kaggle competition report

蔡佳靜 (108070027)

outline

- Report
- model explaination
- Observation
- something I tried to do but failed

Report

	_score	_index	_source	_crawldate	_type
0	391	hashtag_tweets	{'tweet': {'hashtags': ['Snapchat'], 'tweet_id...	2015-05-23 11:42:47	tweets
1	433	hashtag_tweets	{'tweet': {'hashtags': ['freepress', 'TrumpLeg...	2016-01-28 04:52:09	tweets
2	232	hashtag_tweets	{'tweet': {'hashtags': ['bibleverse'], 'tweet_...	2017-12-25 04:39:20	tweets
3	376	hashtag_tweets	{'tweet': {'hashtags': [], 'tweet_id': '0x1cd5...	2016-01-24 23:53:05	tweets
4	989	hashtag_tweets	{'tweet': {'hashtags': [], 'tweet_id': '0x2de2...	2016-01-08 17:18:59	tweets
...	...	...	...	...	...
1867530	827	hashtag_tweets	{'tweet': {'hashtags': ['mixedfeeling', 'butim...	2015-05-12 12:51:52	tweets
1867531	368	hashtag_tweets	{'tweet': {'hashtags': [], 'tweet_id': '0x29d0...	2017-10-02 17:54:04	tweets
1867532	498	hashtag_tweets	{'tweet': {'hashtags': [], 'tweet_id': '0x2a6a...	2016-10-10 11:04:32	tweets
1867533	840	hashtag_tweets	{'tweet': {'hashtags': [], 'tweet_id': '0x24fa...	2016-09-02 14:25:06	tweets
1867534	360	hashtag_tweets	{'tweet': {'hashtags': ['Sundayvibes'], 'tweet...	2016-11-16 01:40:07	tweets

First,I tidy up the data and analyze what I want. I analyze the score and eventually get id, text.

Then I generate the features with BOW, TF-IDF and method provided in <helper> document.

```
import nltk

# build analyzers (bag-of-words)
BOW_500 = CountVectorizer(max_features=500, tokenizer=nltk.word_tokenize, stop_words= "english")

# apply analyzer to training data
BOW_500.fit(train_df['text'])

train_data_BOW_features_500 = BOW_500.transform(train_df['text'])

## check dimension
train_data_BOW_features_500.shape
```

```
# build analyzers (bag-of-words)
BOW_1000 = CountVectorizer(max_features=1000, tokenizer=nltk.word_tokenize, stop_words= "english")

# apply analyzer to training data
BOW_1000.fit(train_df['text'])

train_data_BOW_features_1000 = BOW_1000.transform(train_df['text'])

## check dimension
train_data_BOW_features_1000.shape
```

I trained the features I got by two different ways, Decision tree, Naive Bayes.

```
## build DecisionTree model
DT_model = DecisionTreeClassifier(random_state=1)

## training!
DT_model = DT_model.fit(X_train, y_train)

## predict!
# y_train_pred = DT_model.predict(X_train)
y_test_pred = DT_model.predict(X_test)

## so we get the pred result
y_test_pred[:10]
```

```
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()

y_pred = clf.fit(X_train, y_train).predict(X_test)

y_pred[:10]
```

Latter, I train the feature with best accuracy by Deep learning.

```
from keras.callbacks import CSVLogger
import tensorflow as tf

csv_logger = CSVLogger('logs/training_log.csv')

# training setting
epochs = 25
batch_size = 32

# training!
history = model.fit(X,
```

```
                    y,
                    epochs=epochs,
                    batch_size=batch_size,
                    callbacks=[csv_logger])

print('training finish')
```

After that, I increased the letters I trained and repeat the process above.

```
BOW_1000 = CountVectorizer(max_features=1000, tokenizer=nltk.word_tokenize, stop_words= "english")
```

```
TF_1000 = TfidfVectorizer(max_features=1000, tokenizer=nltk.word_tokenize,stop_words="english")
```

and then I will talk about how I train my data and get its feature.

First, I generate the features in basic way.

```
BOW_500 = CountVectorizer(max_features=500, tokenizer=nltk.word_tokenize, stop_words= "english")
```

Then I generate the features with stop words which provided in nltk libaray.

```
BOW_500 = CountVectorizer(max_features=500, tokenizer=nltk.word_tokenize, stop_words= "english")
```

Model Explanation

I train the data with the best answer in each layer. For example, I train the data by Decision tree and Naive bayes with each features. And I train the data with stop words which get the best accuracy in last layer.

Observation

In the end, I got the best accuracy by getting the feature with TF-IDF, training the data with 1000 features, and deep learning. When I train the model, I observe something. With the stop words, I get the better accuracy with TF-IDF features. However, after deep learning, the accuracy is not that perfect, the progress is less than 1%. Moreover, I observe that the accuracy we get by TF-IDF and naive bayes is better. With deep learning, we get the better accuracy, but the progress is small. The reason can be attribute to the missing of the answer. I have tried to increase the epochs, however the progress is limited.

something I have tried but fail

first, I tried to eliminate the punctuation marks, but when I tried to deal with the features, I faced some errors caused by the data type.

So I tried to work on different way to get the feature, but the word2vec model can’t get the prediction but the relationship between the words.

then I tried to do the bert analyze, but my computer can’t run the data.