From Hashtags to Heartbeats:

Understanding Emotion on Twitter

Sentiment Analysis on sentiment140 dataset and 60,000 additional data scraped from Twitter.

Table of Contents

Abstract

Social media is an integral part of people's lives in today's world. Twitter is a prominent social media tool that is used for a variety of reasons. People use Twitter to share their thoughts, ideas, opinions, and perspectives on various subjects, businesses, goods, and services. Twitter sentiment analysis is a method of determining how people feel about certain subjects or businesses. This project aimed to develop a sentiment analysis system that can analyze the sentiment of tweets and classify them into positive, negative, or neutral categories. To achieve this goal, I combined the sentiment140 dataset with 60,000 additional data scraped from Twitter and trained our model using logistic regression in Python and Scikit-learn. I then integrated the model with the backend and developed a 2-page dashboard using Django Rest Framework as the backend and ReactJS as the front end. The dashboard displays the top 10 trending tags in Toronto from Twitter, along with their sentiments. If the user clicks on sentiment or searches for anything in the search bar, they are taken to the detail page, which shows the sentiment analysis of 100 tweets scraped using Snscrape. The detail page contains various informative sections such as total tweets analyzed, tweets with majority sentiment, total contributors, most liked tweets, most replied tweets, sentiment pie chart, and a bar graph showing devices from which tweets were sent. The results of the sentiment analysis model were evaluated using various performance metrics and achieved an accuracy of 80%.

1 Introduction

Sentiment analysis determines the emotional tone behind a series of words used to understand the attitudes, opinions, and emotions expressed within an online mention. Sentiment analysis has many applications, including social media monitoring, customer feedback analysis, and brand reputation management. In this project, I developed a sentiment analysis system that can analyze the sentiment of tweets and classify them into positive, negative, or neutral categories. However, manually analyzing the mood of many tweets might take time and effort. Hence, I created a web-based dashboard that provides real-time insights and a user-friendly interface to analyze the sentiment of tweets associated with a given hashtag or keyword.

2 Methodology

In this section, I will provide technical details about the methods and approaches used in the project.

2.1 Context and setting of the study:

In this section, I described the context of the study, including the problem we aimed to solve, and the setting in which I conducted the research. I highlighted the significance of our project, which was to provide insights into the public's sentiment towards specific topics, products, or services, which businesses, organizations, or governments could use to make informed decisions. I also provided some background information on sentiment analysis and its associated challenges.

2.2 Specify the study design:

The study was designed to train a machine-learning model to analyze tweets and classify them as positive, negative, or neutral.

2.3 Data collection:

I used a dataset named sentiment140, which contains 1.6 million tweets. The dataset was annotated using emoticons to determine the sentiment of each tweet. We also collected around 60,000 tweets using snscrape. The tweets were then combined and cleaned to create a dataset I used to train our model.

2.4 Identify the main study variables:

The primary study variables in our project were the tweets' sentiment and the associated keywords or hashtags. I also considered other variables, such as the number of contributors, the devices from

which tweets were sent, and the popularity of the tweets in our analysis. The keywords and hashtags were used to search for tweets on a particular topic or theme.

2.5 Data collection instruments and procedures:

I collected the data for our project using two methods. First, I used the sentiment140 dataset, which was freely available online. Secondly, we collected additional tweets to increase the diversity of the data and to include tweets that were not present in the Sentiment140 dataset. For that, we used snscrape, a Python-based tool for scraping data from social media platforms like Twitter. To fetch the additional top 50 trending tweets, we used Tweepy, a Python library for accessing the Twitter API. We used the API to search for tweets using specific hashtags and keywords related to the topic of our project. The tweets were collected in real-time and stored in a cache database to reduce the server load.

2.6 Data Preprocessing:

2.6.1 Data Cleaning:

The first step in data preprocessing is data cleaning, which involves removing unnecessary data, such as URLs, emojis, special characters, and HTML tags. I used Python's regular expression library, re, to remove all these unwanted characters from the tweets.

