Stephen Williams

Abstract

For this research, I did a sentiment analysis of march 16 tweets relating to the Russian/Ukraine war. This is a classification task and before doing any modelling, I had to clean the dataset and attach a label column that observes the sentiment of a tweet. I then converted the tweets into vectors using a word2vec model that I trained. Using this tweet vector dataset, I train and tested a logistic regression and decision tree model. After evaluation these two classifiers, the decision tree did the best on the tweet vector set. It was able to classify the sentiment of tweets better then logistic regression model

Introduction

For my final project, I wanted to continue where I left off in my previous project and conduct sentiment analysis on tweets regarding the Russian/Ukraine war. Sentiment analysis is obtaining opinions or stances from a text or speech. In my case, the sentiment/labels attached to each tweet will be either positive, neutral or negative. Sentiment analysis has been done on tweets written in Arabic on various themes such as art and politics using corpus and lexicon based tools [1]. Another paper has trained question classifier to label questions using a word2vec feature set [2]. Some of these researchers manually label their tweets, but I used a labelling package to do the hard work for simplicity's sake.

I will be using word2vec to transform the tweets into vectors. My task will be classification, where I will use logistic regression and decision tree to train and test this tweet vector dataset to classify the tweets. From there, I will evaluate each model on how good they were at detecting the sentiment in each tweet using various metrics.

Dataset

Firstly, I chose march 16 tweets relating to the Russian/Ukraine war as it was one of my previously selected datasets [3]. It has eighteen features, but I will only use the text column from this dataset. Furthermore, I will be only analyzing tweets that are in English. There are about two hundred thousand tweets that are in English.

Because this dataset did not have a label column that observes the tweet's sentiment, I used vader sentiment analysis module to do so. Vader seemed to be a good sentiment analysis tool as it is specifically tuned to detect opinions within social media, including Twitter [4]. Most tweets within my dataset are not written formally and users tweet with emojis and slang. Hence, it was a good idea to use vader.

Vader outputs a compound metric. It is recommended, that if this compound number is at least greater than .05, then this tweet is positive. If it is less than -.05, it would be negative. Otherwise it is labeled neutral.

After labeling the tweets using Vader, I observed around one hundred thousand tweets labeled negative, seventy thousand positive, and forty thousand neutral. The new dataset has two columns, text and labels.

I had to do a lot of preprocessing. I cleaned this tweet dataset of punctuations,@, urls, and stop words. I also used lemmatizer to normalize the words used. For example, "cleaning" and "clean" are similar sentiments. Therefore, lemmatizer would reduced "cleaning" to "clean" for simplicity.

Method

Firstly, I transformed the “text” feature of my preprocessed data into vectors. I used word2vec, where I represented the words in each tweet as vectors. Word2vec is a neural network that has three layers. Input, hidden and output layer. It takes in words and transforms them into vectors. My motivation for choosing word2vec was that I found it incredible that you can numerically represent words as arrays and compare words using math and graphs. Wicked!

There are two models that can be used to achieve this word vector data set: Skip-gram and continuous bag of words [5]. I used skip-gram, which predicts the context of a given the word. I used the genism word2vec module to create and train my word2vec using skip-gram model.

To use genism, I had to tokenize the tweets into a list of lists. Each tweet is represented as an array with a nested array comprised of split words. I also had to set my parameters where I put my vector dimension size as 200.

After getting my word2vec model , I had to solve an issue. I am trying to analyze tweets and not the individual words within the tweets. Therefore, I had to represent the tweets as vectors. I took the average of the word vectors within the tweet to achieve this [6]. As of result, I created a new dataset set of tweet vectors. My previous "label" column in the preprocessed data, was then attached to this transformed data.

I used two classifiers. First is the logistic regression which predicts binary and multiclass labels. Logistic regression uses a logit function which is the log of odds. The second classifier is decision tree which is great as it is easy to understand. From a target, it splits into branches of the outcomes. These classifiers were trained and tested on my tweet vector set.

Results

model	Training score	Testing score	Precision score	Recall score	F1Score
Logistic regression	.792	.792	.781	.753	.765
Decision tree	.995	.925	.919	.917	.918

Table shows various metrics used to evaluate two models. All values are rounded to the nearest thousandth. All models were trained and tested on tweet vector set.

	Precision	recall	F1-score	support
negative	.94	.95	.94	22929
neutral	.9	.9	.9	8139
positive	.92	.91	.91	14628
accuracy			.93	45696
Macro avg	.92	.92	.92	45696
Weighted avg	.93	.93	.93	45696

Classification report for decision tree

	precision	recall	F1-score	support
negative	.81	.86	.83	22929
neutral	.76	.63	.69	8139
positive	.77	.76	.76	14628
accuracy			.79	45696
Macro avg	.78	.75	.76	45696
Weighted avg	.79	.79	.79	45696

Classification report for logistic regression

Conclusion

To summarize, I first created a word2vec model from the text column of the original dataset. Using this word2vec model, I made a tweet vector dataset by taking the average of the word vectors present in a given tweet. The label column created using vader sentiment was attached to this tweet vector set. I then used this transformed dataset to train and test the logistic regression and decision tree.

Looking at the results, I can say that the decision tree did much better than the logistic regression. For every metric used, the decision tree came out on top. I thought that at least logistic regression would be comparable to a decision tree when working with continuous data.

Future work

I could have also used more features such as hashtags to predict the sentiment. There was a clear class imbalance that I could have oversampled the minority labels by using smote or another technique. I would also like to change the algorithm used to create the word2vec model. I would use continuous bag of words to see if the results of the two classifier models results are different. Next would be to use more models, like random forest, to train and test the tweet vector dataset. And the last to do is optimize the hyperparameters of the logistic regression and decision tree using grid search.

References

[1] Abdulla, N.A., Ahmed, N.A., Shehab, M.A., & Al-Ayyoub, M. (2013). Arabic sentiment analysis: Lexicon-based and corpus-based. 2013 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), 1-6.

[2] Razzaghnoori, M., Sajedi, H., & Jazani, I.K. (2018). Question classification in Persian using word vectors and frequencies. Cognitive Systems Research, 47, 16-27.

[3] https://www.kaggle.com/datasets/bwandowando/ukraine-russian-crisis-twitter-dataset-1-2-m-rows

[4] Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

[5] Johnson, Daniel. “Word Embedding Tutorial: Word2vec Model Gensim Example.” *Guru99*, 14 May 2022, https://www.guru99.com/word-embedding-word2vec.html#6.

[6] https://www.kaggle.com/code/nitin194/twitter-sentiment-analysis-word2vec-doc2vec/notebook