Sentiment Analysis

Prabhath Reddy Gujavarthy, Reshma Chowdary Bobba, Sathvick Reddy Narahari and Vinit Kanani

Nov 2021

Abstract

TO DO

Introduction

The exponential growth of people using the internet has allowed them to share their views, opinions and stories on various web platforms such as Facebook,Twitter or Blogger. With such vast amounts of data, it is difficult to know the emotion of the data. Opinion mining is a critical field to analyze this largely unstructured information to know the current trends or predict the future trends.The objective of our project is to perform sentiment analysis on the twitter dataset to know the underlying intent or tone of the tweets made. The Sentiment140 dataset is collected using Twitter Search API for one such study, and the tweets are labeled on the assumption that positive emoticons such as :) denote positive emotion and negative emoticons such as :(denote a negative emotion. We are using various machine learning algorithms for performing the classification and as an evaluation metric.

Data Preprocessing

We performed various data cleaning techniques before performing classification. Firstly, we checked for missing values in the dataset, and luckily the dataset was compiled properly so they were labeled properly and had no missing values. Since the data is collected from the web it is not very clean and has many stopwords in it. Strings such as 'http', '@..', '#. . . ', adds no significant meaning to the model, and unnecessarily takes up disk storage and increases the time complexity of the model. IMG-1 The total mentions of those characters and tags in the dataset makeup to more than 800k, and so we removed them to improve the model performance. There are many more stopwords that are pre trained in the natural language toolkit(NLTK) library, that take up to about 15% of the total storage taken up by the dataset. We defined a function to remove all those stop words defined in the library, and called it on the train set. ## Train & Test Data Labels:

The data in the train set is labeled as either 0 or 4. '0' meaning negative sentiment and '4' meaning positive sentiment. But in the test data, there is an additional label '2' which says that the tweet is neutral. This is a serious problem as the model trains on only two sets of labels and while testing all of the data with a different label is immediately considered wrong.

We had three different approaches to this problem: Approach which involved removing all the neutral labels from the test An approach which involves classifying all the neutral('2') labels as positive('4') A classic approach where we used a part of train data, set aside to be our test data.

We tried to mix and match all these approaches in our various methods while testing different models. Our data is labeled, so many linear approaches can be used to perform the classification, which is our objective of doing this project. The methods we considered to analyze the sentiment were Logistic Regression, Decision Tree Classifier, Naive Bayes classifier, LinearSVC, SVD, XGBoost Classifier and Random Forest Classifier. All these methods use different approaches to learn from the data and each approach yields different results.

Data Visualization

We plotted word clouds for all the positive and negative labels, to visualize the most frequent words after removing the stopwords.

Word cloud for the entire dataset

IMG 2

Word cloud for positive tweets

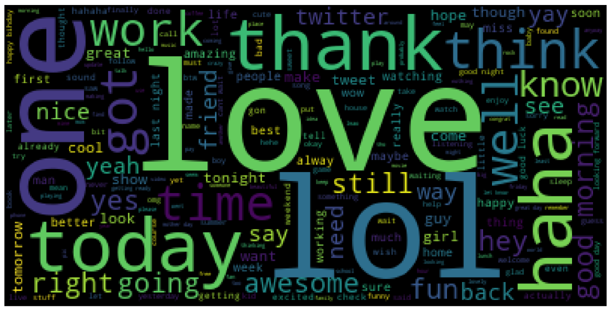

Figure 1: Positive word cloud

Word Cloud for negative tweets

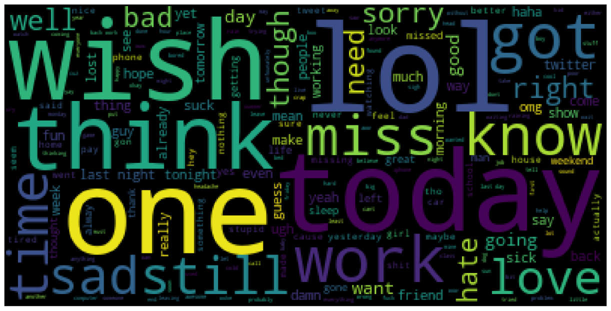

Figure 2: Negative word cloud

Observations Learning from the data: There's clearly a pattern between the words which we generally define as either good or bad. Words such as 'work', 'today', 'hate' etc., usually refers to a negative emotion, while words such as 'good', 'thank', 'like' are referred to as something positive. We have also plotted a barchart to determine the top-10 most used words in the dataset. This can be considered as one of our evaluation metrics as we can visually infer from the data and know that the model learnt something. However it is not very accurate.

Methods

Our data deals with text and its keywords. Since the machine doesn't actually know what those keywords actually are, it needs to learn from that data to know which words determine which sentiment. We used the Tf-Idf vectorizer by SciKit Learn for doing that. It adds weights to a word based on the total number of times the word has appeared in the dataset. The function can also retrieve the number of features we require based on our input. 1. Preprocessing 2. SVD 3. Decision Tree 4. Naive Bayes 5. Topic Modelling

Comparisons

Example Analysis

Conclusions

References