing documents.

$$\text{IDF}(t, D) = \log\left(\frac{N}{\text{df}(t, D)}\right)$$

Where: - IDF(t, D) represents the IDF value of term t in the document corpus D.
- N is the total number of documents in the corpus.
- df(t, D) represents the document frequency of term t, i.e., the number of documents in the corpus containing the term t.

Within this study's context, Inverse Document Frequency (IDF) computes based on the entire "comments" or "reviews" collection and specific "words" or "tokens" occurrence across all documents. IDF is calculated as shown earlier, considering the total dataset "comments" or "reviews" count, and identifying how many contain a given "word" or "token." Each word's IDF value derives from the logarithmic ratio of total documents (N) to those (n) with the specific word. A higher IDF indicates word rarity, often signaling distinctiveness and significance, while a lower IDF suggests commonality, often denoting less informative terms like common "stop words."

After calculating both the Term Frequency (TF) and Inverse Document Frequency (IDF) for each word in the dataset, you can proceed to compute the TF-IDF (Term Frequency-Inverse Document Frequency) scores for the words. TF-IDF is obtained by multiplying the TF value of a word within a specific document by its IDF value across the entire dataset:

$$\text{TF-IDF(word, document)} = \text{TF(word, document)} \times \text{IDF(word)}$$

After applying the TF-IDF (Term Frequency-Inverse Document Frequency) transformation to the collection of text reviews, each review is represented as a vector of numerical values, where each element in the vector corresponds to a specific word in the vocabulary of the dataset. The TF-IDF vector for each review reflects the importance of words within that review relative to the entire collection.

```
(0, 2)        0.6088450986844796
(0, 18)       0.35959372325985667
(0, 12)       0.6088450986844796
(0, 16)       0.35959372325985667
(1, 17)       0.6582448261869973
(1, 0)        0.2163782107827746
(1, 5)        0.2163782107827746
(1, 15)       0.2163782107827746
(1, 4)        0.2163782107827746
(1, 3)        0.32912241309349866
(1, 14)       0.2163782107827746
(1, 13)       0.2163782107827746
(1, 9)        0.2163782107827746
(1, 18)       0.25559291391456657
(1, 16)       0.25559291391456657
(2, 1)        0.19398964434436727
(2, 7)        0.19398964434436727
(2, 10)       0.19398964434436727
(2, 19)       0.19398964434436727
(2, 8)        0.19398964434436727
(2, 11)       0.19398964434436727
(2, 6)        0.19398964434436727
(2, 17)       0.7376706234744687
(2, 3)        0.14753412469489371
(2, 18)       0.2291468179159858
(2, 16)       0.3437202268739787
```

Fig. 3.6. Transformed Reviews: Numerical Vectors of Reviews After TF-IDF Transformation.

As we see in **Fig. 3.5** the reviews are in the format of

$$(row\_index, column\_index) \quad tf\text{-}idf\_score$$

- row index: indicates the index of the review,
- column index: corresponds to the position of a word in the vocabulary, based on the order in which the unique words appear in the vocabulary during the TF-IDF transformation process

- TF-IDF score: associated with each element reflects the importance of the word within the specific review, considering both its local frequency in the review (TF) and its rarity across the entire collection (IDF).

In this context, a high TF-IDF score for a word in a specific document indicates that the word is both frequent in that document (TF) and rare across the entire collection of documents (IDF). This high score implies that the word is considered important and carries more weight in representing the content and theme of that particular document.

On the other hand, a small TF-IDF score for a word indicates that the word is either infrequent in the document (low TF) or appears commonly across the entire corpus (high IDF). Such words are considered less informative and might not contribute significantly to the overall meaning or context of the document.

3.4 The Classifier Architecture: Logistic Regression Model

Logistic Regression serves as a cornerstone in sentiment analysis, a crucial area of natural language processing. It excels in binary classification tasks by predicting sentiment labels (e.g., positive/negative) based on textual data. At its core, the model combines feature weights and bias to compute a linear function, subsequently transformed by the sigmoid function. This transformation yields probabilities that quantify class membership, enabling insightful sentiment predictions. The optimization process, often employing gradient descent, refines the model's parameters by minimizing the binary cross-entropy loss function, which captures the disparity between predicted probabilities and actual sentiment labels.

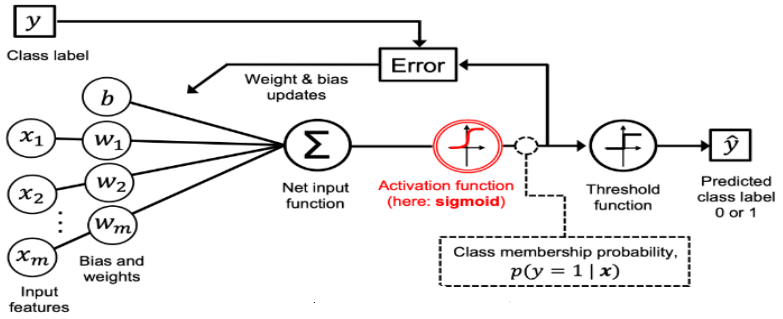

Fig. 3.7. Navigating the Training Odyssey: Weights, Equations, Activation, and Iteration in Sentiment Analysis. [19]

We've completed the entire data preparation process, and the ultimate outcome is that we now possess a numerical representation of our text data. This transformation has been achieved through a technique known as Term Frequency-Inverse Document Frequency TF-IDF, which has turned our words into numbers. This numeric format is crucial because it allows us to feed our data into the logistic regression model and analyze it effectively.

