Project Report

Group 6: Chen Xiangyun, Minwei Cheong, Karen Tee Lin

Abstract

The prevalence of social media as the "collective wisdom" of the society has led to increased interest on sentiment analysis on social media texts. This paper aims to explore the best classification approach for sentiment analysis in social media texts. Supervised machine learning models including Naive Bayes and Support Vector Machine will be used to classify the sentiments and their effectiveness will be compared and evaluated. These approaches are of interest of this paper as they cover the most popular approaches that are widely adopted globally, allowing our results to be compared with prior literatures. Different feature engineering will be conducted using Count Vector, TF-IDF Vectors of Word level, N-Gram level and Character level, Word Embeddings and Text/NLP based.

Introduction and related research

Social media is changing the public discourse in society and influencing people's opinions. The topics discussed on social media range from daily life, politics, entertainments to sciences, hence can be constructed as a form of "collective wisdom" (Asghar, Khan, Ahmad, & Kundi, 2014). The most popular platforms include Facebook, Twitter, Whatsapp, Instagram and Youtube.

In Twitter, users can post tweets, or instant messages of 140 characters or less, which can include hyperlinks to websites, photos, videos, forms, and the like. Tweets are presented in a time-stamped record called a timeline (Aladwani, 2015). Due to this characteristic, users tend to express their real feelings freely in Twitter. Therefore, it is an ideal source for understand people's sentiment towards various topics.

The rise of social media has led to increased interest on sentiment analysis on social media. Globally, two techniques are often used: Supervised Machine-Learning and Unsupervised methods, that use a lexicon with words scored for polarity values such as neutral, positive or negative (Neri, Aliprandi, Capeci, Cuadros, & By, 2012). Chaovalit and Zhou (2005) concluded that supervised techniques perform with a higher accuracy than unsupervised methods. However, the shortcoming for supervised techniques is the high requirement on training data. Unsupervised technique has the advantage of robustness and easy deployment.

Besides the classification methods, the quality and usability of the feature extraction are in the interest of this paper as well. First of all, feature extraction can be applied to different levels of text: characters, words, phrases, sentences, paragraphs or documents. Words, as the smallest units, according to Neri, Aliprandi, Capeci, Cuadros, and By (2012), can have different polarities in different meanings and or in different domains. Furthermore, polarity expressed by a word may be reversed within a phrase through negation. Hence, careful transformation of raw text data into feature vectors is required.

Therefore, using the aforementioned performance measurement, the aim of this project is to investigate the performance of a number of sentiment classifiers on data from social media by the learning-based approaches of Naive Bayes (NB) and Support Vector Machine (SVM). Different feature engineering will be conducted using Count Vector, TF-IDF Vectors of Word level, N-Gram level and Character level, Word Embeddings and Text/NLP based.

Research question and method description

- **Research question**: What is the best classification approach for sentiment analysis in social media texts? Which feature engineering methods will produce the best result for sentiment analysis using NB and SVM?
- **Hypothesis**: SVM is the best classification approach for sentiment analysis in social media texts as it scales relatively well to high-dimensional data. It also captures the dependency between different features unlike Naive Bayes.
  Word Level TF-IDF will provide the best results since it penalises commonly occuring words and gives heavier weightage to rarer words which is crucial for determining the sentiment of the tweet.
- **Programming Framework**: Jupyter notebook. The programming language of choice is Python because of its wide adoption in the software development industry and the scientific world, as a result the language has well-supported libraries providing NLP tool (Li, 2019).
- **The data**: We obtain the dataset of 1.6 million tweets from kaggle Sentiment140[1]. 10,000 is sampled randomly and used for training. We pre-process by removing noise data and including manual labelling (Positive Negative Neutral). We then use 1,000,000 tweets for testing.
- **Classifiers**: We use NB and SVM, and thereby optimising performance using GridSearch to tune the hyperparameters. The significance of choosing these classifiers are as follows:
  - NB: It assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. It is a basic supervised machine learning

[1] Data source from https://www.kaggle.com/kazanova/sentiment140

approach. It is easy and fast to predict class of test data set. It performs well in the case of categorical input variables, which is applicable in our polarity analysis.
    - SVM: It is another supervised machine learning algorithm. The model extracts a best possible hyper-plane / line that segregates the two classes. It is an effective classifier for classification challenges, making it a suitable choice for our polarity analysis.
- **Feature Extraction**
    - Count Vectors: We represent the dataset as a matrix and every row represents a review. The columns are different words from the review, and the cell represents the frequency count for that particular word.
    - Term Frequency - Inverse Document Frequency (TF-IDF): the relative importance of a term in the document is represented by TF-IDF score. We use Word Level, N-gram Level and Character level where the TF-IDF scores represent the importance of every term, combination of N terms and every character respectively.
    - Word Embeddings: words and documents are represented using a dense vector. The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used.