2.6.2 Eliminating bias:

In the sentiment140k dataset, initially, there was a binary label with just positive and negative data as seen in the graph below:

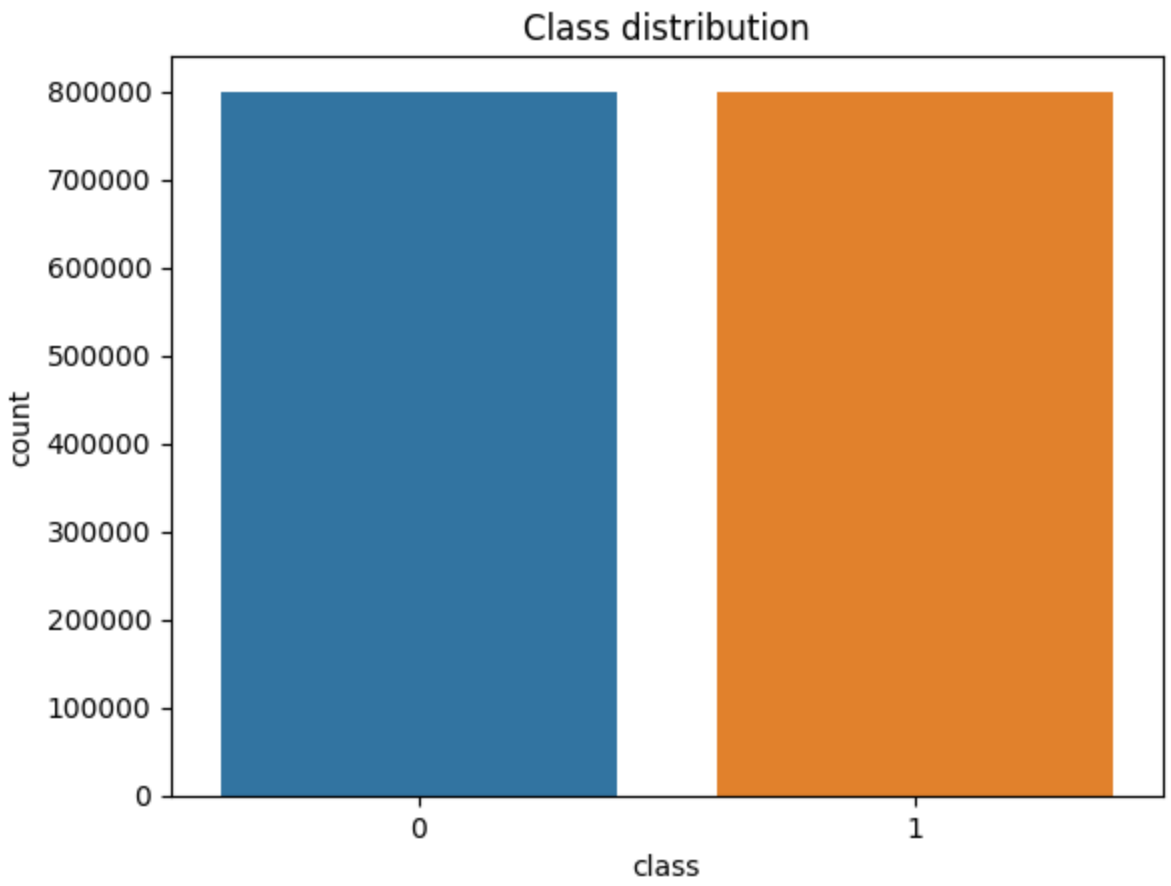

Fig no: 1. Class distribution of the initial Sentiment140 dataset.

So, I used the Vader library to classify the tweets into 3 categories: `positive`, `negative` and `neutral`. But the problem was that data was unevenly distributed as seen in the countplot below:

