FINAL REPORT

Sentiment Analysis on Tweets

Computer Science & Engineering

December 9, 2022

COEN 281: Pattern Recognition and Data Mining
Fall 2022 Project
Professor Manish Marwah

Team RNG

Shujia Liang
Guojian Wang
Andrew Zhang

TABLE OF CONTENTS

1. INTRODUCTION

a. Motivation

Sentiment Analysis is one of the most widely studied applications of Natural Language Processing and Machine Learning. The internet has provided a platform for people to express their views, emotions and sentiments towards products, people and life in general.
Twitter is a major platform of public marketing, communication and information distribution of blockchain projects; yet, as a customer or investor, gathering the sentiment of the community is very crucial on their investment decision: if the community loves the project, it will be a great blue-chip investment option. Given the recent crypto turbulence and major fraud of FTX, sentiments of a community also reveals the confidence level of the users to a project. In this time of year, a positive sentiment can be a very important indicator on the decision making process.
The goal of Sentiment Analysis on Tweets is to harness this data in order to obtain important information regarding public opinion that would help make better business decisions and better product consumption.

b. Problem statement

We aim to implement different machine learning models to perform sentiment analysis on a dataset of tweets. We will be focusing on predicting sentiment from comments of Twitter users in the data set. Our models will parse the comment text of the input dataset and classify each data point into two categories:

-Score 0 for comment with negative sentiment

-Score 4 for comment with positive sentiment

And comparing the performance of these models to find our best model that can be used as tools to learn sentiments of public opinions on a tweet project.

2. RELATED WORK AND CONTRIBUTIONS

A similar work describing the methodologies are from Go, Bhayani, and Huang. They have introduced an approach for automatically classifying the sentiment of Twitter messages. They have used different types of distant supervised learning algorithms (Naive Bayes, Maximum Entropy, and SVM) and have 80%+ accuracy when trained with emoticon data.

Another work from Rathi et al. is using ensemble machine learning techniques, merging SVM with Decision Tree and experimental results.

3. DATA SET

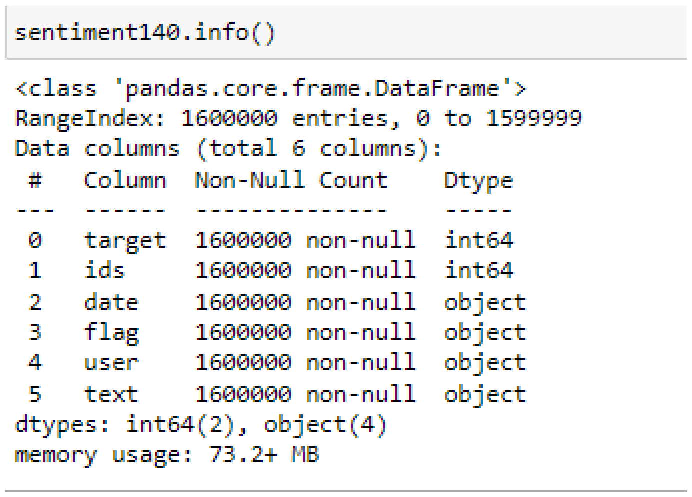
The data set we use to train our model is Sentiment 140 dataset with 1.6 million tweets. Processing the data set with 160W data points is going to be a huge occupation of run time and memory therefore we extract 4W data points(2W data points for each target label) as the dataset that we will be using in this project.
There are six columns present in the dataset:

- Target: the polarity of the tweet (negative, positive)
- Ids: unique id of the tweet
- Date: the date of the tweet
- Flag: refers to the query. If no such query exists then the flag is NO QUERY
- User: refers to the name of the tweeted user
- Text: refers to the text content of the tweet

4. DATA PREPROCESSING

a. Clean Data

The dataset is clean so that there are no NA's or entries falling out of each category.



```
sentiment140.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1600000 entries, 0 to 1599999
Data columns (total 6 columns):
 #   Column  Non-Null Count    Dtype
---  ------  --------------    -----
 0   target  1600000 non-null  int64
 1   ids     1600000 non-null  int64
 2   date    1600000 non-null  object
 3   flag    1600000 non-null  object
 4   user    1600000 non-null  object
 5   text    1600000 non-null  object
dtypes: int64(2), object(4)
memory usage: 73.2+ MB
```

