Sentiment Analysis of twitter comments

Tatiana Nyanina

Jan 2022

Abstract

Analyzing Twitter user sentiment with machine learning models.

1 Introduction

The classification of the text in a positive and negative way has recently been in very high demand among businesses, as there have been opportunities for collecting big data from users. In this connection, in this task, the model will be trained on the training dataset and the complaints will be classified on the test set with preliminary data preprocessing.

2 Related works

In this section, I will outline the work on the analysis of the mood of Twitter users.

The first of the works I will present the following: https://www.kaggle.com/code/irasalsabila/twitter-sentiment.

In this paper, three classes are taken for analysis: neutral, positive, negative. For training, already cleaned data was taken and vectorized with the number of words = 10000, and therefore the quality of the model at the output could be lower than possible. The architecture of the model is consistent with the embedding layer, LSTM layer, etc., the compilation method is used with the RMSprop optimizer, which increases the speed of model learning. As a result of training the model, it was possible to achieve a quality of 83.479%[1].

In the second work which is the source https://www.kaggle.com/code/linkanjarad/pytorch-transformer-classification-acc-0-845.

In this paper, three classes are taken for analysis: neutral, positive, and negative. For training, already cleaned data was taken. The architecture of the model consists of fully connected layers, disconnecting neurons after them, Applies the Sigmoid Linear Unit (SiLU) function. The model has been defined with parameters such as embedding_size=256, src_vocab_size=10000, dropout=0.5, num_classes=3. "Adam" was selected as the optimizer. Loss function – CrossEntropyLoss. As a result of training the model, it was achieved a quality of 85%[2].

[1] https://www.kaggle.com/code/irasalsabila/twitter-sentiment
[2] https://www.kaggle.com/code/linkanjarad/pytorch-transformer-classification-acc-0-845

3 Model description

In this work, raw data from https://www.kaggle.com/datasets/yasserh/twitter-tweets-sentiment-dataset?select=Tweets.csv was used. With the help of tokenization, removal of characters and stop words, followed by lemmatization, the data was preprocessed. After preprocessing the data, the neutral class for binary classification was removed, and data analysis was performed: the classes are balanced. These phrases were vectorized using the tf-idf method. After vectorization, the data were divided into test and training sets. The following classifiers were chosen for training:

The accuracy of the model was 86,8%.

4 Dataset

The dataset is available at: https://www.kaggle.com/datasets/yasserh/twitter-tweets-sentiment-dataset?select=Tweets.csv. The data contains case number, data class, raw data, and raw data. Sample data is shown below[3]:

	textID	text	selected_text	sentiment
1	549e992a42	sooo sad miss san diego	Sooo SAD	1
2	088c60f138	bos bullying	bullying me	1
3	9642c003ef	interview leave alone	leave me alone	1
4	358bd9e861	son put release already bought	Sons of ****,	1

The most often words of positive semantic:

And negative meaning:

[3] https://www.kaggle.com/datasets/yasserh/twitter-tweets-sentiment-dataset?select=Tweets.csv

The goal of this study is to achieve high quality in training a model for classifying user comments into positive and non-aggressive on the training set to determine the class of other comments for the indicated gradation.

5 Experiments

5.1 Metrics

Due to the balance of the sample, quality was chosen as a metric for evaluating the model, the formula is presented below:

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP},$$

where TP – True Positive, TN – True Negative, FN – False Negative, FP – False Positive observations in the model[4].

5.2 Experiment setup

Four models were trained using four classifiers. The results of the models were as follows:

5.2.1 Logistic Regression

```
[[2202  332]
 [ 320 2055]]
              precision    recall  f1-score   support

           0       0.87      0.87      0.87      2534
           1       0.86      0.87      0.86      2375

    accuracy                           0.87      4909
   macro avg       0.87      0.87      0.87      4909
weighted avg       0.87      0.87      0.87      4909
```

5.2.2 KNeighborsClassifier

```
[[1534 1000]
 [1040 1335]]
              precision    recall  f1-score   support

           0       0.60      0.61      0.60      2534
           1       0.57      0.56      0.57      2375

    accuracy                           0.58      4909
   macro avg       0.58      0.58      0.58      4909
weighted avg       0.58      0.58      0.58      4909
```

[4] https://pythonru.com/baza-znanij/metriki-accuracy-precision-i-recall

5.2.3 SGDClassifier

```
[[2218  316]
 [ 332 2043]]
              precision    recall  f1-score   support

           0       0.87      0.88      0.87      2534
           1       0.87      0.86      0.86      2375

    accuracy                           0.87      4909
   macro avg       0.87      0.87      0.87      4909
weighted avg       0.87      0.87      0.87      4909
```

5.2.4 RandomForestClassifier

```
[[2197  337]
 [ 362 2013]]
              precision    recall  f1-score   support

           0       0.86      0.87      0.86      2534
           1       0.86      0.85      0.85      2375

    accuracy                           0.86      4909
   macro avg       0.86      0.86      0.86      4909
weighted avg       0.86      0.86      0.86      4909
```

The lowest quality was shown by the KNN model: 0.66. There are all classes have many observations from another class.

In final training, the default hyperparameters were used because using other parameters resulted in the model having the lowest quality <0.8.

5.3 Baselines

A simple model was used as a basis: logistic regression, RandomTree, KNN and SGD classifier were trained in TF/IDF words.

6 Results

As a result, the model showed the best quality: SGDClassifier:

```
{LogisticRegression(): 0.8672,
 KNeighborsClassifier(): 0.5844,
 SGDClassifier(): 0.868,
 RandomForestClassifier(): 0.8576}
```

Model quality = 86,8%.

The model is well applicable to data arrays with a small number of rows.

7 Conclusion

Finally, the goal was achieved. The data was preprocessed with stopword removal and lemmatization. Comments have been vectorized. The classifiers are trained. The best was the model using the SGD with quality = 87% on the test data.

References

1. Twitter Sentiment. [Date of the application = 11.01.2022] URL: https://www.kaggle.com/code/irasalsabila/twitter-sentiment
2. Pytorch Transformer Classification Acc: 0.845. [Date of the application = 10.01.2022] URL: https://www.kaggle.com/code/linkanjarad/pytorch-transformer-classification-acc-0-845
3. Twitter Tweets Sentiment Dataset. [Date of the application = 11.01.2022] URL: https://www.kaggle.com/datasets/yasserh/twitter-tweets-sentiment-dataset?select=Tweets.csv
4. Estimating ML/DL Models: Error Matrix, Accuracy, Precision and Recall. [Date of the application = 11.01.2022] URL: https://pythonru.com/baza-znanij/metriki-accuracy-precision-i-recall