Depression Detection using Sentiment Analysis

Pranjal G Mahajan¹ Tushar B Kute²

¹Cummins College of Engineering, Pune, India ²MITU Skillologies, Pune, India

Abstract— Twitter sentiment analysis is an application of sentiment analysis on data from Twitter (tweets), in order to extract sentiments conveyed by the user. Several studies carried out have shown the correlation between social media and depression. We aim at contributing to the research on depression detection using Sentiment analysis. We have pre-processed the data, applied feature extraction and feature selection. Thereafter, we measured the performance of two machine learning algorithms: Naive Bayes and Logistic Regression. The results of this study showed that Logistic Regression has outperformed Naive Bayes with accuracy 82.26%.

Keywords: Text Classification; Machine Learning; Logistic Regression; Social Media; Depression

I. INTRODUCTION

According to the World Health Organization (WHO), more than 300 million people worldwide are suffering from depression, which equals about 4.4% of the global population. Depression is also the number one cause of suicide, the second leading cause of death among adolescents and a difficult disease to treat, because those suffering from it are often reluctant to report. The level of negativity in the choice of words can help to get insights of what might be happening to the mental state of a person. This creates an opportunity to analyze social network data for the user’s feelings and sentiments to investigate their moods and attitudes when they are communicating via these online tools.

This paper uses a machine learning approach based on linguistic features to detect depression in Twitter users. In order to extract sentiment from tweets, sentiment analysis is used. The aim while performing sentiment analysis on tweets is basically to classify the tweets in different sentiment classes accurately. In this field of research, different methodologies have advanced, which propose strategies to train a model and then test to check its efficiency.

Sentiment analysis approaches can be broadly categorized into two classes – lexicon based and machine learning based. Lexicon based approach is unsupervised as it proposes to perform analysis using lexicons and a scoring method to evaluate opinions. Whereas machine learning approach involves use of feature extraction and training the model using a feature set and some dataset. In this paper, we have done analysis using machine learning based approach.

Performing sentiment analysis is challenging on Twitter data due to various reasons such as Limited tweet size with just 140 characters, use of slang and outdated words, the use of hashtags, user reference and URLs, using different language in between, using repeated words or symbols to convey an emotion. The basic steps for performing sentiment analysis using machine learning approaches includes data collection, preprocessing of data, feature extraction, selecting baseline features, sentiment detection and performing classification.

The remainder of the paper is organized as follows: Section 2 discusses the related works on this topic; Section 3 describes the dataset and methods used for data preprocessing, feature extraction and classification; Section 4 presents the results; Section 5 presents the discussions of the findings; and Section 6 draws the conclusion on the study as well as present some directions for possible future works.

II. RELATED WORK

Numerous research relating to the application of machine learning techniques for text classification has been carried out in various areas such as Finance, Safety and Economical problems to name a few. Tagging content or products using categories as a way to improve browsing or to identify related content on your website. Platforms such as E-commerce, news agencies, content curators, blogs, directories, and likes can use automated technologies to classify and tag content and products. News agencies use text classification for classifying various news based on topics and also for fake news detection. Text classification can also be used to automate Customer Relationship Management (CRM) tasks. It helps to gain insights about the popularity of a brand or new trends. The Automating the content tags on website and app can make user experience better and helps to standardize them. It can also be used for spam detection and intent detection.

III. METHODOLOGY

A binary task of classifying sentiment into positive and negative classes needs to be performed. The process methodology consists of the following steps - dataset description, pre-processing, feature extraction, model training.

Fig. 1: General Methodology for sentiment analysis

A. Dataset Description

The dataset used for depression detection is Sentiment140 dataset. The dataset is CSV with emoticons removed.

The dataset consisted of 16,00,000 tweets out of which 8,00,000 were labeled positive and 8,00,000 were labeled negative manually. A binary task of classifying sentiment into positive and negative classes needs to be performed.
The data file format has 6 fields -
0 - polarity of the tweet (0 = negative, 4 = positive)
1 - id of the tweet
2 - date of the tweet
3 - query.
4 - user that tweeted
5 - text of the tweet

The input is text of tweet field column and the output is polarity of the tweet column.