In order to explain the procedural sequence adopted by the model, let's take a closer look at the next set of actions. We'll describe these steps clearly and concisely, and we'll provide a practical example using numbers to help illustrate how everything work

1. Input Feature Initialization: The initial phase of our logistic regression model's architecture involves the utilization of TF-IDF vectors. These vectors, representative of individual comments, serve as the essential input features for our model. At the commencement of the model's operation, a crucial step ensues: the random initialization of weights associated with each distinct feature. Simultaneously, the bias term is also randomly initialized. This meticulous process establishes the foundation upon which subsequent training and optimization endeavors are built.

Fig. 3.8. Setting the Stage: Initializing Feature Weights and Bias.

2. Linear Combination: Having initialized the feature weights (w) and bias (b), the logistic regression model proceeds to compute a linear combination of these weights and the corresponding TF-IDF feature values for each comment. This aggregation captures the collective influence of the various features on the sentiment prediction. The linear combination is calculated as follows:

$$\text{Linear Combination} = \sum_{i=1}^{n} w_i \cdot \text{TF-IDF}_i + b$$

where n represents the total number of features, w_i is the weight associated with the i-th feature, and TF-IDF_i is the TF-IDF value of the i-th feature.

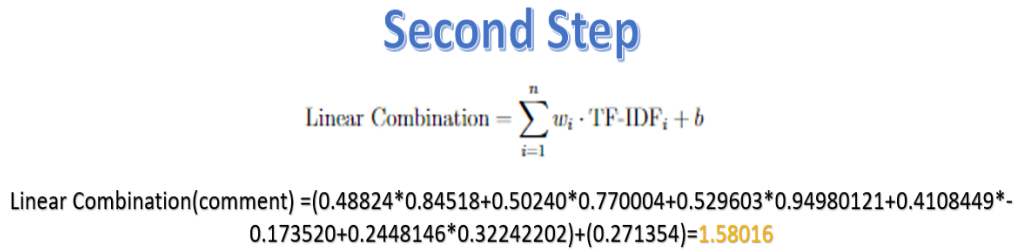

Fig. 3.9. Weighted Ensemble: Aggregating Features for Prediction.

3. Probability Thresholding and Sentiment Classification: Subsequently, the model employs the sigmoid function, a pivotal element of its architecture. The sigmoid function maps the computed linear combination onto a sigmoid curve, transforming it into a probability score bounded between 0 and 1. The sigmoid function is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

where x represents the computed linear combination. This transformational process converts the numeric output into probabilities that are easy to comprehend. This allows the model to assess the possibility of a given sentiment label (e.g., positive or negative) quantitatively based on the input features. The sigmoid transformation's smooth nature enables detailed sentiment predictions, facilitating a seamless shift across various sentiment levels. This step combines the intricate weight aggregation process with a probabilistic interpretation, creating a coherent mechanism for sentiment analysis.

After undergoing sigmoid transformation, the logistic regression model employs thresholding to classify sentiment labels. Utilizing a predefined threshold, commonly set at 0.5, the model distinguishes between positive and negative sentiment predictions. Probabilities exceeding the threshold lead to the assignment of the positive sentiment label, while those below it indicate negative sentiment.

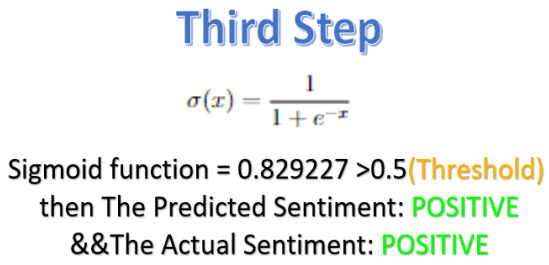

Fig. 3.10. Predicting Sentiment: Mapping to Probabilities and Classifying.

4. "Empowering Model Training: Loss Function Refinement. With sentiment labels now assigned through probability thresholding, a crucial aspect of model development comes to the forefront—the enhancement of the loss function. This step involves fine-tuning the binary cross-entropy loss function, which serves as a guiding metric for the model's predictive performance.

Binary Cross-Entropy Loss Function

The binary cross-entropy loss function is a commonly used loss function in machine learning, particularly for binary classification problems. It measures the difference between the predicted probabilities of the model and the actual binary labels of the data.

