Sentiment Analysis using Machine Learning Algorithms

Dhrumil Mistry, Kalp Ranpura & Rashika Jain

BS Honors Computer Science, Ahmedabad University
Ahmedabad, India

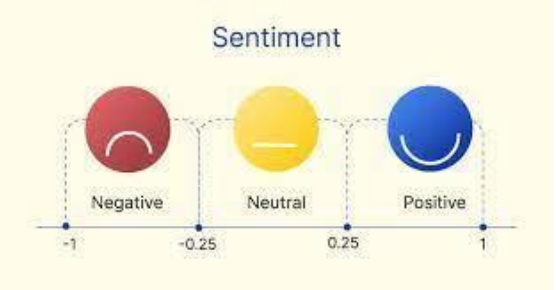
Abstract- Sentiment analysis or opinion mining is the computational study of people's opinions, sentiments, attitudes, and emotions expressed in written language. It has been one of the most active analysis areas in natural language processing and text mining for many years. There are various existing models and algorithms for analyzing user sentiments. They can be classified into knowledge-based, statistical, or hybrid.

Keywords-Sentiment Analysis, Natural Language Processing, Prediction, Artificial Intelligence, Machine Learning, Deep Learning, Python.

1. Introduction

Sentiment Analysis is a powerful approach that enables companies to understand the user emotions in their marketing campaigns. It plays a vital role in consumer loyalty satisfaction, advertisements, and promotional success, monitoring brand reputation on social media, gaining insights from customer feedback, and, most importantly, product acceptance. Understanding consumer psychology allows them to alter their product roadmap with more precision and increases product and brand recall.

Sentiment analysis is a tool used to analyze texts for polarity, i.e., positive to negative. Training the machines learns to detect sentiment automatically without human input. The term emotion-based marketing encompasses emotional consumer responses, such as "positive," "negative," "neutral," "uptight," "disgust," "frustration," and others. These models can detect beyond mere definitions like sarcasm, context, or misapplied words. We can save hundreds of hours by using sentiment analysis as it can process the data and understand it as a human would.



The figure above depicts how the variation in the consumer reviews will help the model learn and predict a likely and accurate output. A substantial number of models fundamentally center around one of the two things, such as subjectivity/objectivity and feature/aspect. We cannot wholly automate sentiment analysis as, eventually, the outcome is to be validated by a human analyst.

2. Literature Survey

Much work has been done on Sentiment analysis, where individuals have utilized different methodologies and technologies. The most utilized apprṣoaches are the Bag of Words (BoW), AI algorithms, and deep learning procedures. The most normal methods used by individuals worldwide incorporate characterization models like Naive Bayes, Support Vector Machine, Random Forest, Decision Tree, KNN Classifier, etc. Albeit every one of the calculations functioned well when the datasets were pre-processed accordingly.

In one of the articles, the author performed sentiment analysis with a Naive Bayes Classifier. The author divided the dataset into positive, negative, and neutral values by rating them larger than 1, lower than 1, and 1, respectively.[3]

In this blog, the author has explained the NaiveBayes algorithm in detail and explained how it works with examples. the author has selected google play reviews as their data set and achieved an accuracy of 85% in the dataset.[2]

3. Implementation

Our entire project was written and compiled in python. We will provide our implementation in a stepwise process:

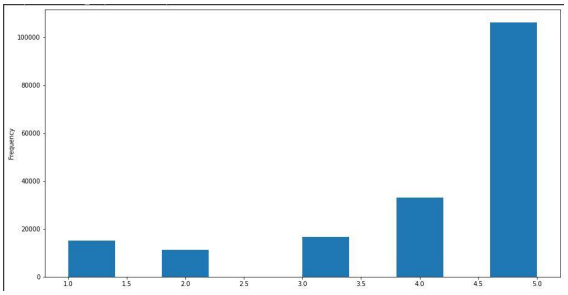
A. Fetching Data

For our data collection, we looked up numerous datasets online. We needed data that had personal opinions in textual form as reviews. So, we chose a data set of Amazon baby products for training our model. It also had rating information which further helped in model training. Although this is a review dataset, there is a lot of comparison and sarcasm involved in the data. The model may not perform well over real-world data. The dataset had three columns, namely Name, Review, and rating.