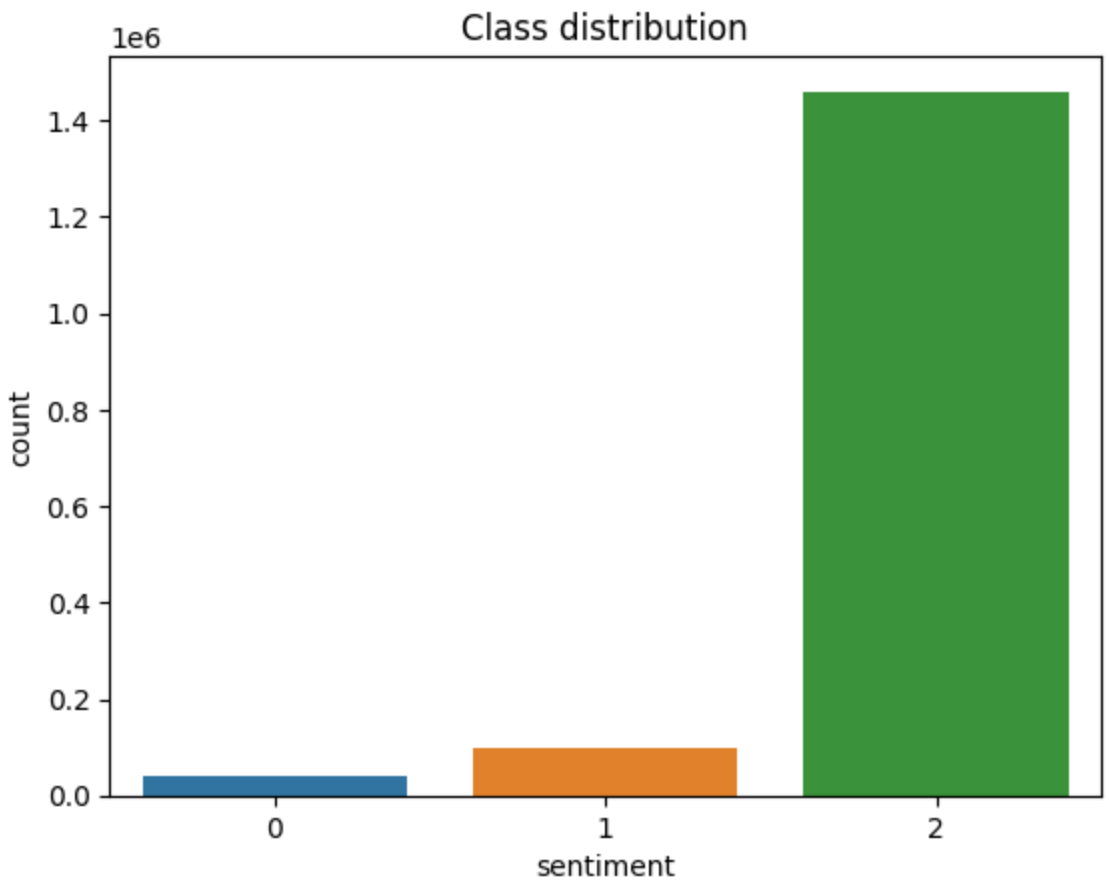

Fig no: 2. Class distribution after classifying tweets with Vader

The situation was the same for the data we scraped too. The number of neutral data was way too more than positive and negative combined. So, we removed the biases and made the data evenly balanced.