For a single data point, let's denote the true label (ground truth) as y (either 0 or 1) and the predicted probability of the positive class (class 1) as p. The binary cross-entropy loss is defined as:

$$L(y, p) = -[y \cdot \log(p) + (1 - y) \cdot \log(1 - p)]$$

Where:

- y is the true binary label (0 or 1).
- p is the predicted probability of the positive class (class 1).

The binary cross-entropy loss has several important properties:

(a) When the true label y is 1, the loss is primarily influenced by the negative logarithm of the predicted probability of the positive class. As the predicted probability approaches 1, the loss approaches 0.

(b) When the true label y is 0, the loss is primarily influenced by the negative logarithm of the complement of the predicted probability of the positive class (i.e., the predicted probability of the negative class). As the predicted probability approaches 0, the loss approaches 0.

(c) The loss is non-negative and increases as the predicted probability deviates from the true label.

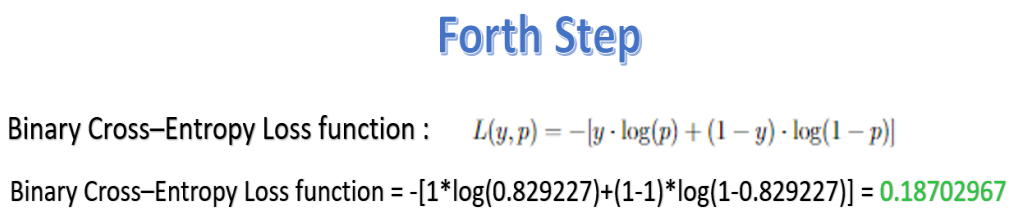

Fig. 3.11. Guiding the Model: Enhancing Predictive Performance through Loss Function Refinement.

5. Empowering Model Training: Optimization Mastery. Once the loss function is refined, the focus shifts to optimizing the model's performance. This phase entails the utilization of the average gradient descent optimization algorithm. Through iterative adjustments to the model's parameters, such as weights and bias, the optimization process enhances the model's predictive accuracy and convergence toward the desired outcomes.

Gradient Descent Optimization Function Gradient Descent is a fundamental optimization method that helps guide the model toward its best possible setup. It operates like a conductor leading a symphony, coordinating a series of adjustments to the model's parameters. Instead of looking at just one example at a time, it examines groups of training data. This clever approach smooths out any erratic changes that might occur with single examples and ensures a steadier path towards finding the best solution.

$$\text{New Parameter} = \text{Old Parameter} - \alpha \times \nabla \text{Loss}(\text{Data}, \text{Parameters})$$

Here:

- New Parameter is the updated value of a specific model parameter (weight or bias).
- Old Parameter is the current parameter value.
- α is the learning rate, determining the step size for adjustments.

  Optimizing Learning Rate Selection through Validation: By following a systematic method, the dataset is split into 80% for training and 20% for validation. Different learning rates, ranging from 0.0001 to 1, are tested during training rounds. After each round, the validation accuracy is checked, and the learning rate that gives the best accuracy is chosen. This method smartly improves how the model gets better and predicts.

- $\nabla \text{Loss}(\text{Data}, \text{Parameters})$ represents the gradient of the loss function with respect to the parameter, calculated over the entire dataset.

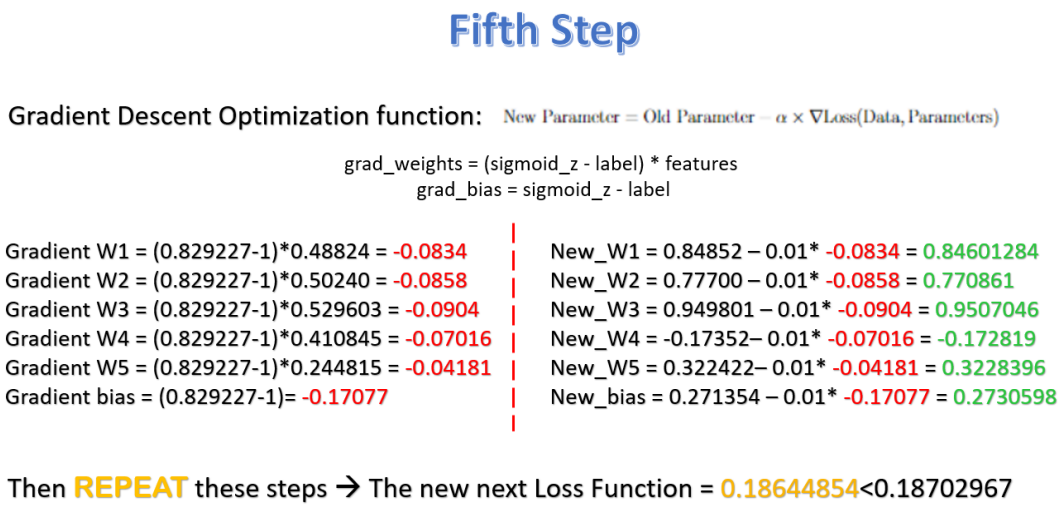

Fig. 3.12. Optimization Symphony: Navigating Toward Enhanced Performance.