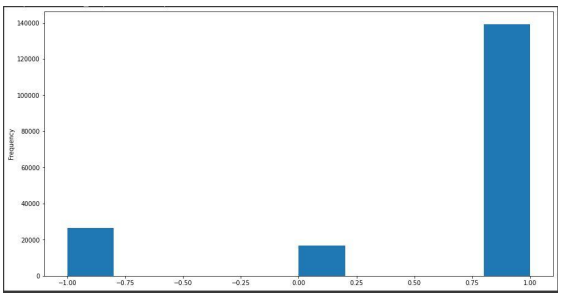
B. Pre-processing the Data

The data we retrieved from Amazon first had to be cleaned for training the model. We first identified the amount of data available in the data set. The Dataset consisted of almost 1,80,000 rows.

After plotting the data, it was known that the dataset had a far more enormous amount of reviews with ratings of more than three compared to the other ratings.

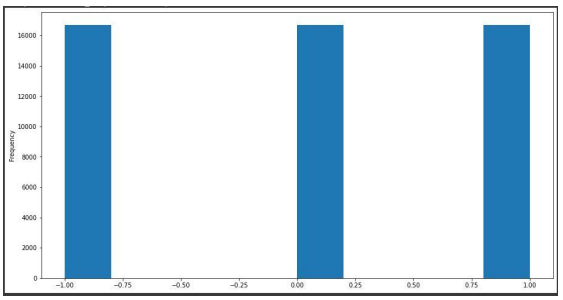


We had to classify the data into Positive, Negative, or Neutral sentiment. Due to this, we have assumed that the reviews with ratings 1&2 would be negative, reviews with ratings 4&5 would be positive, and reviews with ratings precisely equal to 3 would be neutral. So we divided the dataset into only three ratings, i.e., 1, -1, and 0.



Fitting this Data into our algorithm may skew the results more towards positive, which will reduce the model's accuracy, and the model would not work appropriately on real-world data.

Due to this, we had to tweak the dataset to consist of an equal no. of Positive, Negative, and Neutral reviews. So finally, after tweaking the dataset, we had approximately 16,000 rows of each. Thus the dataset consisted of about 50,000 rows.



We had also decided to build a model which classifies the review into only positive and negative sentiments. So for that model, we had dropped all the Neutral ratings from the original dataset and balanced the number of positive and negative ratings. So we had approximately 26,000 reviews of both the sentiments, respectively.

C. Feature Engineering

After balancing the dataset, use several NLP methods and tools to make the content more machine-readable and improve the models' accuracy and efficiency.

The first thing we did was to Lemmatize the reviews so that the no. of words in the vocabulary could be reduced. For effective Lemmatization, we used POS(position of speech) tagging, which tags each word as a noun, verb, adjective, etc. Each word can be lemmatized to a proper word that makes sense.

After lemmatization, we removed all the punctuations from the Lemmatized reviews. We converted the words to lowercase to reduce the noise.

After removing punctuations and converting the content to lowercase, we removed all the stop words with the help of the stop words parameter in the vectorizer.

Also, in the vectorizer, we selected a n_gram range of up to 2 words and a max_df of 0.75 to give the features some weight according to their usage in the data.

D. Implementation of our Sentiment Analysis Model

After cleaning and preprocessing the data, we had two datasets. In both Datasets, we split the data into two. One contains only Positive and Negative Sentiments, and the other contains Positive, Negative, and Neutral sentiments. The algorithms we selected were Multinomial Naive Bayes, Decision Tree Classification, and Logistic regression. We implemented NaiveBayes and Decision Tree in both of our Datasets. However, as Logistic Regression is a binary classifier, we could only use it in our Positive-Negative Dataset.