Not all the columns in this dataset carry useful information that is needed for this project to train our model that performs sentiment analysis . In fact, only the 'Text' and 'Target'

columns are the main columns that we want to keep and explore so we will extract these three columns from the dataset for training.
Moreover, we split the dataset for training and testing purpose:

Training set: 3W data points
Testing set: 1W data points

b. Tokenization

In order to determine features for the training object, contents in the ‘Text’ column should be filtered into tokens that are semantically meaningful. To do this we used the python package called NLTK which contains several tokenizers, including one for tweets. We use that tokenizer and in addition we do the following:

- remove stopwords
- make all tokens lower case
- removing twitter handles
- remove punctuations, http links

And we "lemmatize" the tokens. That means we convert different forms of a word to a common basic form, so that they can be recognized as the same word.

c. Feature Extraction

We targeted all the unique words in the “Text” column of our training dataset to be features to train our models.
There are a total of 25,535 unique words in the training dataset. We used them to form a bag of word representation in which the frequency of each word for each data point is stored. Hence a feature matrix was populated for both training and testing.
Shape of the Training Matrix: (30000, 25535)
Shape of the Testing Matrix: (10000,25535)

5. MODELS

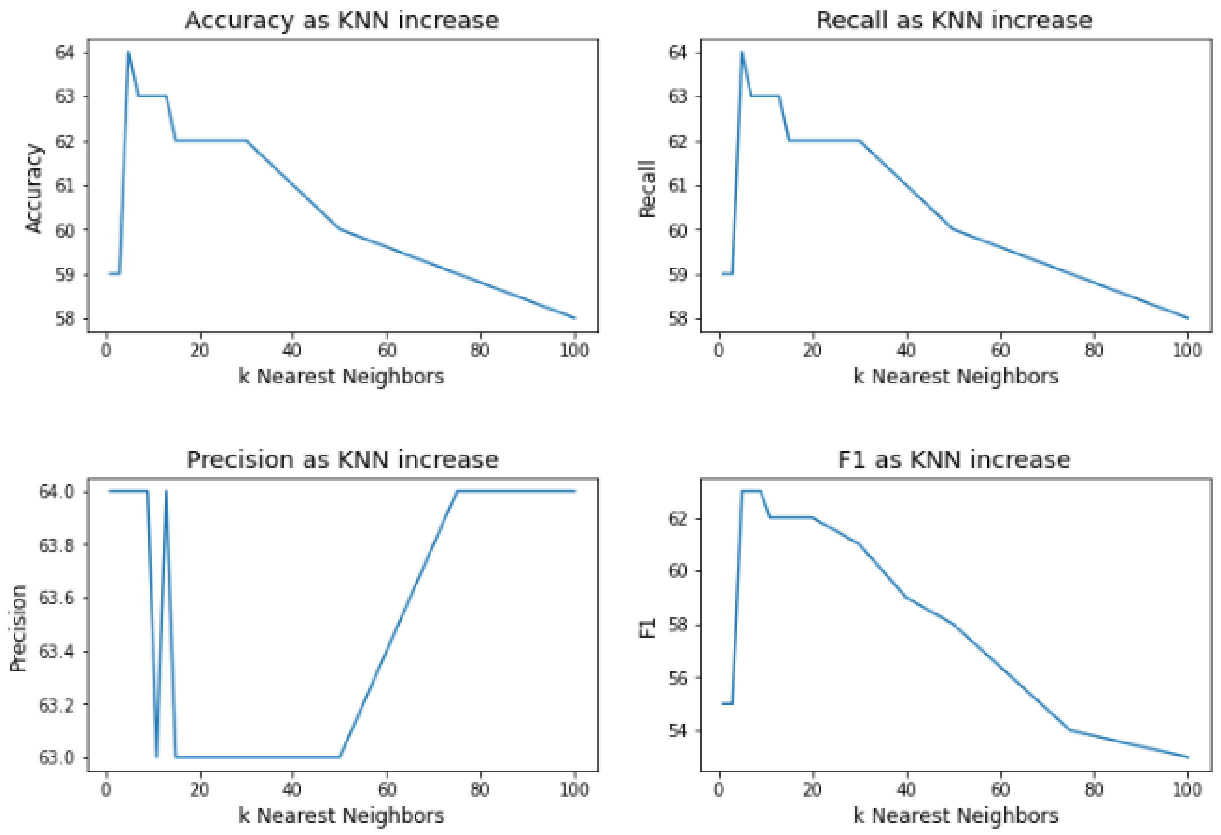
a. Base model - K nearest neighbor

i. <u>Determine Hyperparameter - K</u>

We initially determined to test the performance of our KNN model with a list of K choices with value showed below:

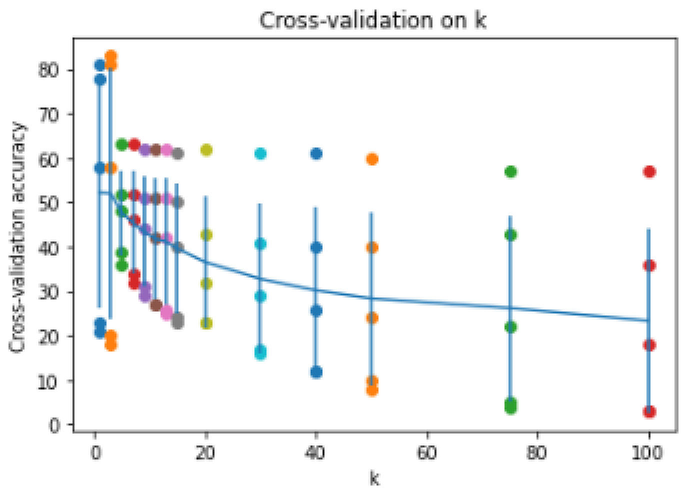
```
k_choices=[1, 3, 5, 7, 9, 11, 13, 15, 20, 30, 40, 50, 75, 100]
```

Without cross validation to determine the best K choice, we tested the performance of our KNN classifier with each K value.



As the above figure shows, the best performance comes from K values ranging into 1-11. The best accuracy, recall, precision, and f1 scores are 0.64, 0.64, 0.64, and 0.63. However, these statistics cannot represent the overall performance of each K value we use and are not steady.

Thus we need to use cross validation to determine the best K. We performed 5-folds cross validation on the dataset and visualized the performance statistics.



The figure above was created by plotting the K values with the average and standard deviation of the accuracies of each K. From this figure we can determine that the best K value is 1 since it has the best average performance and less standard deviation.

ii. <u>Results</u>

Our final prediction with the KNN model using the best K =1 generated the performance report as below:

```
Confusion Matrix for k = 1 is:

[[1508 3492]
 [ 607 4393]]

Classification Report for k = 1 is:

              precision    recall  f1-score   support

           0       0.71      0.30      0.42      5000
           4       0.56      0.88      0.68      5000

    accuracy                           0.59     10000
   macro avg       0.64      0.59      0.55     10000
weighted avg       0.64      0.59      0.55     10000


             CPU times: total: 7min 27s
```

With 1W data points, we got:

TP cases: 1508
FN cases: 3492
FP cases: 607

TN cases: 4393
Accuracy: 59%
Precision: 64%
Recall: 59%

b. Bernoulli Naive Bayes

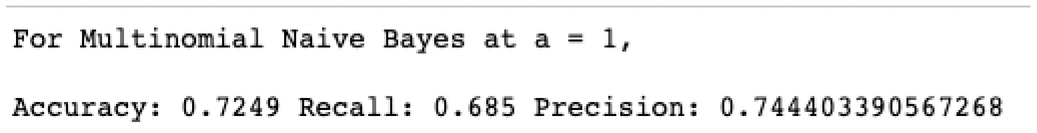
We have applied BernoulliNB of scikitLearn, and tuned the smoothing parameter Laplacian alpha, as we only use 3% of the original dataset and we want to avoid the zero-probability case. We have tried alpha = 0, 0.5, 1, 1.5, and 10, and the results of accuracy, precision, and recall are shown below:

Alpha	Accuracy	Recall	Precision
0	73%	72%	74%
0.5	73%	71%	73%
1	73%	72%	74%
1.5	73%	72%	74%
10	74%	75%	73%

And the results have shown that applying Laplacian smoothing will not affect the results too much, and we follow the convention of alpha = 1 on our final performance evaluation.