In a continuous rhythm, steps 2 to 5 replay like a song on loop. We keep adjusting the model's steps, like tuning an instrument, seeking the best accuracy. With each repetition, we get closer to a skilled performer in sentiment analysis, ready to hit the right notes for accurate predictions.

3.5 Visualization

I created a basic user interface (UI) using Python's Widgets library. This UI makes it easy to test different reviews quickly. You can input reviews and see how the model predicts their sentiments. This simple interface is designed to be user-friendly and lets you easily check how the model responds to different text inputs.

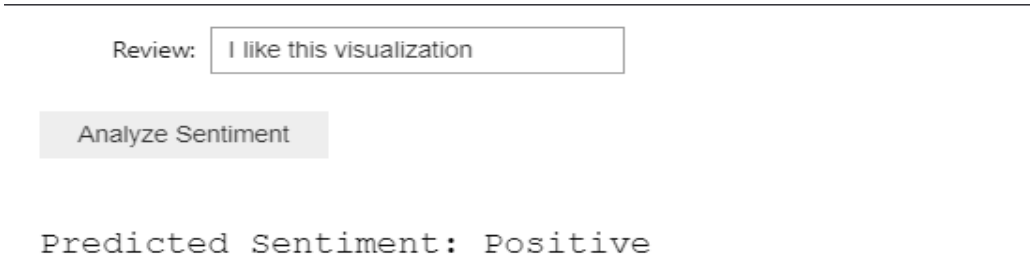

Fig. 3.13. Model Performance Visualization: A Simple Visual Representation of Sentiment Analysis Results

4. EVALUATING SENTIMENT ANALYSIS RESULTS

In the previous sections, We looked closely at the details of Logistic Regression and its role in sentiment analysis. Now, after training our model and making sentiment predictions, it's crucial to evaluate how well it's performing. This section focuses on the evaluation process, where we measure the effectiveness of our model in a systematic manner.

We employ two important tools for this purpose: the F1 score and a concept called a confusion matrix. These tools help us understand how accurately our sentiment analysis model is working. They provide valuable insights into its precision and reliability, allowing us to gauge its overall performance.

By using the F1 score and confusion matrix, we can gain a clear picture of how effectively our sentiment analysis model distinguishes between positive and negative comments. This evaluation process ensures that our model is not only doing its job well but also provides us with valuable pointers for potential improvements, contributing to its ongoing refinement.

4.1 F1 Score

The F1 score is a single metric used to assess the performance of a sentiment analysis model. It takes into account both precision and recall, which are measures of how accurately the model identifies positive and negative sentiments in Amazon reviews. The F1 score balances these two aspects, providing a more comprehensive evaluation of the model's effectiveness in correctly classifying sentiments. It is particularly useful when the dataset has imbalanced classes, ensuring a balanced assessment of the model's overall performance. Higher F1 scores indicate better sentiment analysis accuracy [29].

4.1.1 Precision: Capturing the Model's Precision in Positive/Negative Identification [15]

Precision is a pivotal evaluation metric in sentiment analysis, assessing the accuracy of predictions for both "positive" and "negative" sentiment labels by a model. It is calculated individually for each sentiment class as the ratio of true positive predictions (correctly identified instances of a specific sentiment class, whether "positive" or "negative") to the total number of predicted instances of that sentiment class (including both true positives and false positives):

For "positive" sentiment class:

$$\text{Precision}_{\text{Positive}} = \frac{\text{True Positives}_{\text{Positive}}}{\text{True Positives}_{\text{Positive}} + \text{False Positives}_{\text{Positive}}}$$

For "negative" sentiment class:

$$\text{Precision}_{\text{Negative}} = \frac{\text{True Positives}_{\text{Negative}}}{\text{True Positives}_{\text{Negative}} + \text{False Positives}_{\text{Negative}}}$$

Precision underscores the model's precision in correctly identifying instances of both "positive" and "negative" sentiments. It reveals the model's ability to accurately classify true positive instances while minimizing the inclusion of false positives. Elevated precision values indicate a reduced frequency of false positive predictions, demonstrating the model's effectiveness in sentiment classification for both sentiment classes, "positive" and "negative."