Research results

The following shows the results we have obtained. To measure the performance of the classifiers, we will calculate their accuracy. More specifically, we will use F1 score, which measures accuracy using recall and precision. Precision is the ratio of true positives to all predicted positives. Recall is the ratio of true positives to all actual positives. F1 score is the weighted average of precision and recall, hence it provides a more balanced view compared to using either precision or recall. We will also use Area under curve (AUC), which represents measure of separability. It tells us how much the model is capable of distinguishing the classes. The higher the AUC, the better the model is at separating the different classes. Throughout the implementation of the project, we tried to improve the accuracy through various methods. More specifically, we had to change the parameters of SVM to optimize it.

From what we observed, among the various feature extraction methods we used, word-level TF-IDF performed the best using Naive Bayes. This is shown in having the highest accuracy, F1 score and AUC when Naive Bayes is used. Word embeddings seemed to have the worst performance, with an accuracy of close to 0.5 which is equivalent of being random.

```
Epoch 1/5
 - 20738s - loss: 1.1232 - acc: 0.4787
Epoch 2/5
 - 28315s - loss: 0.7264 - acc: 0.4964
Epoch 3/5
 - 43382s - loss: 0.7044 - acc: 0.4950
Epoch 4/5
 - 11835s - loss: 0.6998 - acc: 0.5046
Epoch 5/5
 - 7630s - loss: 0.6993 - acc: 0.4947
```

General Accuracy:

	Count Vectors	Word Level TF-IDF	N-gram Level TF-IDF	Character Level TF-IDF	Word Embeddings
NB	0.732367	0.738448	0.6651	0.710357	0.514959
SVM	0.711067	0.688683	0.659411	0.712113	

F1 Score:

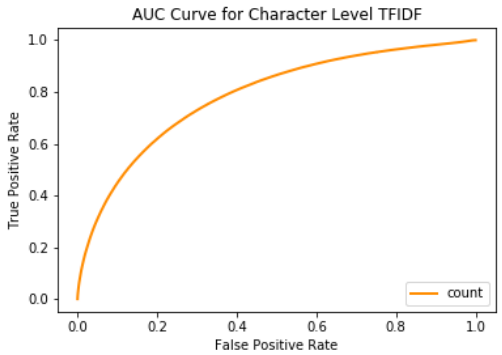
	Count Vectors	Word Level TF-IDF	N-gram Level TF-IDF	Character Level TF-IDF	Word Embeddings
NB	0.706706	0.729965	0.643190	0.690804	0.550102
SVM	0.728117	0.656599	0.680402	0.711616	

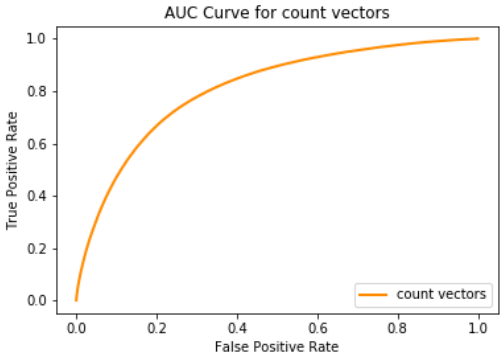
AUC:

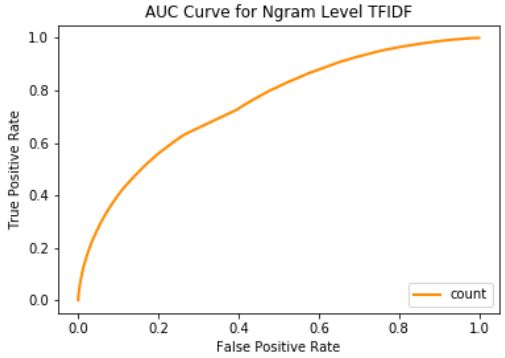
	Count Vectors	Word Level TF-IDF	N-gram Level TF-IDF	Character Level TF-IDF	Word Embeddings
NB	0.808662	0.819986	0.749967	0.784497	0.526121
SVM	0.786111	0.757449	0.713998	0.783925	

