Udacity Machine Learning Nanodegree Capstone

Quora Duplicate Question Detection

Nujood Ahmed

January 9, 2019

1. Definition

1.1. Project Overview

Quora is a place to gain and share knowledge—about anything. It's a platform to ask questions and connect with people who contribute unique insights and quality answers. This empowers people to learn from each other and to better understand the world.

Over 100 million people visit Quora every month, so it's no surprise that many people ask similarly worded questions. Multiple questions with the same intent can cause seekers to spend more time finding the best answer to their question, and make writers feel they need to answer multiple versions of the same question. Quora values canonical questions because they provide a better experience to active seekers and writers, and offer more value to both of these groups in the long term.

The main goal of this project is to build a high-quality knowledge base, it's important that we ensure each unique question exists on Quora only once. Writers shouldn't have to write the same answer to multiple versions of the same question, and readers should be able to find a single canonical page with the question they're looking for.[2]

1.2. Problem Statement

Most of the users don't check if their question has already been asked by someone else, which will lead to duplicate questions posts. This necessitates having a prediction system that can check if a question is duplicated. Before the user can post a question, the system will suggest an existing one.

So, the problem can stated as follow: predict whether a pair of questions are duplicates or not.

For example, given the following two questions:

Q1: What is the easiest way to get a job in the USA?

Q2: What is the easiest way to get a job in Saudi Arabia?

The system will label them as either Duplicate or Non-duplicate, this problem will be tackled as a Natural Language Processing problem.

1.3. Metrics

I used two kind of metrics to evaluate my model:

1- **Accuracy**:

$$accuracy = \frac{truepositives + truenegatives}{totalsmaples}$$

This accuracy will be measured on the test set (10% of the total dataset).

1. False negatives, where a duplicate pair is not detected as such, will lead to degraded user experiences. However, it is also rather harmless, since no information is lost. There is still an opportunity to manually flag them as duplicates, or build such a feature on the platform that is using the duplicate detector.

2. False positives, where a question pair that's not a duplicate, but is marked as such, is more dangerous. It can lead to very degraded user experience, since it directly hinders the user trying to perform an action. It is better for the system to err on the side of false negatives than false positives.

2. Log loss (or cross-entropy loss in binary classfication):

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j}\log(p_{i,j})$$

2. Analysis

2.1. Data Exploration

Below is the head (first 20 entries) of the Quora dataset:

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0
5	5	11	12	Astrology: I am a Capricorn Sun Cap moon and c...	I'm a triple Capricorn (Sun, Moon and ascendan...	1
6	6	13	14	Should I buy tiago?	What keeps childern active and far from phone ...	0
7	7	15	16	How can I be a good geologist?	What should I do to be a great geologist?	1
8	8	17	18	When do you use シ instead of し?	When do you use "&" instead of "and"?	0
9	9	19	20	Motorola (company): Can I hack my Charter Moto...	How do I hack Motorola DCX3400 for free internet?	0
10	10	21	22	Method to find separation of slits using fresn...	What are some of the things technicians can te...	0
11	11	23	24	How do I read and find my YouTube comments?	How can I see all my Youtube comments?	1
12	12	25	26	What can make Physics easy to learn?	How can you make physics easy to learn?	1
13	13	27	28	What was your first sexual experience like?	What was your first sexual experience?	1
14	14	29	30	What are the laws to change your status from a...	What are the laws to change your status from a...	0
15	15	31	32	What would a Trump presidency mean for current...	How will a Trump presidency affect the student...	1
16	16	33	34	What does manipulation mean?	What does manipulation means?	1
17	17	35	36	Why do girls want to be friends with the guy t...	How do guys feel after rejecting a girl?	0
18	18	37	38	Why are so many Quora users posting questions ...	Why do people ask Quora questions which can be...	1
19	19	39	40	Which is the best digital marketing institutio...	Which is the best digital marketing institute ...	0

Number of training data: 404287
Percentage of duplicate question pairs: 36.92%
Total number of questions: 537929
Number of questions appearing multiple times: 111778