B. Preprocessing

The efficiency depends upon the pre-processing of data as pre-processing reduces ambiguity in feature extraction. Hence data pre-processing is an important step in Sentiment analysis.
Below are few steps used for preprocessing:
1) Converting All Text to Lowercase
In case we are using case sensitive analysis, we might take two or more occurrences of the same word. This might lead to mispredictions.
2) Remove All Punctuation Marks
Punctuation marks do not convey any sentiment and thus should be removed.
3) Remove All Numbers and Special Characters
Numbers and special characters also do not convey any sentiment. Numbers may cause an increase in the features. Also, they can be mixed with words. Hence their removal can help in associating two words which were otherwise considered different.
4) Remove Usernames and URLs
Usernames and URLs do not convey any sentiment and add to the features. Hence they should be removed.
5) replace three or more occurrences of a data with two occurrences (for example, convert coooooooooool to cool).

C. Feature Extraction

Feature extraction can be done by applying the Term Frequency Inverse Document Frequency (TF-IDF) using ngram Features- Unigram (1-word) and bigram (2-words) to find the weight of a particular feature in a text and hence filter the features having the maximum weight.

1) TfidfVectorizer
It is used to convert a collection of raw documents to a matrix of TF-IDF features. It performs the task of CountVectorizer followed by TfidfTransformer. It computes word counts, TF values and IDF values all at once and then gets Tfidf scores of a set of documents.

   - TF-IDF is a numerical statistic that reflects the value of a word for the whole document (here, tweet). The TF values, IDF values and the Tfidf score is calculated using the following formulas:

   TF(w) = (Number of times term w appears in a document) / (Total number of terms in the document)
   IDF(w) = log_e(Total number of documents / Number of documents with term w in it)
   Tfidf score = TF(w) * IDF(w)

   The max_df, min_df, max_ngram_range, stop_words are some parameters that can be passed to the TfidfVectorizer. When building the vocabulary to ignore terms that have a document frequency strictly higher than the given threshold (corpus-specific stop words), max_df feature is used. When building the vocabulary to ignore terms that have a document frequency strictly lower than the given threshold, min_df feature is used. The stop_words parameter can be used when we need to ignore the stopwords while building the vocabulary. It can be set to string (‘english’), list, or None (default=None).

2) n-gram features
When n-gram feature is applied, more than one word is considered at a time. This is advantageous as many time some words convey sentiments more effectively when used together. A n-gram of size 1 is referred to as a unigram; size 2 is a bigram (or, less commonly, a “digram”). N-grams almost always boosts accuracy. The N-gram frequency method provides an inexpensive and highly effective way of classifying documents.

D. Feature Selection

Feature selection is the method of reducing data dimension while doing predictive analysis. It is difficult and time-consuming to train data with large number of features. Hence, to reduce the number of features, we selected most important features that capture maximum information about dataset. Having irrelevant features in your data can decrease the accuracy of the models and make your model learn based on irrelevant features. The training time reduces significantly using feature selection.

E. Model Training

To train the model, Logistic Regression and Naive Bayes algorithms were used. These algorithms are described briefly as follows:

1) Logistic regression

Logistic regression is a simple classification algorithm. We try to predict the probability that it belongs to “0” class or “1” class. It uses a logistic function that always returns a value between 0 and 1 to model a binary dependent variable. The logistic function is given by:

   \[ f(x) = \frac{1}{1 + e^{-\beta x}} \]

   If the value is above the threshold, the class is predicted as “1” class else it is predicted as “0” class.

2) Gaussian Naive Bayes

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of conditional independence between every pair of features given the value of the class variable.

In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. The likelihood of the features is assumed to be Gaussian,

   \[ P(X|Y = c) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x - \mu)^2}{2\sigma^2}} \]

   hence, conditional probability is given by:
Other functions can be used to estimate the distribution of the data, but the Gaussian (or Normal distribution) is the easiest to work with because you only need to estimate the mean and the standard deviation from your training data. Gaussian Naive Bayes is used when there is processing of large data.

IV. RESULT

In this section we report the results of the experiments, i.e. the performance of the machine learning algorithms (Logistic Regression and Naive Bayes) when applied to the dataset:

A. Accuracy Score

To quantify the quality of predictions and interpret the performance, accuracy score is calculated for different algorithms and compared.

The table 4.1.a shows accuracy scores for Linear Regression classifier and Naive Bayes classifier using unigram and bigram features.