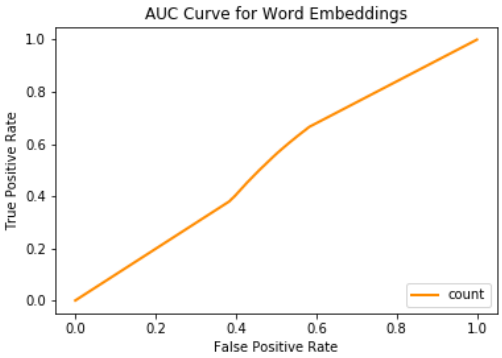
AUC plots for Naive Bayes and SVM

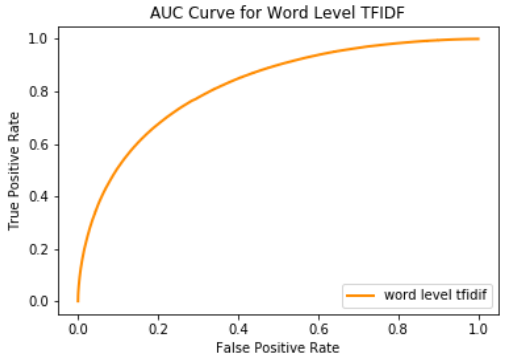
Naive Bayes





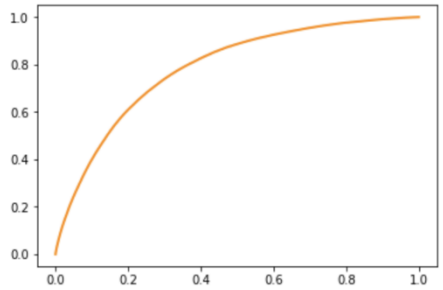




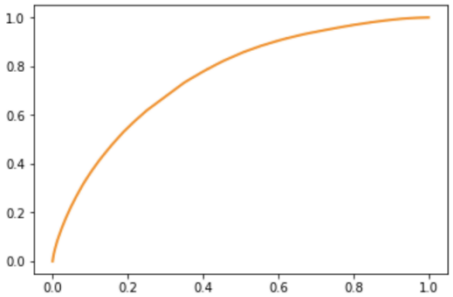


SVM

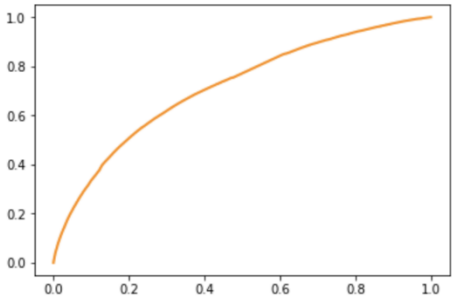
a) Count Vector



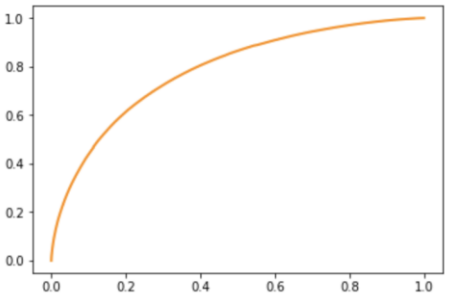
b) Word Level TF-IDF



c) N-gram Level TF-IDF



d) Character Level TF-IDF



The plots above are the AUC-ROC curve, the area under the ROC curve. The higher the AUC, the better the model is at predicting. The ROC curve is plotted with the true positive rate against the false positive rate. As we can see from the plots above, the plots for SVM generally seem to be smoother than Naive Bayes. The graph of word embeddings is almost linear which explains its bad performance. But overall, the graph with the steepest growth is word-level TF-IDF using Naive Bayes. The true positive rate increases much faster than the false positive rate.

Discussion and reflection on the project

Summary of the study

Over the last few years, Twitter has become the go to source of information about unfolding news events that have not yet been properly picked up by news agencies and reported, or where news reporting is vague and unreliable for some reason. In our study, we will focus on five levels of feature engineering - namely, count vectors, the three TF-IDF levels as well as word embeddings, on the twitter 1.6 million dataset. Using Naive Bayes, SVM methods, we will be testing the hypothesis that count vectors will be the most important feature when it comes to predicting the sentiment of the comment.

Key Issues around the Data and Algorithms

We use naive bayes and support vector machine on the five different types of feature extraction methods. Instead of using all words, we use max_features to tokenize the sample. In this case, we perform dimension reduction by stating a maximum number of words that can be chosen, effectively reducing the model complexity and size. For both Naive Bayes and SVMs, GridsearchCV is used to find the optimal parameters for each feature extraction method, and this yields stellar performances for the SVM algorithm in Count Vectors and also the three TF-IDF methods.