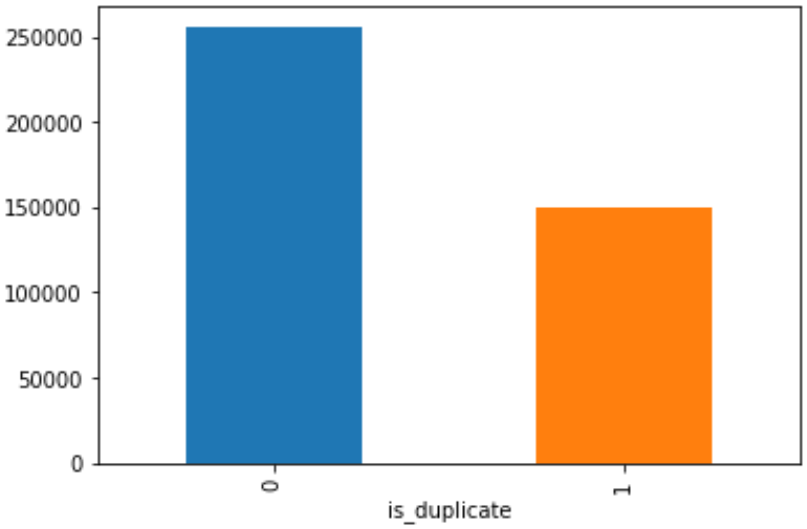


Figure: amount of duplicates and non-duplicates in training dataset

	id	qid1	qid2	is_duplicate
count	404287.000000	404287.000000	404287.000000	404287.000000
mean	202144.340337	217243.151093	220955.212082	0.369201
std	116708.673691	157751.614317	159903.168488	0.482589
min	0.000000	1.000000	2.000000	0.000000
25%	101071.500000	74436.500000	74726.500000	0.000000
50%	202145.000000	192181.000000	197053.000000	0.000000
75%	303216.500000	346573.000000	354692.000000	1.000000
max	404289.000000	537932.000000	537933.000000	1.000000

Figure: descriptive statistics of the training data

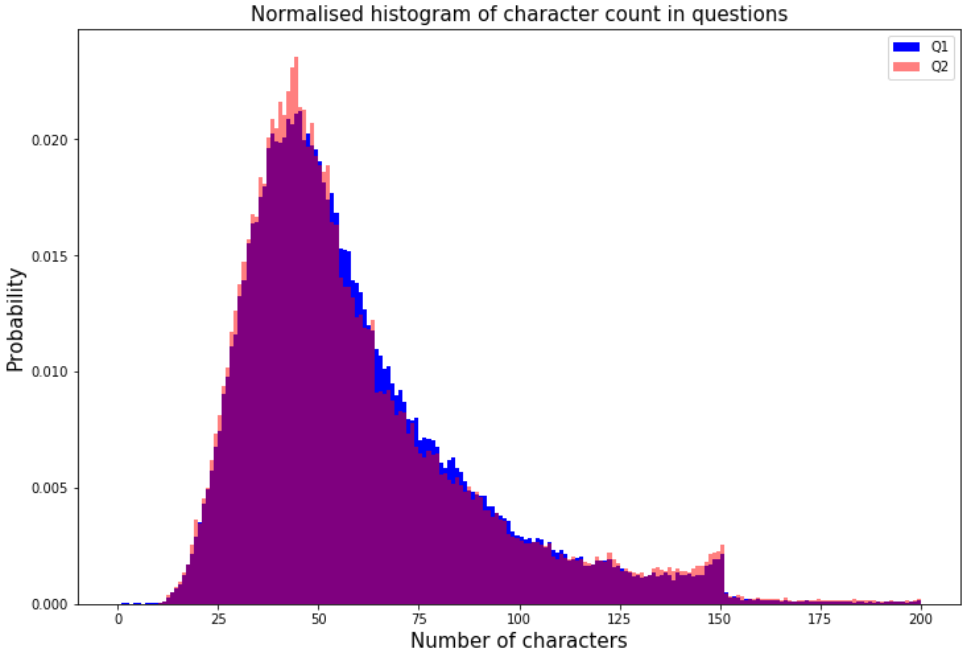
Explore the training set Questions:

```
Questions with question marks: 99.87%
Questions with [math] tags: 0.12%
Questions with full stops: 6.31%
Questions with capitalised first letters: 99.81%
Questions with capital letters: 99.95%
Questions with numbers: 11.83%
```

An interesting way to get a sense of the data is to generate a word cloud graph, which shows a bunch or words in a graph. The bigger the word font the more frequent it shows up in the word corpus. Here I combined question1 and question2 into one series and using wordcloud library to generate plots.