4.1.2 Recall: Profiling the Model's Sensitivity in Positive/Negative Detection [15]

Recall (also known as sensitivity) is a critical evaluation metric in sentiment analysis, measuring the model's ability to identify instances of both "positive" and "negative" sentiment labels. It is calculated individually for each sentiment class as the ratio of true positive predictions (correctly identified instances of a specific sentiment class, whether "positive" or "negative") to the total number of actual instances of that sentiment class (comprising both true positives and false negatives):

For "positive" sentiment class:

$$\text{Recall}_{\text{Positive}} = \frac{\text{True Positives}_{\text{Positive}}}{\text{True Positives}_{\text{Positive}} + \text{False Negatives}_{\text{Positive}}}$$

For "negative" sentiment class:

$$\text{Recall}_{\text{Negative}} = \frac{\text{True Positives}_{\text{Negative}}}{\text{True Positives}_{\text{Negative}} + \text{False Negatives}_{\text{Negative}}}$$

Recall provides insights into the model's effectiveness in capturing instances of both "positive" and "negative" sentiments. It emphasizes the model's ability to correctly classify true positive instances while minimizing the exclusion of false negatives. Higher recall values indicate a reduced occurrence of false negative predictions, showcasing the model's proficiency in sentiment classification for both sentiment classes, "positive" and "negative."

4.1.3 F1 Score: Balancing Precision and Recall

The F1 score harmonizes precision and recall through a balanced formulation, encapsulating their combined impact on sentiment analysis. Mathematically, it is calculated as:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1 score quantifies the model's proficiency in achieving a delicate equilibrium between precision and recall. By considering both false positives and false negatives, it encapsulates the model's capability to provide an accurate and comprehensive sentiment analysis outcome. A higher F1 score signifies a superior balance between precise and sensitive sentiment classification [30].

4.2 Confusion Matrix

A confusion matrix is a tabular representation used in machine learning to evaluate the performance of a classification model. It summarizes the number of correct and

incorrect predictions by comparing the model's classifications to the actual labels. The matrix is structured with four values: true positives (correct positive predictions), true negatives (correct negative predictions), false positives (incorrect positive predictions), and false negatives (incorrect negative predictions). The confusion matrix provides valuable insights into the model's precision, recall, and other evaluation metrics, aiding in the assessment of its classification accuracy and potential areas for improvement [31].

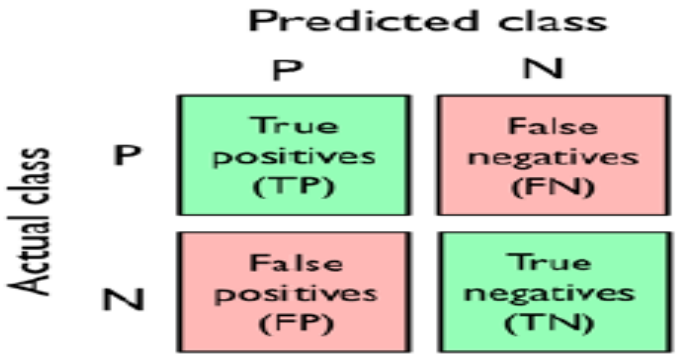

Fig. 4.1. Understanding Sentiment Prediction: The Confusion Matrix [19]

4.3 Experimental Results

In this section, we present the results of our sentiment analysis model on the training, validation, and test sets. We analyze the performance of the model using key metrics including precision, recall, and F1 score.

1. Training Set Results

   - Precision : 0.914
   - Recall : 0.917
   - F1 Score :0.918

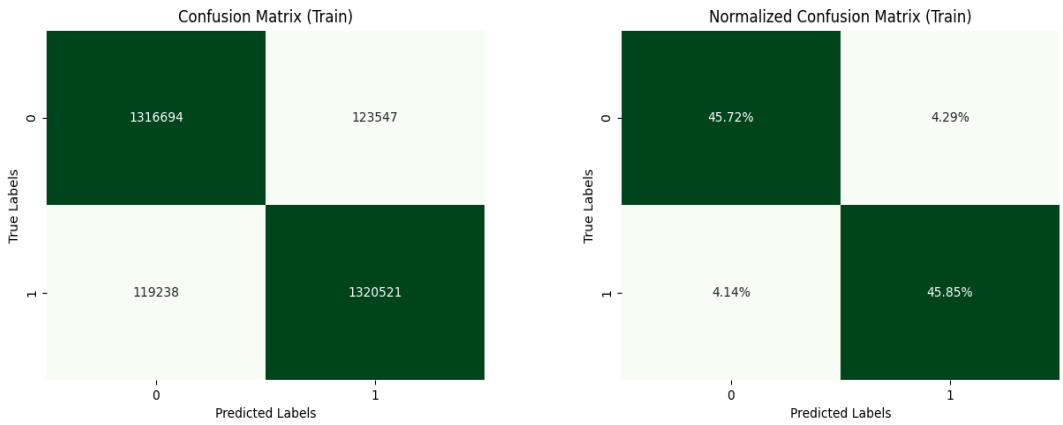

Fig. 4.2. Confusion Matrix for Training Dataset: Evaluating Model Performance on Training Data.

Confusion Matrix for Training:

- True Positives (TP): 1,316,694 (45.72%)
  - These are instances where the model correctly predicted a positive sentiment.
- False Positives (FP): 123,547 (4.29%)
  - These are instances where the model incorrectly predicted a positive sentiment when the actual sentiment was negative.
- False Negatives (FN): 119,238 (4.14%)
  - These are instances where the model incorrectly predicted a negative sentiment when the actual sentiment was positive.
- True Negatives (TN): 1,320,521 (45.85%)
  - These are instances where the model correctly predicted a negative sentiment.

