Problem Set 5

Ran Tao

(Dated: April 1, 2021)

Abstract

In this problem set, I built two different classifier, namely, naive bayes, and a simple neural network, to sort Twitter messages into 5 categories, Extremely Negative (0), Negative (1), Neutral (2), Positive (3), Extremely Positive (4). It turns out that using a simple neural network with a word embedding layer performs better.

I. INTRODUCTION

I was asked to build a classifier that can sort Twitter messages into 5 categories, Extremely Negative (0), Negative (1), Neutral (2), Positive (3), Extremely Positive (4). I first clean the text data, then tokenize the data, and then build a Naive Bayes classifier. Then I tokenize, vectorize and pad the data and build a simple Neural Network. In this Neural Network, I also embedding and pooling the data. Then I used Confusion Matrix, Precision Score, Recall Score, and F1 Score to evaluate the model. It turns out the simple Neural Network classifier performs better.

II. DATA EXPLORATION

The data set contains two parts. The first part is the label. 0 represents Extremely Negative, 1 represents Negative, 2 represents Neutral, 3 represents Positive, and 4 represents Extremely Positive. The second part the text from Twitter. It has 37041 data.

III. DATA PREPROCESSING, MODEL EVALUATION AND MODEL SELECTION

The Twitter data is very 'dirty'. Beside words, it also contains other symbols, like @, http, ?, !, *... So, I first delete those symbols by importing re and using re.sub(). When building Naive Bayes model, I tokenize the data using RegexpTokenizer() from NLTK, then I used CountVectorizer() from sklearn to convert a collection of text documents to a matrix of token counts. Then I split the dataset into training data and testing data, and used MultinomialNB() from sklearn.naive_bayes to build a Naive Bayes Classifier. Then I computed the Confusion Matrix, Precision Score, Recall Score, and F1 Score to evaluate the model. The Confusion Matrix, see Fig.1. The Micro Precision score is:0.46, the Micro Recall score is:0.46, Micro F1 score is:0.46.

When building a simple Neural Network, I first used one-hot encoding to encode the label data. Then I used Tokenizer() from tensorflow to tokenize the data, I called tokenizer.texts_to_sequences to vectorize the data, then used pad_sequences() to pad the data. Then I build a simple Neural Network with a Embedding and Pooling layers. The activation

```
Confusion matrix is
[[ 481  783   24  210   20]
 [ 223 1355  110  912   98]
 [  34  434  693  823   84]
 [  49  582  158 1912  401]
 [   7  131   19  878  692]]
```

FIG. 1. Confusion Matrix of Naive Bayes Classifier

function is softmax. The accuracy and loss of the training and validation data see Fig.2.

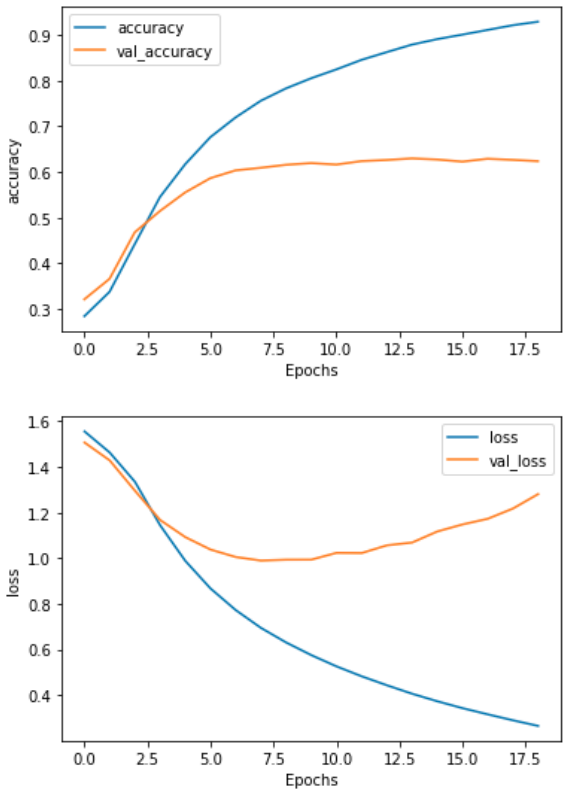

FIG. 2. The accuracy and loss of the training and validation data

After fitting this model data, I computed the Confusion Matrix, Precision Score, Recall Score, and F1 Score to evaluate the model. The Confusion Matrix, see Fig.3. The Micro Precision score is:0.63, the Micro Recall score is:0.63, Micro F1 score is:0.63.

In conclusion, the Neural Network classifer performs better than the Naive Bayes Model.

```
Confusion matrix is
[[ 882  541   34   40   21]
 [ 222 1798  285  355   38]
 [  13  395 1297  344   19]
 [  42  500  283 1982  295]
 [   3   32   24  600 1068]]
```

FIG. 3. Confusion Matrix of the Neural Network

IV. INTERPRETATION

The Neural Network classifer with a word embedding layer performs better than the Naive Bayes model. Given a specific dataset, the neural network can better more Twitter messages into the correct categoires.

V. CONCLUSIONS

I test two classifier, namely, Naive Bayes and Neural Network, that can sort Twitter messages into 5 categories, Extremely Negative (0), Negative (1), Neutral (2), Positive (3), Extremely Positive (4). It turns out the simple Neural Network classifier with a word embedding layer performs better.

DATA AVAILABILITY

Data is available at https://www.kaggle.com/c/usc-dsci552-section-32415d-spring-2021-ps5/data.

CODE AVAILABILITY

Code is available at https://github.com/usc-dsci552-32415D-spring2021/problem-set-05-rantao-usc/blob/main/Twitter