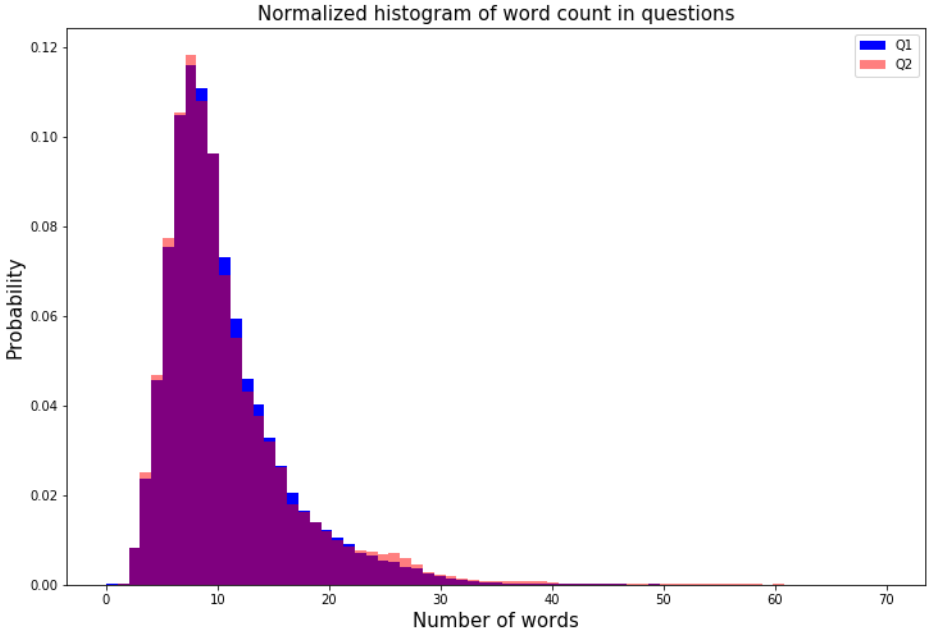
2.2. Exploratory Visualization

The first plot shows a normalized histogram of character count in both question1 and question2. Blue color represents question1 and red color represents question2. From the graph we can see that the character counts distributions are almost the same for question1 and question2. They all peak at around 45 characters and have an average of 55 characters. One interesting feature is at 150 characters there is a slight increase in numbers and then an abrupt drop. This may be from the rule of Quora website limiting the amount of words user can type in, with the exception of math equations or hyperlinks, hence explaining the little tail above 150 character count.



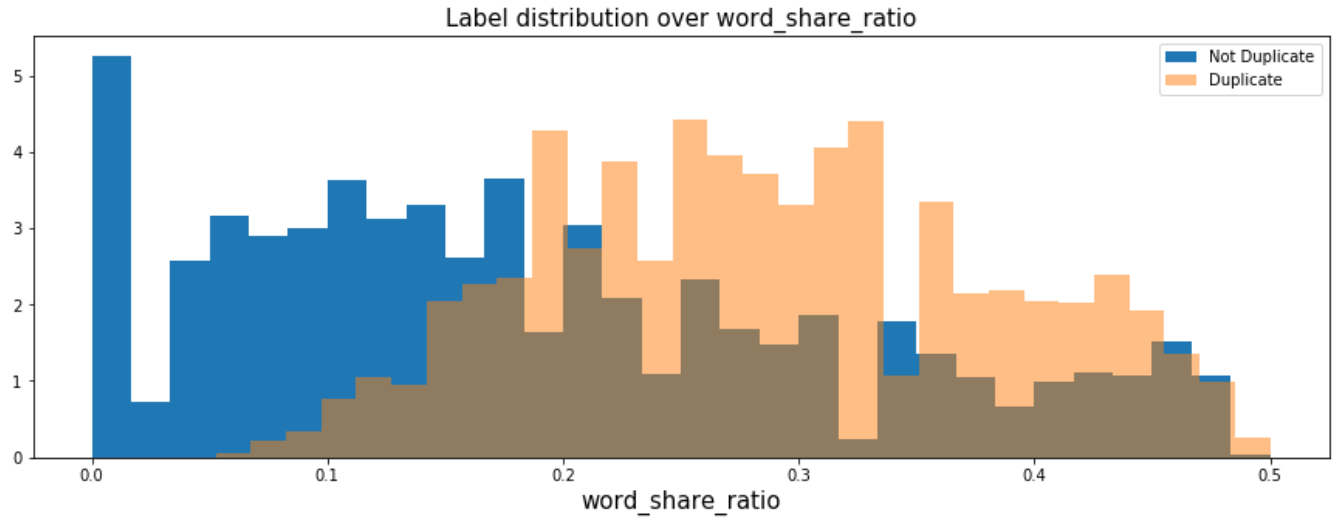
We can see that most questions have anywhere from 15 to 150 characters in them. It seems that the test distribution is a little different from the train one.