The confusion matrix shows positive results in sentiment analysis, with a significant 45.72% True Positive rate. The model successfully predicts positive

sentiments, aiding accurate sentiment identification. Additionally, a balanced 45.85% of True Negatives demonstrates the model's ability to recognize negative sentiments effectively. This breakdown highlights the model's skill in distinguishing between positive and negative sentiments during training.

2. Validation Set Results

   - Precision : 0.910
   - Recall : 0.913
   - F1 Score : 0.912

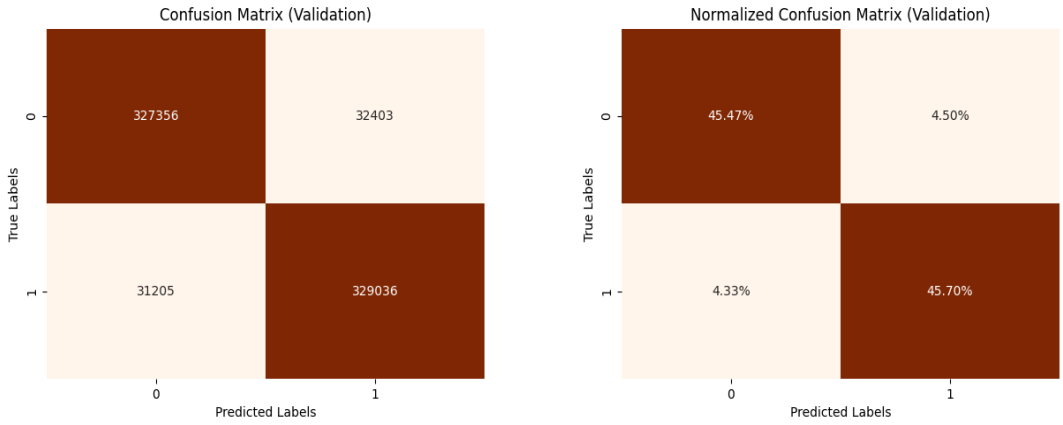

Fig. 4.3. Confusion Matrix for Validation Dataset: Assessing Model Generalization

Confusion Matrix for Validation:

   - True Positives (TP): 327,356 (45.47%)
   - False Positives (FP): 32,403 (4.5%)
   - False Negatives (FN): 31,205 (4.33%)
   - True Negatives (TN): 329,036 (45.7%)

3. Test Set Results

- Precision : 0.904
- Recall : 0.913
- F1 Score : 0.908

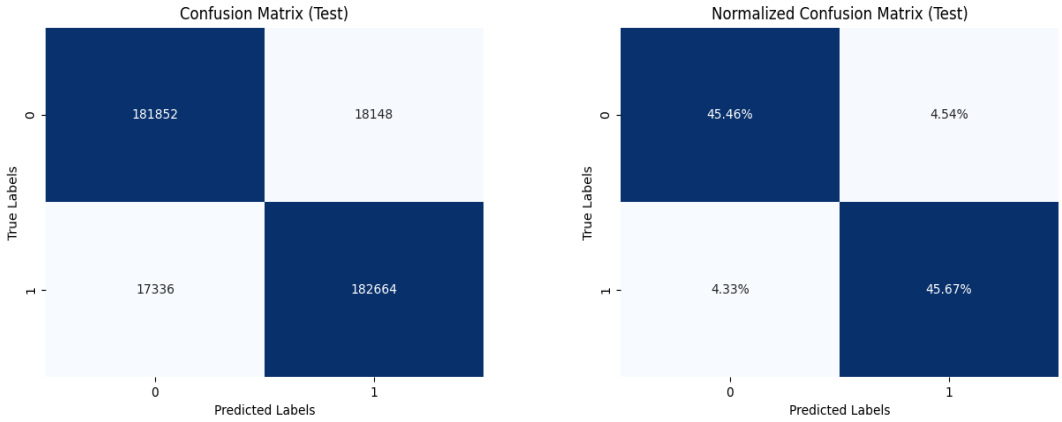

Fig. 4.4. Confusion Matrix for Testing Dataset: Measuring Model Effectiveness on New Data

Confusion Matrix for Test:

- True Positives (TP): 181,852 (45.46%)
- False Positives (FP): 18,148 (4.54%)
- False Negatives (FN): 17,336 (4.33%)
- True Negatives (TN): 182,664 (45.67%)

As observed, the F1 scores for the training, validation, and testing phases exhibit remarkable similarity. This coherence arises from the significant variation in dataset sizes. Specifically, the training dataset comprises an extensive collection of 2,800,000 comments, while the validation dataset encompasses a moderate subset of 720,000 comments. In contrast, the test dataset is more modest, containing only 400,000 comments. Consequently, this convergence in F1 scores across diverse dataset sizes

underscores the model's robust generalization capability, substantiating its adeptness in consistently addressing sentiment classification tasks across varying dataset magnitudes.