Among the various feature extraction vectorizers, we conclude that our hypothesis on word level TF-IDF is accurate, since word embeddings had the worst performance while word-level TF-IDF had the best performance. This might have been attributed to the fact that the words that have appeared less frequently contributed more significantly to weighing the model. Word importance will be increased if the number of occurrence within same document (i.e. training record). On the other hand, it will be decreased if it occurs in corpus (i.e. other training records).

However, our hypothesis on SVM did better than SVM is not accurate based on the results. Overall, NB did better than SVM in terms of f1 score, and this could be due to the results being almost linearly separable, and also independent word occurence.

Counting word vectors did similarly well, a close second to word-level TFIDF. The reason behind using this approach is the counter-logic of TFIDF. In count vectors, the philosophy is that the keyword or important signal will occur again and again. So if the number of occurrence represent the importance of word, the greater the frequency, the more important it is.

After some eyeballing, we picked up that a possible reason could be that common words (words which will appear in multiple comments) are helpful in distinguishing between classes. Function words like pronouns are very common and would be down weighted in TF-IDF, but given equal weight to rare words in countvectorizer, which could have attributed to countvectorizer having a relatively good classification score.

Word Embeddings performed poorly for both Naive Bayes and Logistic regression. We believe that if it had been used for a neural network as a pre-processing layer, the results would have been much better. This is since Recurrent Neural Networks (RNN) and Long Short-Term Memory networks introduce a memory into the model. Having a memory in a network is useful because, when dealing with sequenced data such as text, the meaning of a word depends on the context of the previous text. However, due to logistical constraints, we were unable to produce results in time for the RNN model.

Future Research

Sentiment analysis using unsupervised learning methods like RNN and CNN can be conducted to compare the effectiveness of unsupervised and supervised learning methods. With such advances in sentiment analysis, it provides a new research direction since understanding consumers' opinions helps corporations make better decisions. With the rise of social media, there is an exponential growth in the amount of data and opinions online. Thus, sentiment analysis comes in useful in analysing opinions in all forms of platforms online and corporations see the commercial potential in it.

Contribution and Project Assessment

Our team has an amicable and effective working relationship. The team always brainstormed together to ensure we have a common goal. After this the team would

allocate the work to each member effectively. Karen mainly runs the training of the models. Minwei conducts the research for our classification approach and runs the training of the models as well. Xiangyun runs the dataset preparation and feature engineering of the project, as well as reporting the research and the results that the team has obtained.

The project generally worked well, while some possible areas of improvement include time allocation for model training. It is a challenging yet fulfilling learning experience for the team where the team is encouraged to try out different feature engineering and classification approaches. The commitment of the team on this project is evident, in terms of the proactiveness in conducting research, clarifying doubts, and relentlessly training the models. The team functions well and the contribution of the teammates are in accordance to the plan. Overall, this is an effective and collective group effort.

Conclusion

In a nutshell, among the various feature extraction vectorizers, we conclude that our hypothesis on word level TF-IDF is accurate. However, NB is a better classification approach for social media text analysis than SVM. The TF-IDF vectorizer performs similarly to that of the Count Word Vectorizer on a word level, but on a sub-word level (N-gram, and character) more work needs to be done. In future studies, it would be helpful to consider using other dimension reduction methods such as semantic compression or Singular Value Decomposition to sieve out groups of keywords, to further optimize analysis. And as a bonus, it becomes much easier to visualize the data with these much fewer dimensions, which means it is much easier for our human eyes to understand the relationship among the documents. Once we will have reduced the dimensionality, the data can be fit in other classification methods such as K-means Clustering algorithm to group the documents based on the distance among the documents which are calculated based on the reduced dimensions. Different kinds of data could also be explored, such as audio or visual data.

References

Asghar, M. Z., Khan, A., Ahmad, S., & Kundi, F. M. (2014). Predicting the Future With Social Media. *Journal of Basic and Applied Science Research*, 181-186.

Chaovalit, P., & Zhou, L. (2005). Movie review mining: A comparison between supervised and unsupervised classification approaches. *Proceedings of the Hawaii International Conference on System Sciences*.

Li, H. (2019). Pproject Specification: Sentiment Classification of Social Media v.1.0.

M.Aladwani, A. (2015). Facilitators, characteristics, and impacts of Twitter use: Theoretical analysis and empirical illustration. *International Journal of Information Management*, 15-25.

Neri, F., Aliprandi, C., Capeci, F., Cuadros, M., & By, T. (2012). Sentiment Analysis on Social Media . *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 919-926.