Second graph shows the normalized histogram of number of words from question1 and question2.

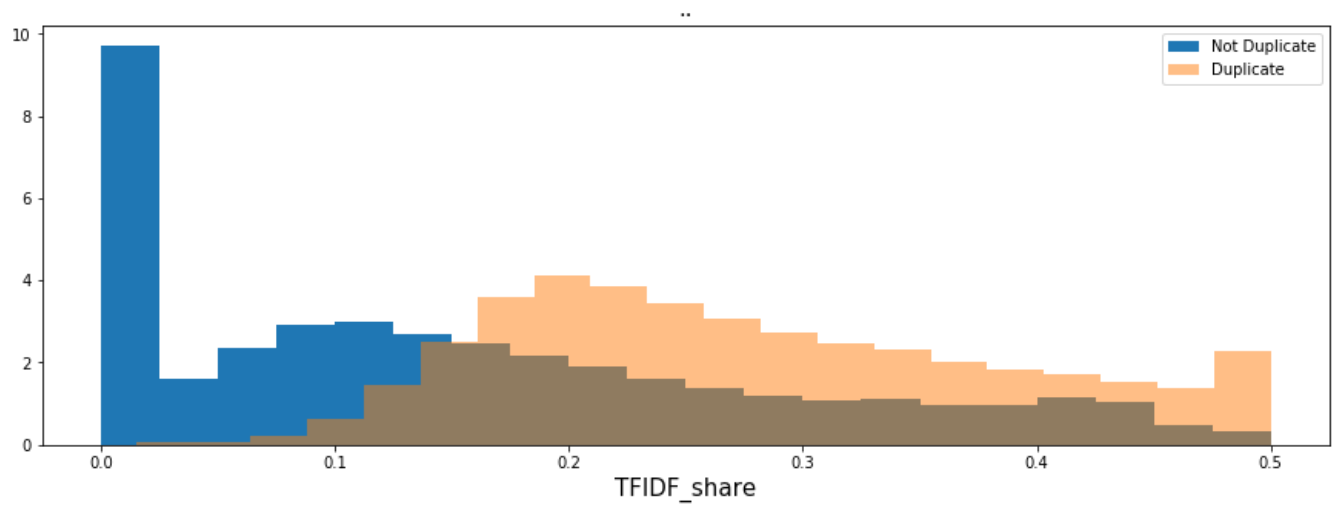


We see a similar distribution for word count, with most questions being about 10 words long. It looks to me like the distribution of the training set seems more "pointy", while on the test set it is wider. Nevertheless, they are quite similar.

The third graph shows the bar plot of word_share_ratio. Word_share_ratio is an extracted feature from the training data. The larger the ratio the more overlapped the questions pair is. In this graph, I use beige color for duplicate question pairs and blue color for non-duplicate question pairs. As we can see, duplicate question pairs generally have more words shared between them. This observation is what we would expect.



The final graph I generated is the TFIDF word share ratio. TFIDF word share ratio is also a feature extracted from the training data. The bigger the number, the more similar the two questions are.



2.3. Algorithms and Techniques

Since this is a natural language process binary classification problem, I will be using neural network algorithm. First I will clean up the strings by removing leading and trailing white space and converting to lower case. Then I will extract features from the attribute I am given on the data. These features include: 1. word count 2. character count 3. word share 4. TF-IDF share.

The algorithms I will be implementing are: 1. MLPClassifier 2. Sequential Algorithm.

