Tweet Sentiment Analysis Through Political Tweets

Winston Zhang, Michael Chen, Gokul Ganesan, Michael Wang

Abstract

Applications like twitter contain a large amount of information that could potentially be useful for many areas of studies. In this paper, we present a sentiment analysis of tweets related to the 2020 U.S. election. We propose a system using traditional TF-IDF, Delta TF-IDF and a support vector machine. Our experiment result shows that Delta TF-IDF statistically outperforms traditional TF-IDF in this particular task.

1 Introduction

Sentiment analysis is a natural language processing technique that involves extracting information from text data and determining the overall sentiment of the text. Sentiment is usually divided into positive, negative, and neutral. Sentiment analysis is commonly used on large datasets in order to gauge the sentiment towards a particular event, service, or product. In this paper, we examine two related weighting schemes' effectiveness in sentiment analysis.

The dataset we used includes tweets related to the 2020 US Presidential Election, collected between July 1, 2020 and November 11, 2020. The dataset uses the VADER (Valence Aware Dictionary and Sentiment Reasoner) algorithm to produce a score. The VADER algorithm is a sentiment analysis algorithm that is designed for social media dataset. It takes into consideration the context in which the word is used as well as other social media features such as emojis. For example, it will take into account negative words that can negate a word's meaning, such as "not cool" and determine a score accordingly. The VADER algorithm is suited for informal and abbreviated languages, such as slang, which is commonly used on social media platforms.

In this paper, we use both TF-IDF(term frequency-inverse document frequency) and Delta TF-IDF as weighting schemes and compare the performance of Delta-TFIDF to the baseline performance of TF-IDF.

2 Related work

Justin Martineau and Tim Finin presented an improved feature space for text categorization tasks by extending based on the traditional term frequency-inverse document frequency weighting method. Based on the idea behind TF-IDF, which weighs words based on the distribution of the words, the authors proposed an extension named Delta TF-IDF. Delta TF-IDF takes into account the difference in word distribution between categories of datasets in order to identify and up-weight the words with differentiated distribution between categories. With these features, this weighting scheme is well-suited for tasks like sentiment analysis. In the paper, the authors present an empirical evaluation of Delta TF-IDF on a number of sentiment analysis tasks, showing that it outperforms traditional TF-IDF and a number of other weighting schemes.

Man LAN et al. presented a similar weighting scheme named relevance frequency to Delta TF-IDF after reviewing the expediency of using TF-IDF as the sole weighting scheme in text categorization. Relevance frequency is acquired by calculating the relative frequency of each term in a document and comparing it to the relative frequency of each term in a corpus of

documents. Although not provided with the detailed mathematical derivation process of the relevance frequency scheme like Justin Martineau did in the Delta TF-IDF paper, the authors manifested that the term assigns more appropriate weights to the terms in terms of different documents compared to traditional TF-IDF using comparison between experiment data weighted by TF-IDF and TF-RF(relevance frequency).

Munir Ahmad et al. presented a detailed procedural guidance for sentiment analysis tasks using TF-IDF as the weight scheme and support vector machine as the classifier. The detailed process includes the utilization of stopwords Handler and tokenizer for pre-processing of the data, using traditional term frequency-inverse document frequency to weight the terms based on the distribution, support vector machine to classify the weighted vector and ways to calculate precision, recall and F-score using the experimental results.

3 Methods and Implementation

3.1 Overview

As shown in Figure 1, our method is first to pull the content of the tweets from twitter using Twitter API and the 'tweet_id' in the U.S. election dataset and pre-process the dataset using various traditional techniques then vectorize the data using TF-IDF and Delta TF-IDF and finally train the support vector machine using the vectors.

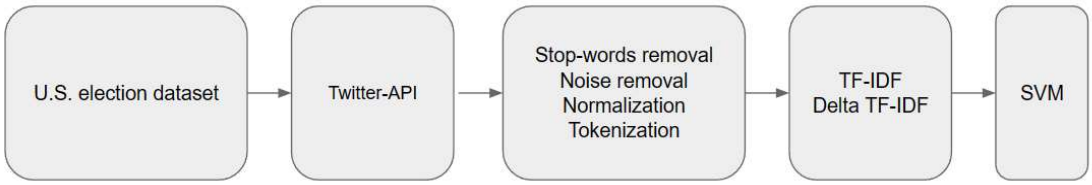

Figure 1: Proposed Methodology

3.2 Preprocessing

The US election dataset we used needed to be preprocessed. Preprocessing data is an important step to ensure the data is in a suitable format. The dataset has columns representing "Tweet-ID", "Created-At", "From-User-Id", "To-User-Id", "Retweet-Count", "Party-Name", "Score", "Scoring-String", "Negativity", "Positivity", "Uncovered-Tokens", and "Total-Tokens". For our program, we needed the actual tweet text but were only given the tweet ID. To solve this issue, we used the Twitter API to extract the tweet text using the tweet ID. Of the columns in the dataset, only a few would be needed for our program. Python's csv library was used to read data from the .csv file and to create an iterable object in which we can perform loops and other functions to manipulate the data. Python's re library was used to remove tweet texts that were not relevant to the overall tweet (ie. @ usernames). Regular expressions were used to only keep the text we wanted.