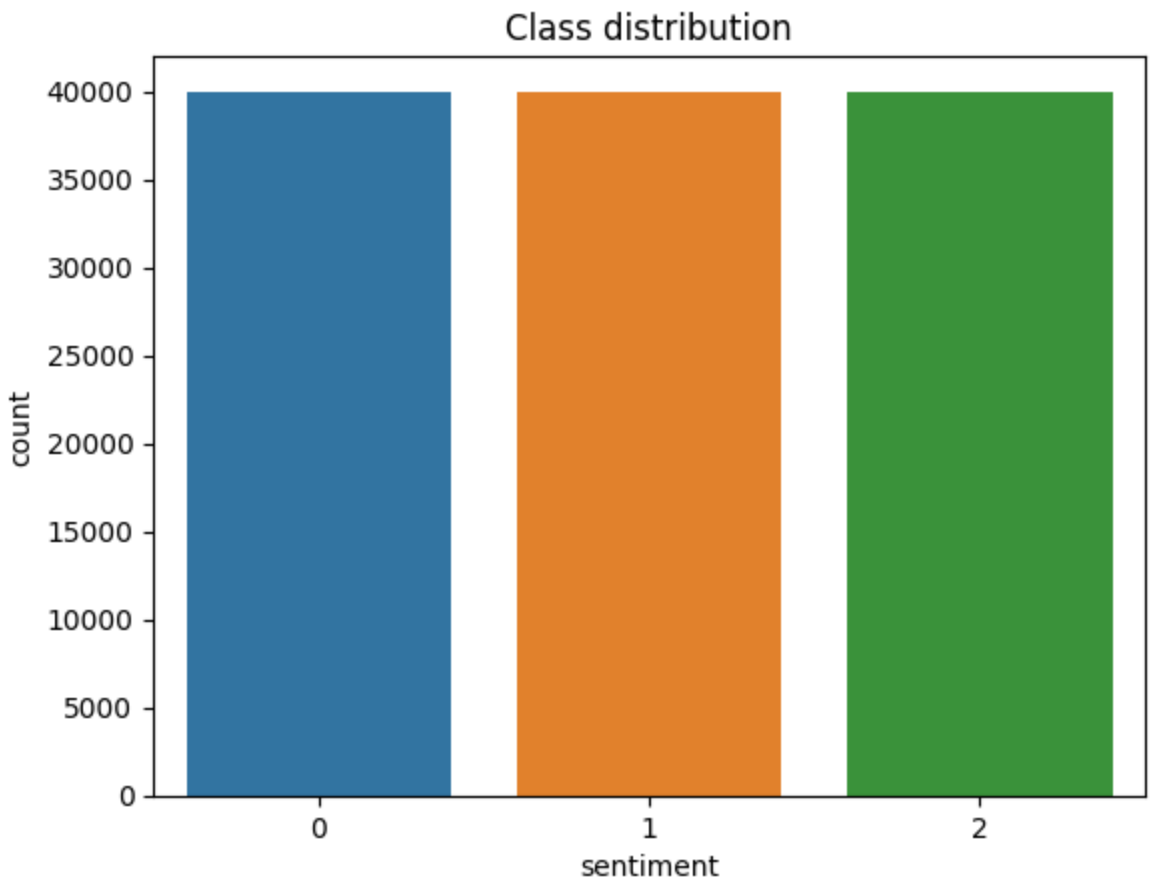

Fig no: 3. Class distribution after removing biases in data

2.7 Text Preprocessing:

Text preprocessing involves cleaning and transforming the text data to prepare it for sentiment analysis. The steps involved in text preprocessing are discussed below:

2.7.1 Tokenization:

For tokenizing, I used the NLTK library's word_tokenize function to perform tokenization on the text data. Each word in the row was tokenized and stored in a data frame for further processing.

2.7.2 Stop Words Removal:

Stop words are common words such as “those”, “this”, “is”, “they”, and “and” that do not add any meaningful information to the text data. So, we removed stopwords from the text data using the NLTK library's stopwords function.

2.7.3 Lemmatization:

Lemmatization was applied which uses a dictionary-based approach that considers the context along with the word. I used the WordNetLemmatizer algorithm from the NLTK library to perform lemmatization on the tokens and store them in a new data frame column.

2.8 Outline analysis methods:

After preparing the dataset, I used Logistic Regression to build the sentiment analysis model. I used the scikit-learn library in Python to implement the Logistic Regression model. Bag-of-Words approach was used to represent the text data as numerical features. The Bag-of-Words approach is a simple technique that counts the frequency of each word in the text and represents it as a vector. I used the CountVectorizer function in scikit-learn to implement the Bag-of-Words approach.

After training the model, I evaluated its performance using various metrics such as accuracy, precision, recall, and F1 score. A confusion matrix was also generated to visualize the performance of the model.

For the web part of the project, I used Django Rest Framework as the backend and ReactJS as the front end. I used tweepy to fetch the top 50 trending hashtags in Toronto and displayed the top 10 on the home page. I used snscrape to scrape a maximum of 100 tweets for each searched keyword on the detail page. The sentiment of the tweets was calculated on the backend and displayed on the front end. I used Redis cache to store the scraped tweets for the home page and the searched

keyword results. The cache was refreshed every 30 minutes for the home page and every 5 minutes for the searched keyword results.