I will use validation data set to compare these 2 algorithms and find the best one.

2.4. Benchmark

The benchmark model in my problem is the random forest model. The reason is that random forest is easy to implement and naive enough to be compared with other algorithms.

I ran the model on the train set and the baseline model gives an accuracy of 71.94%.

3. Methodology

3.1. Data Preprocessing

First I strip the string from leading and trailing white space, and then I convert it to all lower case. I also remove NaN values in character_count by replacing them with zeros.
There are a total of 6 features. Question1 number of words, Question1 character count, Question2 number of words, Question2 character count, word share ratio, TFIDF vector share ratio. The first four are extracted using basic python string functions.

Word_share is calculated as the number of words appearing in both question1 and question2 divided by the total number of words in question1 and question2. The result is a normalized value measuring how much word-overlap is between question1 and question2.
For TFIDF factors, I did not use the sklearn TFIDF library. Instead, I wrote a similar function implementing this TFIDF idea so that I don't feel I am using a blackbox function, and it gives me more freedom to do the TFIDF vector share calculation. First I combine all questions into one big corpus, and then calculate weight for each word. The weight is calculated by the inverse of word count plus epsilon (=5000) such that the less common words get higher weights. I ignored words that only appear once in the whole corpus for the reason that it may be typos. Then I calculate the tfidf_word_share_norm using share_weight divided by total_weight. Share_weight is the total weight of the words appearing in both question1 and question2. Total_weight is the total weight of every words in either question1 or question2. The result is a normalized TFIDF_weight_share. Since TFIDF discounts on the common words that appear

universally in our language, this measure represents more closely to how similar two questions are.

3.2. Implementation

Before starting to build the network, I started building the benchmark Random Forest classifier. it gave an accuracy of 71.94%

Then, I used the Sequential Algorithm:

1. Create the model

```
combinedModel = Sequential()
```

2. Add activations to the model: Exponential linear unit(elu) and Sigmoid activation function(sigmoid)

```
combinedModel.add(Dense(15, input_dim = X_train.shape[1],
activation='elu'))
combinedModel.add(Dense(5, activation='elu'))
combinedModel.add(Dense(1, activation='sigmoid'))
```

3. Compile the model

```
combinedModel.compile(loss="binary_crossentropy", optimizer="adam",
metrics=["accuracy"])
```

3. Fit the model with epochs of 20

```
combinedModel.fit(X_train, y_train, epochs=20, batch_size=64,
validation_data=(X_valid, y_valid))
```

the sequential model gave an accuracy of 72.68 and the loss: 0.4740.

This model accuracy is better than our benchmark. I tried to use MLPclassifier in order to raise the accuracy but i got an accuracy of 66.55% which is even worse than the benchmark model.

3.3. Refinement

I proceeded to process the inputs for the model - generating the work matrix for my word embeddings. This is done in a couple of steps:

1. Splitting the training dataset (**train.csv**) into a 80-20 split, using the 20% as a validation set of sorts to ensure better generalization can be achieved to achieve better results on the holdout test set on Kaggle.
2. Cleaning the datasets through importing NLTK's stopwords as well as a couple of regex formatting to ensure words are consistent.
3. Iterate through every sentence in both columns of both the training and testing datasets.
4. Since word vectors cannot be inputted and trained upon, they have to be converted into a numerical representation before training.

Regarding the tuning of the parameters, I decided to manually handpick a few values for the tunable hyper-parameters such as the number of epochs as well as the batch size.

4. Results

4.1. Model Evaluation and Validation

The final model's parameters were as such:

- Batch size of 64
- 20 epochs
- Adam optimizer, clipping the norm at a value of 0.9

The following activations were added to the model: Exponential linear unit(elu) and Sigmoid activation function(sigmoid)

The Adam optimizer was adopted as it is usually deemed the best optimizer by most

Model	Log-Loss Error	Accuracy
Random Forest Classifier	9.68	71.94%
Sequential Network Model	0.4740	72.68%