To find the best parameters for our model from a given range, we used the GridSearchCV method. Although Grid search may take time and computation power, it crosses validates with all the available parameters to give the most accurate result.

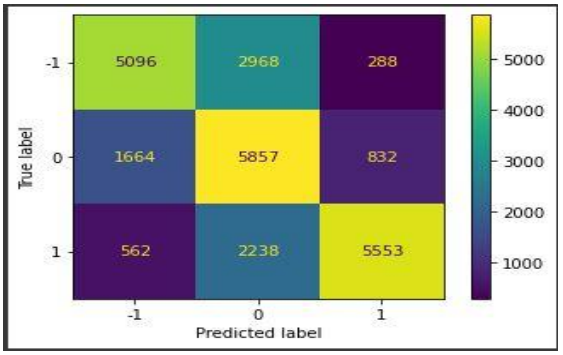
4. Results

In total, we had implemented five models divided among two datasets. So we will discuss the accuracy score of each model along with its confusion Matrix and how it performs on real-world data.

A. **Multinomial Naive Bayes**

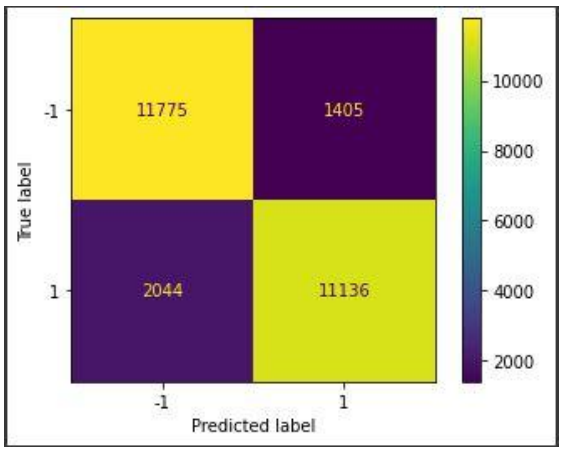
- 3 Class Dataset:

With the help of Grid Search, we found the most optimal parameter for this model to be accurate when alpha is equal to 1. When tested on the test data, it gave us an accuracy of 65.87%. The confusion Matrix obtained is shown below.



- 2 Class Dataset:

With the help of Grid Search, we found the most optimal parameter for this model to be accurate when alpha is equal to 1. When tested on the test data, it gave us an accuracy of 65.87%. The confusion Matrix obtained is shown below.

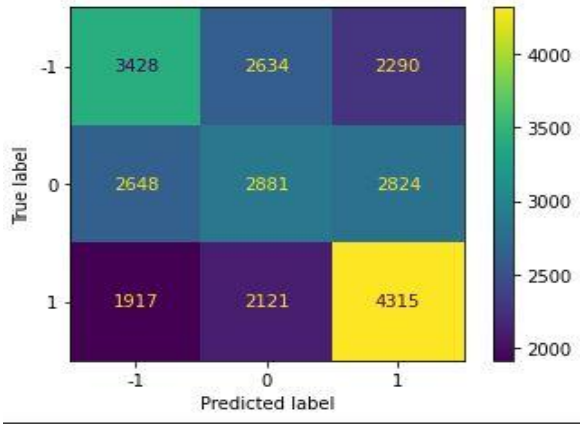


B. **Decision Tree Classifier:**

- 3 Class Dataset:

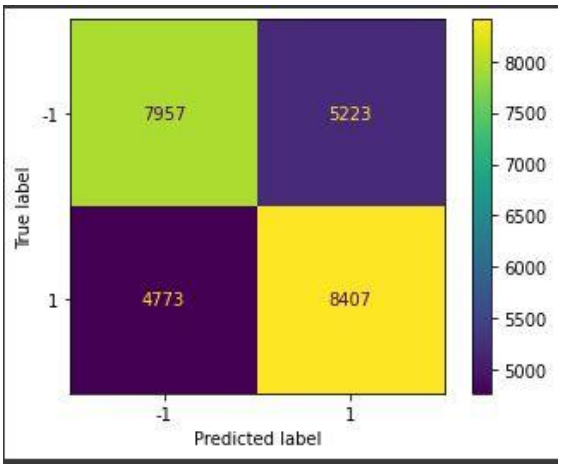
With the help of Grid Search, we found the most optimal parameter for this model to be accurate when (criterion= 'entropy', max_features= 7, min_samples_leaf= 1)

. When tested on the test data, it gave us an accuracy of only 42.397%. The confusion Matrix obtained is shown below.



- 2 class Dataset:

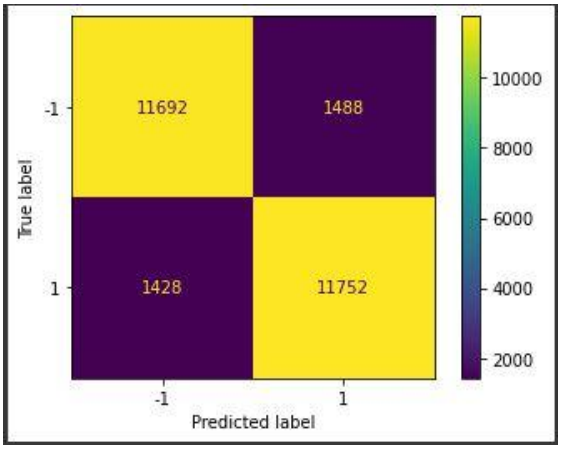
With the help of Grid Search, we found the most optimal parameter for this model to be accurate when alpha is equal to 1. When tested on the test data, it gave us an accuracy of 62.73%. The confusion Matrix obtained is shown below.



C. **Logistic Regression**

This binary classification model can be very accurate if used correctly. This model gave an accuracy of 89% on the test data, which is the highest among all the three models. The confusion matrix shown below can be identified by how accurate

the model is.



4. Conclusion

All our models performed well on the dataset except the decision tree model. Although it worked okay in binary classification, it gave a very low accuracy in 3 class classification. The model worked best on both the dataset and the real-world data. It was observed that as we improved the parameters. In the vectorizer and tweaked the model's hyperparameters, the models worked very well on the real-world data, which was not the case before. One more which is to be noted is that as the model was trained from a product review dataset, there were many reviews where the product rating was low. However, the user had compared the product, so comparative words like "Better," which convey a positive sentiment, may show the sentiment is negative.

ACKNOWLEDGEMENT

We want to thank Prof. Mehul Raval and Ahmedabad University for allowing us to conduct this study as part of our end-semester evaluation project for CSE 523 Machine Learning. Further, we extend our gratitude to the TAs for guiding us through this study.

REFERENCES

[1] Sentiment Analysis & Machine Learning. MonkeyLearn Blog. (2020, April 20). Retrieved March 20, 2022, from https://monkeylearn.com/blog/sentiment-analysis-machine-learning/#:~:text=Sentiment%20analysis%20is%20a%20machine,detect%20sentiment%20without%20human%20input

[2] Performing sentiment analysis with naive Bayes classifier! Analytics Vidhya. (2021, July 13). Retrieved March 20, 2022, from https://www.analyticsvidhya.com/blog/2021/07/performing-sentiment-analysis-with-naive-bayes-classifier/

[3] Foy, P. (2021, July 26). Naive Bayes for sentiment analysis & Natural Language Processing (NLP). MLQ.ai. Retrforved March 20, 2022, from https://www.mlq.ai/sentiment-analysis-with-naive-bayes/

[4] Kuzminykh, N. (2020, October 23). Sentiment Analysis in python with textblob. Stack Abuse. Retrieved March 20, 2022, from https://stackabuse.com/sentiment-analysis-in-python-with-textblob/