Overall, our methodology consisted of collecting and preprocessing data, training a Logistic Regression model, and integrating the model with a web application using Django and ReactJS.

3 Results:

I evaluated the performance of the sentiment analysis model using various metrics such as accuracy, precision, recall, and F1 score. Firstly, the model was trained on the whole data, where the amount of `neutral` data was more than `positive` and `negative` combined. There, I got average accuracy of 94%, but the precision, recall and f-1 score for `negative` data were too low, i.e. precision: 68%, recall: 48%, f1-score: 56%. We can also see the abnormality in the confusion matrix below:

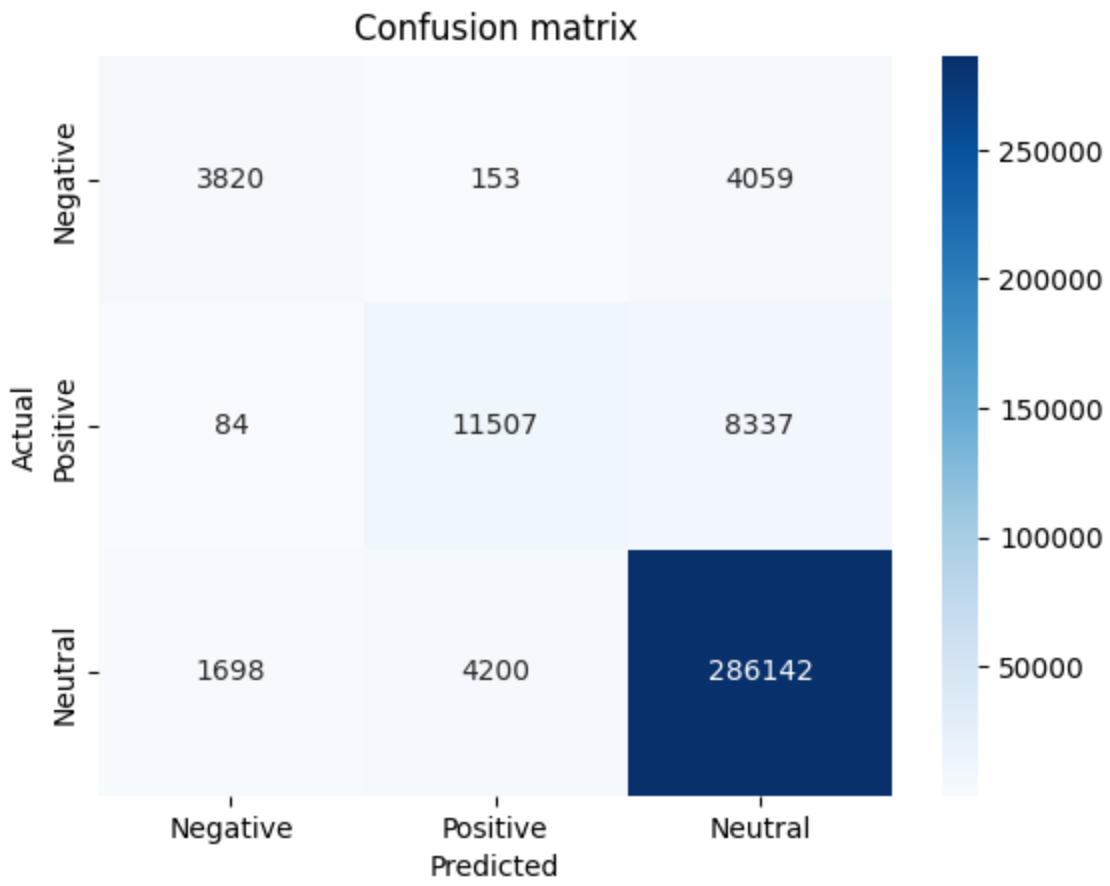

Fig no: 4. Confusion matrix of the initial model with biases in data

The confusion matrix shows that the model was able to correctly classify most of the positive tweets, but it struggled with classifying negative tweets.

