Data Mining HW#2

103061135

Before Preprocessing

The dataset of this competition is pretty messy, so I use some code to clean them and same them as *.csv* file for later usage. Then, I don’t need to load all of them every time I want to train the model. In the raw data, there are lots of different data I can use such as “*_score*”, “*_index*”, “*_source*” and “*_crawldata*”. However, in the end, I just reserve the “*_source*” part, since I can’t come up a way to make good use of them.

An example of the data:

{"_score": 391, "_index": "hashtag_tweets", "_source": {"tweet": {"hashtags": ["Snapchat"], "tweet_id": "0x376b20", "text": "People who post \"add me on #Snapchat\" must be dehydrated. Cuz man.... that's <LH>"}}, "_crawldate": "2015-05-23 11:42:47", "_type": "tweets"}

Preprocessing

After clean the data, I have ‘*text*’, ‘*hashtag*’ and ‘*tweet_id*’ as features now. I use the same method used in the lab. I convert the text data to feature vector using TFIDF.

First attempt

First, I using **Xgboost** to classify the **TFIDF** feature vectors. However, it takes too much time to train on my laptop (even can’t get a single result to submit). To solve this problem, I reduced the **max_features** in **TFIDF** to 1000. But the results don't seem to be very good.

Second attempt

Then, I try them the same feature vector with a **naïve neural network** – the one that we use in the lab. The performance is pretty good compared to **Xgboost**. Also, I found out the training time also reduces a lot. So my next step is to try to increase the **max_features** of the **TFIDF**. The performance keeps growing as **max_features** keep increasing.

- Difficulty

The data set is really too large, it takes too much time to train on the machine, and due to time constraints, I have a hard time trying other architectures. Also, I didn't have a good solution for using *'hashtag'* and '*tweet_id*', so I gave up. The final version of the features I used only contain ‘text’ the embedded using **TFIDF**.
Also, I planned to train the neural network with more **max_features**, but my machine did not seem to be able to handle it. I also think that continuing to increase feature size may not be a good idea, as it will cause the curse of dimensionality that professors mentioned in class. Finally, I stopped my progress with **max_features** = 5000 in naive neural network models and **TFIDF**.