c. Multinomial Naive Bayes

We have applied the MultinomialNB of the scikitLearn package, and applied the same alpha = 1 as above to keep the consistency. We have fit the same training set and do the testing with results shown below:



```
For Multinomial Naive Bayes at a = 1,

Accuracy: 0.7249 Recall: 0.685 Precision: 0.744403390567268
```

6. PERFORMANCE EVALUATION

Comparing the accuracy and precision of the prediction, BNB performed slightly better than MNB, and both of them are 14% better than the baseline model (KNN at k = 1); the recall of BNB is significantly better than the MNB (6%, comparing to the ~1% difference they have over accuracy and precision).

There are a couple of reasons that BNB is good for this problem specifically. Since it records the presence and absence of one label, It is not good at handling long passages. For instance, it might address the entire book in the "China" section as the word "China" occurs once in the book. But since we are only analyzing meaningful words in a 140-character tweet, the number of duplicated words in the same sentence is tiny.

Also, BNB is not good at dealing with multidimensional data, but since there is no high dimensional features we need to deal with in this work, we are able to minimize the disadvantages of BNB.

Since this problem is purely binary sentiment analysis, and the dataset is pretty large, those two advantages of BNB can be applied to this problem.

Compared to the BNB, multinomial naive bayes (MNB) count the number of occurrences of each instance. As we stated above, the number of duplicated words in the same sentence is tiny, so that in the MNB model, the occurrences are 0 and 1 for the most of the time. Thus, the accuracy and precision of MNB and BNB are very close to each other.

By definition, recall = true positive / (true positive + false negative), and the relatively larger performance difference between BNB and MNB in this model are more likely due to the over-rejections in the MNB. As MNB counts the occurrences, the false negative occurs as the predicted value is smaller than the actual value. For instance, one person can use unequal numbers of words in both positive and negative ways in the same tweet, and counting occurrences will justify it differently from counting the presence.

7. LEARNING

This project exposed us to a variety of data preprocessing techniques and machine learning algorithms that can be used for natural language processing. We learnt that the preprocessing of data can lead to a great improvement in the performance of the model. Having strong data preprocessing skills is necessary in a machine learning project.

Moreover, feature determination is critical to a machine learning project. Having advanced and powerful algorithms to extract features is an important part for model optimization.

8. CONCLUSION AND FUTURE STEPS

Based on the performance analysis of the three models performed in this project, we can say that the BNB model is our best tool to learn about the sentiments of public opinions on a tweet project; the performance of our models did not meet our expectations. We conclude that this is due to limitations of the algorithms we used to select features. Our current features selection technique is simple and not powerful enough to represent the actual attributes of a dataset and we might be having an issue of overfitting with our current result.
Our next step should be trying to optimize the performance of our project.There are a number of resources and tools on the internet that we can further explore to find solutions to this problem and optimize the performance of our model. The TF-IDF method could be a good example of our next approach to optimization. Additionally, in our future works, we should try to perform more advanced machine learning models on the dataset and combine with the three models used in this project to perform a profound performance analysis and get a better understanding of the attributes of each model.

9. Division OF LABOR

Team Member(s)	Tasks Assigned
Guojian Wang	Data preprocessing(clean and tokenize data)
Andrew Zhang	Feature extraction. Cross Validation. KNN implementation and performance test.
Shujia Liang	Bernoulli and Multinomial Naive Bayes implementation and performance test. Overall performance analysis.

Github Repository:
https://github.com/W1407134/Sentiment-Analysis-on-Tweets

10. REFERENCES

1. Alec Go, Richa Bhayani, Lei Huang, Twitter Sentiment Classification Using Distant Supervision. CS224 2009 Final Project, Stanford University.
2. M. Rathi, A. Malik, D. Varshney, R. Sharma and S. Mendiratta, "Sentiment Analysis of Tweets Using Machine Learning Approach," in 2018 Eleventh International Conference on Contemporary Computing (IC3), Noida, India, 2018 pp. 1-3.
3. Marios Michailidis, Social Machine Learning with H2O, Twitter, and Python. https://www.linkedin.com/pulse/social-machine-learning-h2o-twitter-python-marios-michailidis/
4. Dataset reference: https://www.kaggle.com/datasets/kazanova/sentiment140