A common step in preprocessing is stop word removal. It involves removing stop words, or common words that do not contribute much to analyzing text. Removing these words will

reduce the size of the text and make processing data easier. Examples of stop words used are “a”, “the”, “an”, “and”, “or”, “but”, etc. The stop words were predefined in a list saved in stop_list.py.

After reading the data from the dataset and completing the stop word removal, it was checked to see if there are any missing or incorrect values. For example, it was checked if the “Party-Name” column only contained political parties and if the “Tweet-Text” column contained strings. After retrieving only the necessary information, the dataset was then separated into positive and negative tweets. We keep the same amount of positive and negative tweets in order to cancel out variables in the Delta TF-IDF formula when calculating the scores. Whether or not a tweet is considered positive or negative was determined by the values in the “Negativity” and “Positivity” columns. The higher value determines the overall sentiment of the tweet. After preprocessing, we ended up with “positive.txt” and “negative.txt” where each line represents a tweet and is in the form of [“Party Name” “Tweet-Text”].

Using the two .txt files, we were able to calculate the Delta TF-IDF values for each tweet. The data is then processed through a SVM. SVM (support vector machine) are used for classification. It is useful because it can handle linear and non-linear data. SVMS are also efficient in terms of memory and computation.

3.3 Vectorization

3.3.1 TF-IDF

TF-IDF(term frequency-inverse document frequency) is a common method for weighing importance of terms in documents using the distribution of the terms in the overall dataset. The scheme down-weights the common terms and up-weights rare terms in documents based on the assumption that rarer terms provide more information about a specific document’s features.

3.3.2 Delta TF-IDF

TF-IDF weights terms based on their distribution in the overall dataset, enabling the differentiation of important and unimportant terms. However, in text categorization, the scheme ignores the difference in the terms’ distribution across different categories because the inverse document frequency only reflects the terms’ distribution in the overall dataset. With this disadvantage in mind, we can use an extension of TF-IDF by calculating the difference between a term’s distribution in multiple categories, hence Delta TF-IDF. Assuming we have two categories in a text classification task named positive and negative, we have:

$$Delta\ TFIDF_{t,d} = TF_{t,d} \times ln(\frac{|P|}{|P_t|}) - TF_{t,d} \times ln(\frac{|N|}{|N_t|})$$

where $TF_{t,d}$ is the traditional term frequency of term t in document d, $|P|$ and $|N|$ is the document counts in respective data categories, $|P_t|$ and $|N_t|$ are the document counts in respective categories that contain term t. By using balanced dataset for both categories, we can have:

$$Delta\ TFIDF_{t,d} = TF_{t,d} \times ln(\frac{|N_t|}{|P_t|})$$

In this way, we have a scheme that could achieve the effect of TF-IDF and reflect the unevenness in a term's distribution.

3.4 Classification

3.4.1 Support Vector Machine

Support vector machines (SVMs) are a type of supervised learning algorithm that can be used for classification, regression, and outlier detection. SVMs are based on the idea of finding a hyperplane in a high-dimensional space that maximally separates different classes. In the case of sentiment analysis of tweets, the classes could be positive and negative sentiment.

There are several reasons why SVMs might be a good choice for this use case. First, SVMs are very effective in high-dimensional spaces, which is often the case when working with text data. This is because the number of dimensions (i.e., the length of the feature vectors) is typically much larger than the number of samples, which can make it difficult for other algorithms to find patterns in the data. SVMs, on the other hand, are able to handle high-dimensional spaces very well and can still produce good results even when the number of dimensions is much larger than the number of samples.

For this use case, SVMs are ideal because they are robust to overfitting. This is important when working with text data because it can be easy to overfit to the training data if the feature space is very large and complex. SVMs use a regularization parameter to control overfitting, which helps to ensure that the model generalizes well to unseen data.
In addition to their effectiveness in high-dimensional spaces and their robustness to overfitting, SVMs are also very efficient. They can be trained quickly, even on large datasets, and they have a low memory requirement. This makes them well-suited for real-time applications, such as sentiment analysis of tweets in near real-time.