4.4 Performance Evaluation on Alternative Dataset

Subsequently, I evaluated the effectiveness of my model on an alternative dataset to validate its performance. For this purpose, I employed movie reviews sourced from IMDB, encompassing a total of 50,000 reviews paired with their corresponding sentiments. After subjecting the dataset to my established preprocessing techniques, I applied the logistic regression model. The results garnered were notably promising, reinforcing the model's proficiency in delivering accurate outcomes.

Movies Dataset Results :

- Precision : 0.901
- Recall : 0.853
- F1 Score : 0.877

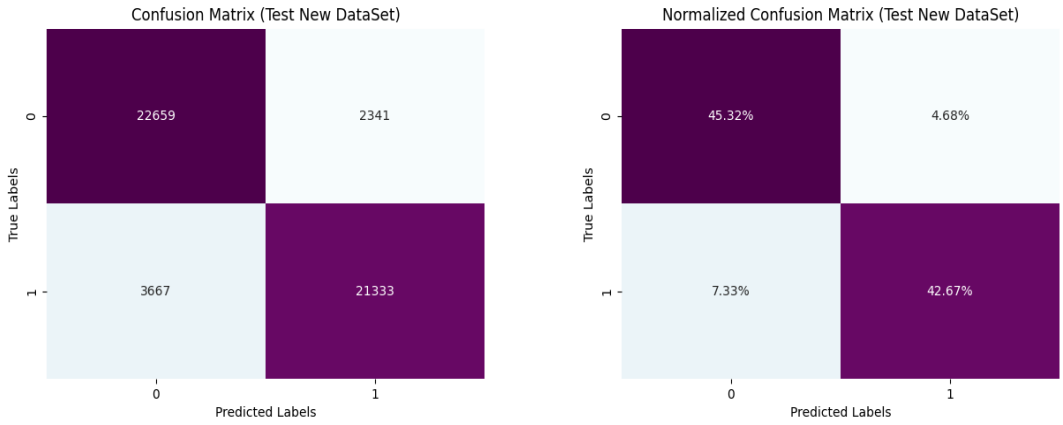

Fig. 4.5. Confusion Matrix for Alternative Test Dataset: Assessing Model Performance on New Data

Confusion Matrix for Test on Alternative Dataset:

- True Positives (TP): 22,659 (45.32%)
- False Positives (FP): 2,341 (4.68%)
- False Negatives (FN): 3,667 (7.33%)
- True Negatives (TN): 21,333 (42.67%)

The test confusion matrix, evaluated on an alternative dataset, presents insightful findings. The notable 45.32% True Positive rate demonstrates the model's ability to correctly predict positive sentiments. Despite a 4.68% False Positive rate, the model maintains a balanced 42.67% True Negative rate, effectively identifying negative sentiments. Additionally, the 7.33% False Negative rate underscores areas for potential improvement in correctly identifying positive sentiments. These results provide a comprehensive understanding of the model's performance on the alternative testing dataset.

Remarkably, the logistic regression model, trained on an extensive Amazon product review dataset comprising 3.6 million entries, exhibited extraordinary adaptability during its evaluation on a distinct movie review dataset encompassing 50,000 reviews. Despite the inherent differences between these two domains, the model achieved an impressive accuracy of 87%, underscoring its capacity to generalize sentiment patterns across diverse subjects. This intriguing revelation suggests the existence of universal emotional cues and shared linguistic conventions within various review contexts. For example, the term 'amazing' can denote positivity whether it pertains to a product or a movie, underscoring the model's ability to identify common sentiment indicators across different topics

5. CONCLUSION

In conclusion, this bachelor thesis embarked on a comprehensive journey through the intricate landscape of sentiment analysis applied to Amazon reviews, employing the reliable framework of logistic regression. The exploration began with a thorough examination of the foundational concepts within machine learning, encompassing sentiment analysis types, natural language processing, and feature extraction methods. The significance of logistic regression was underscored, revealing its prowess in modeling sentiment-based patterns.

The methodology section outlined a careful step-by-step process, explaining how the dataset was prepared by cleaning, removing unnecessary words, and simplifying words. This eventually led to the strong use of TF-IDF and the application of a logistic regression method. The following analysis revealed the effectiveness of this approach, shown by the precision, recall, and F1 score numbers, highlighted by the helpful confusion matrices.

This journey has helped us understand how language, data, and machine learning work together, allowing us to understand people's feelings from text. As sentiment analysis becomes more important in everyday technology, this study shows that logistic regression can provide helpful insights and predictions. Through this thesis, we've learned not just the technical side, but also the creative aspect of sentiment analysis, contributing to the wider world of artificial intelligence progress. Looking back on our journey, we're on the edge of a new era in sentiment analysis, using data and logistic regression to make smarter decisions.