Figure1

4.2. Justification

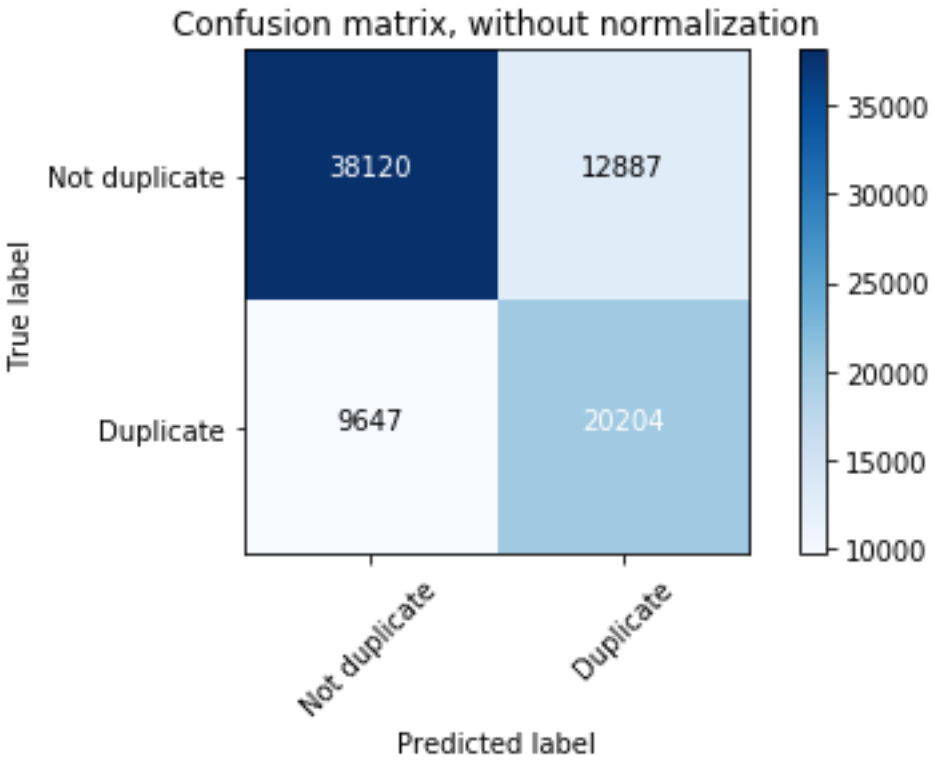
The results were compared for both models with reference to Figure1. A marked improvement in every metric that I have sought to compare for my original benchmark model and the refined model was observed.

I believe with the huge reduction in the weighted log-loss error which accounts for the imbalance in the dataset.

Last but not least, I managed to achieve a high accuracy but given that my computing power was limited, I believe I could have achieved a better score through more rigorous testing given a better GPU.

5. Conclusion

5.1. Free-Form Visualization



The Figure shows the confusion matrix for our model. t 38120 of the data was labeled non-duplicate when it's actually non-duplicate. and 20204 labeled duplicate when it's true that it's duplicate. I would consider this as a good result.

5.2. Reflection

Natural Language problems have been and are still a challenging form of issue to deal with. Even though much progress has been established within the field of Deep Learning and Machine Learning, great results that we may have achieved - may not mean that our prediction models have captured the semantic, syntactic meaning behind words.

Given that this was the first time I chose to do neural network model, I took a fair bit of research before I understood what was going on.

Plus, this project was tricky to me, the tricky part is to select what I want to use for my features, I first thought about the number of words and character length, and then the amount of the same words shared between two questions. Since this is obviously a NLP related problem, I thought of using TFIDF to process the sentences. Finally I ended up with 6 features.

I have tried to add more features extracted from TFIDF vectors, but did not get any significant accuracy gain. My reflection on this is: adding features is not always better. Because in my original features I already implemented my own TFIDF weighted word share, adding more TFIDF vectors later may not improve my model accuracy. It may still improve a little, as I have observed, but not by much. I would probably not want to spent an extra week of training time for a 1% accuracy improvement.

5.3. Improvement

To improve this project in the future, I will do more sentiment analysis. Right now I only have done simple string preprocessing and then strip out the words. In the future, I will spend more effort on natural language processing techniques. I may try extracting the punctuation, upper/ lower letter, special characters, foreign languages, math symbols, emojis...etc, for more detailed analysis.

If computing power allowed, I can also try doing deep learning neural network on this problem. Since we can extract over a million TFIDF vectors