<table>
<thead>
<tr>
<th></th>
<th>Naive Bayes</th>
<th>Linear Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>unigram</td>
<td>77.5959375%</td>
<td>80.151299999999</td>
</tr>
<tr>
<td>bigram</td>
<td>80.4728125%</td>
<td>82.259375%</td>
</tr>
</tbody>
</table>

Table 4.1: (a) Accuracy scores

We consider unigram features as the baseline for all the observations. The accuracy score of Naive Bayes was 77.5959375% using unigram features and increased to 80.4728125% using bigram features. The accuracy score of Logistic Regression was 80.151299999999% using unigram features and increased to 82.259375% using bigram features.

B. Precision, Recall and F-measure

1) Using unigram features

The classification report of Naive Bayes is as follows:

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.77</td>
<td>0.79</td>
<td>0.78</td>
<td>159815</td>
</tr>
<tr>
<td>4</td>
<td>0.78</td>
<td>0.76</td>
<td>0.77</td>
<td>160185</td>
</tr>
<tr>
<td>accuracy</td>
<td>0.78</td>
<td>0.78</td>
<td>0.78</td>
<td>320000</td>
</tr>
<tr>
<td>weighted avg</td>
<td>0.78</td>
<td>0.78</td>
<td>0.78</td>
<td>320000</td>
</tr>
</tbody>
</table>

From the classification report of Naive Bayes, the f1-score for class 0(negative) is higher than that for class 4(positive).

The classification report of Logistic Regression is as follows:

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>159815</td>
</tr>
<tr>
<td>4</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>160185</td>
</tr>
<tr>
<td>accuracy</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>320000</td>
</tr>
<tr>
<td>weighted avg</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>320000</td>
</tr>
</tbody>
</table>

From the classification report of Logistic Regression, the f1-score is higher for class 0(negative) is the same as that for class 4(positive).

2) Using bigram features

Using bigram features, means considering two words at a time. Bigram features are the most commonly used n-gram features.

The classification report of Naive Bayes is as follows:

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.79</td>
<td>0.83</td>
<td>0.81</td>
<td>159815</td>
</tr>
<tr>
<td>4</td>
<td>0.82</td>
<td>0.78</td>
<td>0.80</td>
<td>160185</td>
</tr>
</tbody>
</table>

The classification report of LogisticRegression is as follows:

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.81</td>
<td>0.80</td>
<td>0.80</td>
<td>320000</td>
</tr>
<tr>
<td>4</td>
<td>0.81</td>
<td>0.80</td>
<td>0.80</td>
<td>320000</td>
</tr>
</tbody>
</table>

From the classification report of Naive Bayes, the f1-score for class 0(negative) is higher than that for class 4(positive).

The classification report of Logistic Regression is as follows:

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>159815</td>
</tr>
<tr>
<td>4</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>160185</td>
</tr>
<tr>
<td>accuracy</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>320000</td>
</tr>
<tr>
<td>weighted avg</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>320000</td>
</tr>
</tbody>
</table>

From the classification report of Logistic Regression, the f1-score is higher for class 0(negative) is the same as that for class 4(positive).

V. DISCUSSION

Adding bigrams to feature set improves the accuracy of text classification model. This is because features as the combination of words provides better significance rather than considering single words as features.

The accuracy and f1-score is higher for Logistic Regression. Hence, Logistic Regression is a better algorithm to predict compared to Naive Bayes considering accuracy.

The time required is more in case of Logistic Regression classifier. Naive Bayes classifier performs better compared to Logistic Regression as it has a higher bias but lower variance compared to Logistic Regression. Training SVM classifiers, Decision tree classifiers and Random Forest classifiers requires huge amount of time due to the complex nature of their working. This makes these algorithms unfit to use for predictions.

VI. CONCLUSION

Twitter sentiment analysis focuses on analyzing the sentiments of the tweets and feeding the data to a machine learning model in order to train it and then check its accuracy, so that we can use this model for future use according to the results. It comprises of steps like data collection, text pre-processing, feature extraction, feature selection, training and testing them model. In this paper, we have considered
unigram features as baseline and trained and tested two machine learning algorithms (Linear Regression and Naive Bayes). The study has shown that Logistic Regression has better performance than other algorithms in terms of accuracy and training time.

Twitter sentiment analysis is difficult to implement and has lower accuracy due to slang used, spelling mistakes, use of outdated words and the short forms of words in the tweets. Many analyzers don’t perform well when the number of classes are increased. We can further extend this study and improve the accuracy using various boosting and bagging algorithms.

REFERENCES