5.1 Future Work

Looking ahead, there are exciting possibilities for further study. We could explore ways to make the sentiment analysis even better by using more detailed information about the reviews and the people writing them. Trying out more advanced techniques like deep learning might help us discover new insights. It could be interesting to see how well our model works in different situations or languages. Also, we could look into analyzing sentiments in real-time and how it can be used with new technologies like language understanding. This could make our analysis even more accurate and useful for online shopping and other areas.

6. APPENDIX

This appendix lists some important sources.

```
def analyze_sentiment(_):
    with output:
        output.clear_output()
        review = text_input.value
        ##############(Review Cleaning)#############
        processed_review = processed_comments([review])
        ##############(Removing Stop Words)#############
        review_no_stopwords=remove_stop_words([processed_review])
        ##############(Stemming : Removing suffixes)#############
        review_stemmed=stem_text([review_no_stopwords])
        ############## Vectorize the stemmed review #############
        review_vector = vectorizer.transform(review_stemmed)
        sentiment = logistic_regression.predict(review_vector)[0]
        ############## Predict sentiment #############
        sentiment_label = "Positive" if sentiment == 1 else "Negative"
        print(f"Predicted Sentiment: {sentiment_label}")
```

LIST OF REFERENCES

LIST OF REFERENCES

[1] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Upper Saddle River, New Jersey, USA: Prentice Hall, 2008.

[2] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[6] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Pearson, 2019.

[7] B. Liu, *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Studies in Natural Language Processing, Cambridge University Press, 2020.

[8] M. Russell and M. Klassen, *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Instagram, GitHub, and More*. O'Reilly Media, 2018.

[9] M. Ganis and A. Kohirkar, *Social Media Analytics: Techniques and Insights for Extracting Business Value Out of Social Media*. IBM Press, Pearson Education, 2015.

[10] G. Miner and Others, *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*. Academic Press, 2012.

[11] J. Silge and D. Robinson, *Text Mining with R: A Tidy Approach*. O'Reilly Media, 2017.

[12] P. Uhr, J. Zenkert, and M. Fathi, "Sentiment analysis in financial markets: A framework to utilize the human ability of word association for analyzing stock market news reports," *IEEE Systems Journal*, p. 7, 2014.

[13] D. Evans, J. McKee, and S. Bratton, *Social Media Marketing: The Next Generation of Business Engagement*. Serious skills, Wiley, 2010.

[14] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media, 2009.

[15] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[16] D. S. Moore, G. P. McCabe, and B. A. Craig, *Introduction to the Practice of Statistics*. W. H. Freeman, 9th ed., 2017.

[17] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: The MIT Press, 2012.

[18] E. Alpaydin, *Introduction to Machine Learning*. Cambridge, MA, USA: The MIT Press, 2014.

[19] S. Raschka, Y. H. Liu, V. Mirjalili, and D. Dzhulgakov, *Machine Learning with PyTorch and Scikit-Learn: Develop Machine Learning and Deep Learning Models with Python*. United Kingdom: Packt Publishing, 2022.

[20] P. Turney, "Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, (Philadelphia), pp. 417–424, 2002.

[21] V. Jagtap and K. Pawar, "Analysis of different approaches to sentence-level sentiment classification," *International Journal of Scientific Engineering and Technology*, pp. 164–170, 2013.

[22] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proceedings of the ACL*, 2005.

[23] J. Cheng, S. Zhao, J. Zhang, I. King, X. Zhang, and H. Wang, "Aspect-level sentiment classification with heat (hierarchical attention) network," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*, pp. 97–106, 2017.

[24] N. N. A. Sinnasamy and N. N. A. Sjaif, "Sentiment analysis using term based method for customers' reviews in amazon product," *International Journal of Advanced Computer Science & Applications*, vol. 13, no. 7, 2022.

[25] C. Rain, *Sentiment Analysis in Amazon Reviews Using Probabilistic Machine Learning*. PhD thesis, Swarthmore College, 2013.

[26] O. Al-Haribi, "A comparative study of feature selection methods for dialectal arabic sentiment classification using support vector machine," *IJCSNS International Journal of Computer Science and Network Security*, vol. 19, no. 1, pp. 167–176, 2019.

[27] W. Chen, C. Lin, and Y.-S. Tai, "Text-based rating predictions on amazon health & personal care product review," in *Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering (NLPKE)*, 2015.

[28] S. Buttcher, C. L. A. Clarke, and G. V. Cormack, *Information Retrieval: Implementing and Evaluating Search Engines*. The MIT Press, 2016.

[29] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," *Proceedings of the 23rd international conference on Machine learning (ICML-06)*, pp. 233–240, 2006.

[30] A. J. Sullivan and P. J. McCarthy, "On the validity of the f-measure," in *Proceedings of the 31st international conference on Machine learning (ICML-14)*, pp. 1381–1389, 2014.

[31] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, 2019.