SVMs can be used in multiple 'kernel' modes such as linear, non-linear, polynomial, and radial basis function (rbf) which define how the hyperplane is fitted to the data, which can be hard to decide ideally since the high dimensionality of the data can make the data hard to visualize and therefore predict the perfect kernel to use. We chose to use a linear kernel as it is the least performance intensive and provides negligible drawbacks in accuracy and precision. This likens the SVM to a Linear Regression in a multi dimensional space (the feature space).

4 Evaluation

There are several criterion that can be used to evaluate the performance and reliability of a sentiment analysis model - the most popular of which (for a classification model such as this one) are accuracy, precision, recall, and F-score, which we use to enable better comparative analysis to the relevant texts we reference.

4.1 TF-IDF Model Metrics

Confusion Matrix:

9021	2822
2279	9879

Accuracy	Precision	Recall	F-score
78.75%	76.17%	79.83%	0.7796

4.2 Delta TF-IDF Model Metrics

Confusion Matrix:

5743	284
627	5346

Accuracy	Precision	Recall	F-score
92.41%	95.29%	90.16%	0.9265

4.3 Evaluation Writeup

This data is the result of the model's classification after being trained over 50,000 tweets, with both vectorization methods being compared. The two models were tested on an identical set of 12,000 tweets. The Delta TF-IDF performed drastically better with respect to its F-score (0.9265) compared to the TF-IDF trained model (0.7796).

Although TF-IDF is capable of weighting the importance of terms based on their overall distribution, it lacks the ability to measure the differences of terms' distribution across categories thus Delta TF-IDF was proved to be more effective in this case.

5 Reflection and Limitation

5.1 Dimension Explosion

The most obvious problem we encountered during the course of the project was the excessively large data generated by the TF-IDF and Delta-TF-IDF project. During the project, we experimented with different datasets of different sizes and discovered a decrementing marginal dimension increase with the increase of data size. However, the dimension produced by every term remains extremely large where the size of the produced vector file is about 200 times the size of the original data file on average.

5.2 Arithmetic Properties of Delta TF-IDF

Aside from the dimension oversize, there are some interesting arithmetic properties about Delta TF-IDF. The arithmetical benefit of Delta TF-IDF is that it is easy to compute and we can simply ignore the terms which have the same distribution among different categories because in this case, we will get a Delta TF-IDF as logarithm of 1 which is 0. As a result, it is safe to ignore the terms which have the same distribution over categories. However, in the situation where a term is absolute unique[1] to a category, we need other ways to fit the term into the frame of Delta TF-IDF because the formula tells us that in this case, we can either suffer from division by zero or logarithm of zero which is undefined. We can, however, use the relevance frequency term-adding *e* or 2 in the logarithm depending on the base-but this method could not resolve the division by zero problem. In our experiments, terms with absolute unique distribution were abandoned. Although with sufficient data, this shortcoming did not cause much damage to the result, it is still worthwhile to point out that many experiments using random sampling may be subjected to this problem.

6 Conclusion

In this paper we implemented a sentiment analysis system using TF-IDF, Delta TF-IDF and a support vector machine. Using the U.S. 2020 election tweets, we see that Delta-TF-IDF statistically outperformed traditional TF-IDF in specific text classification tasks, in this case, sentiment analysis. The result proves that a weighting scheme which is capable of reflecting the unevenness of terms' distribution between categories is more appropriate for text classification. However, during the research, we also notice some disadvantages of Delta TFIDF in terms of its size and arithmetic properties.

[1] absolute unique distribution refers to the situation where a term only appears in one category

7 References

1) Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. *Machine Learning: ECML-98, 137–142.*

2) Martineau, J., & Finin, T. (2009). Delta TFIDF: An Improved Feature Space for Sentiment Analysis. *International Conference on Weblogs and Social Media*.

3) Ahmad, M., Aftab, S., & Ali, I. (2017). Sentiment Analysis of Tweets using SVM. *International Journal of Computer Applications*, *177*(5), 25–29.

4) Lan, M., Tan, C. L., & Low, H. (2006). Proposing a new term weighting scheme for text categorization. *National Conference on Artificial Intelligence*, 763–768.

5) Bakliwal, A., Arora, P., Madhappan, S., Kapre, N., Singh, M., Varma, V., 2012. Mining sentiments from tweets., *WASSA@ ACL, pp. 11–18.*

6) Medhat, W., Hassan, A., Korashy, H., 2014. Sentiment analysis algorithms and applications: *A survey. Ain Shams Engineering Journal 5, 1093 – 1113.*