Later, we trained our model again with less but balanced data. The model achieved the following results:

- Testing accuracy: 92.7%
- Training accuracy: 96.99%

Label	Precision	Recall	F1-score
Negative	92%	96%	94%
Positive	92%	95%	93%
Neutral	95%	87%	91%

Table no 1: Accuracy Scores for each label

Below is the confusion matrix for the final model:

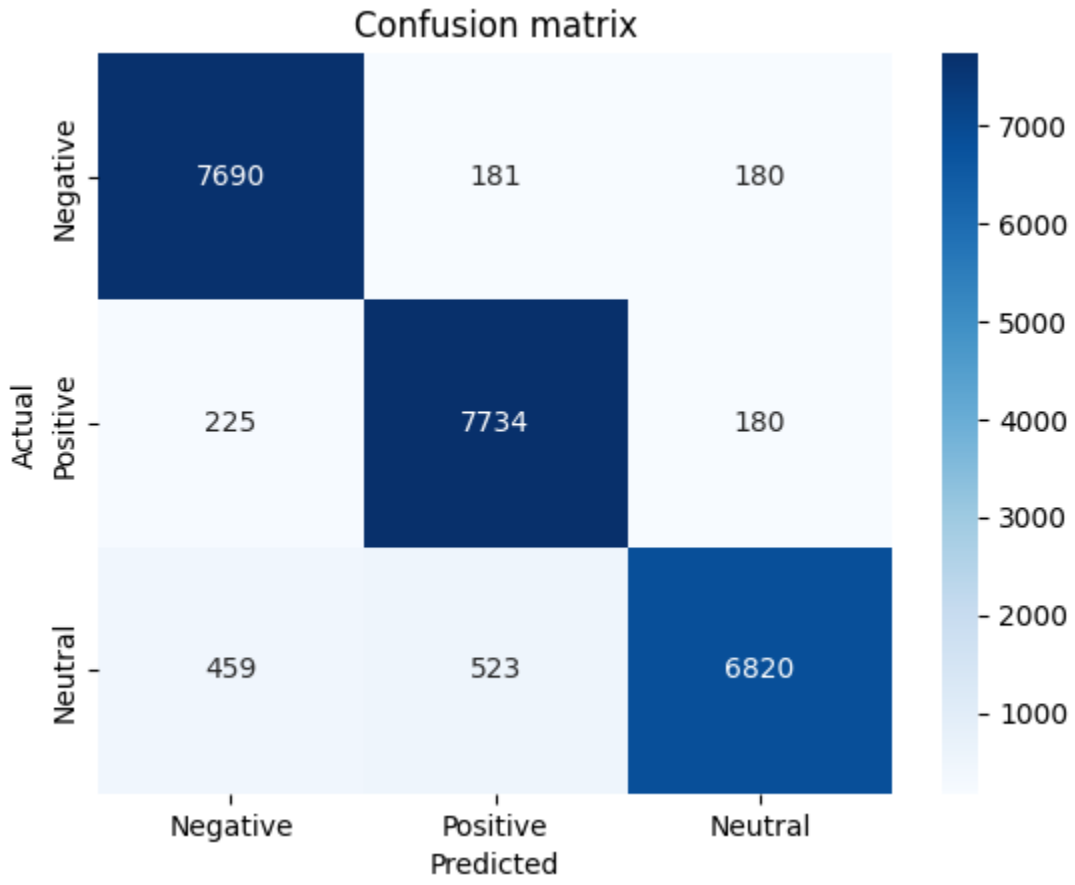

Fig no: 5. Confusion matrix of the final model without biases in data

In terms of the web application, I was able to successfully fetch the top 10 trending hashtags in Toronto using tweepy and display them on the home page. I was also able to scrape and analyze tweets on the detail page and display various statistics such as total tweets analyzed, sentiment pie chart, and bar graph of devices from which tweets were sent. Below are a couple of snapshots of the web application:

Fig no: 6. Homepage consisting of the current trending events

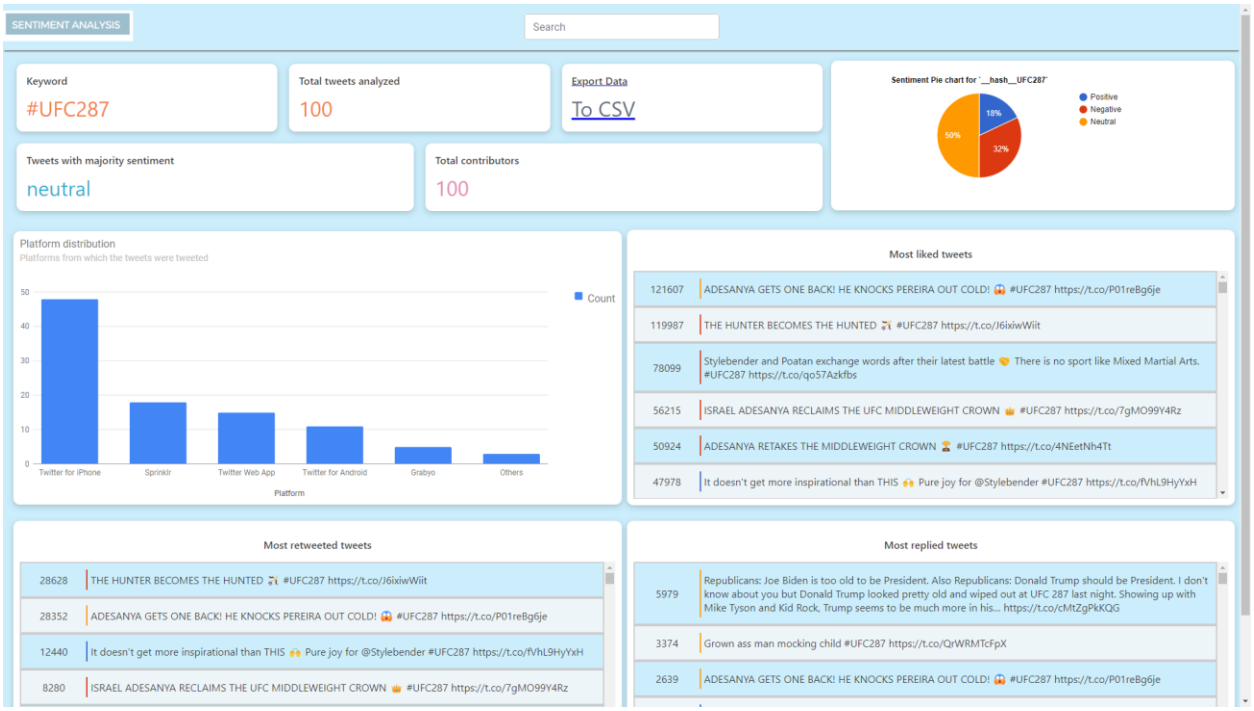

Fig no: 6. Page consisting of detailed analysis of the tweets scraped.

Conclusion:

Our study aimed to develop a sentiment analysis model using logistic regression and integrate it into a web-based dashboard that provides real-time sentiment analysis of tweets related to trending topics in Toronto. The model achieved an accuracy of 92.7% on the test dataset and was integrated into a web-based dashboard that allows users to view sentiment analysis of tweets related to trending topics in Toronto. This study shows that sentiment analysis can be a valuable tool for understanding public opinion on various topics. The dashboard provides a real-time and easy-to-use platform for monitoring sentiment on Twitter. I have also provided a feature on the dashboard to export the analyzed tweets into a csv format so that the users can take reference for future use.

Future scope:

This study has several future research avenues. One possible direction is to develop more advanced sentiment analysis models that can account for linguistic nuances and context. Another direction is to expand the scope of our dashboard to include sentiment analysis of other social media platforms such as Facebook and Instagram. I also plan to incorporate sentiment analysis of images and videos in the future.

References:

[1] Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, 1(12), 2009.

[2] Twitter Inc. (2021). Twitter Inc. Q2 2021 Earnings Release. Retrieved from https://investor.twitterinc.com/financial-information/quarterly-results/default.aspx

[3] Sentiment Analysis of User-Generated Twitter Updates Using Various Classification Techniques, R. Parikh and M. Movassate "2009 Final Report for CS224N