Text as Data: assessed exercise

Mohammad Otoofi

2263537O

1 Q1

The TfidfTransformer transforms a count matrix to a normalized tf-idf representation. However, CountVectorizer represents a document with term frequency vector which its length is equal to the size of vocabulary. As shown in the fig1, CountVectorizer gives better results for both Logistic Regression and SVM. these result show that common repeated words in the documents, which are penalized by tf-idf, are effective in this particular case and can be used to classify subreddits more effectively.

Model	Precision	Recall	Accuracy	F1
CountVectorizer + LogisticRegression	0.595	0.746	0.690	0.633
CountVectorizer + SVM	0.083	0.115	0.282	0.067
CountVectorizer + DummyClassifier(most_frequent)	0.050	0.012	0.230	0.019
CountVectorizer + DummyClassifier(stratified)	0.033	0.029	0.082	0.030
TfidfVectorizer + LogisticRegression	0.315	0.568	0.523	0.347
TfidfVectorizer + SVM	0.050	0.012	0.230	0.019
TfidfVectorizer + DummyClassifier(most_frequent)	0.050	0.012	0.230	0.019
TfidfVectorizer + DummyClassifier(stratified)	0.033	0.029	0.082	0.030

Figure 1: Macro results of different combination of vectorizers and classifiers

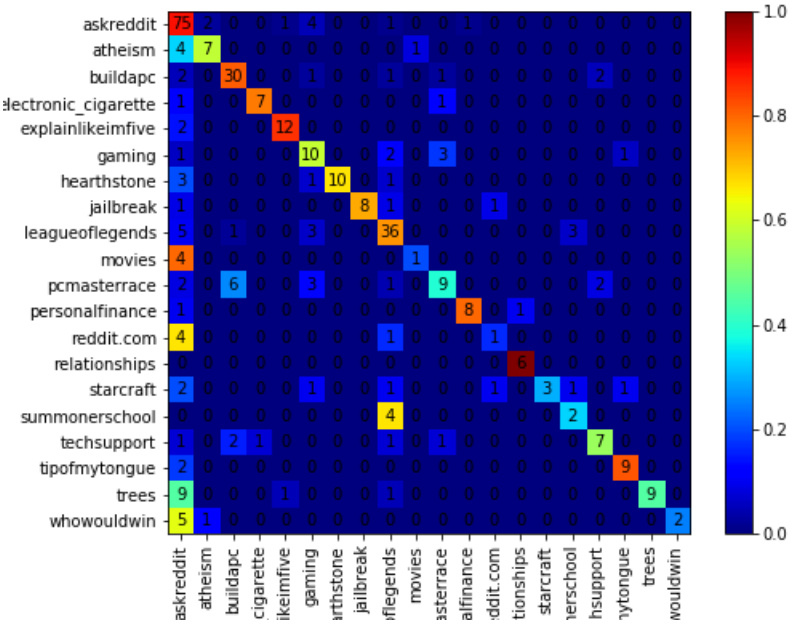

Figure 2: Confusion matrix for Logistic regression with CountVectorizer

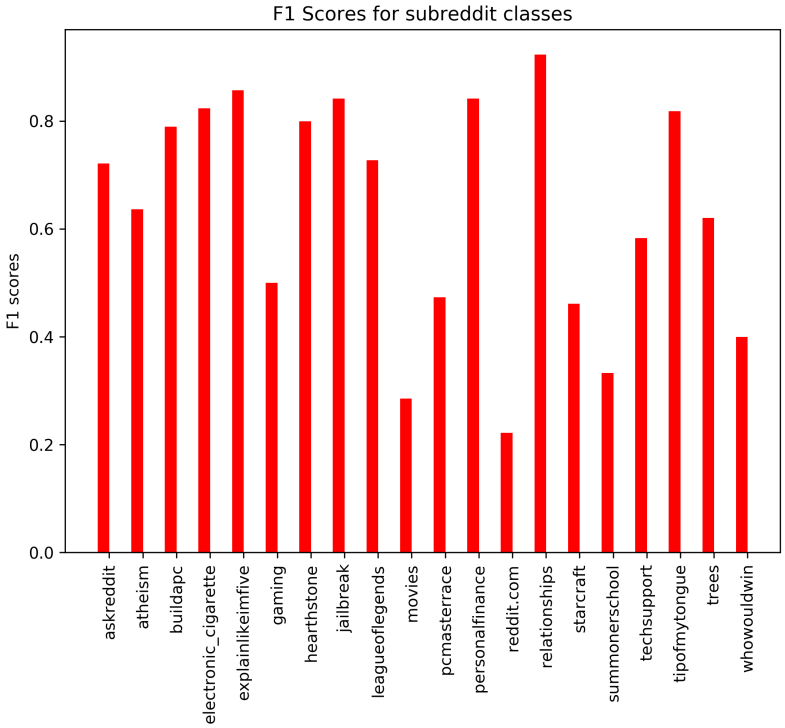

Figure 3: F1 Scores for subreddit classes for Logistic regression with CountVectorizer

2 Q2

In terms of regularizer, "C", for both SVM and Logistic Regression the largest value, 100, is getting the best result which means weaker regularizer performs better. It could be because the training data is from wide spectrum so that the model is not overfitted to training data. Furthermore, Logistic Regression gives better result if log scaling is applied to the representation. This makes high frequent words less discriminative. In addition, both SVM and Logistic Regression perform better with a smaller vocabulary size which means the first 5000 frequent words are more discriminative and important. In addition, Logistic Regression can classify the subreddits better with unigram model while SVR can perform better with trigrams which provides more features to SVR for classification.

Model	C	Sublinear TF	Ngram range	Vocabulary size
SVR	100	False	$(1, 3)$	5000
Logistic Regression	100	True	$(1, 1)$	5000

Figure 4: Best parameters using TF-IDF

Model	Precision	Recall	Accuracy	F1
SVR	0.074	0.060	0.279	0.052
Logistic Regression	0.676	0.767	0.762	0.699

Figure 5: Macro results of the best models

3 Q3

Precision	Recall	F1
0.228	0.370	0.247

Figure 6: Macro Precision, Recall and F1 scores of the LogisticRegression classifier

Class	F1
1	0.33
2	0.09
3	0.56
4	0.65
5	0.08
6	0.21
7	0.04
8	0.08
9	0.09
10	0.34
Avg	0.48

Figure 7: F1 performance for each class

4 Q4

List of features:

- A tokenizer that includes punctuation as tokens instead of removing it (including "?" and "!"): for a reader punctuation helps to understand the intention of the writter better. Thus, adding punctuation can help in the same way to increase performance. In addition since reddit posts consists of question, answer, or giving own opinion about a topic, punctuation is widly use and can be considered as a dicriminative features.
- Raw depth of the comment: depth of a comment could be informative so that question and elaboration posts would have more comments than other posts.
- A binary feature for whether the current author is also the author of the initial post: this can give a clue if the post is an appreciation.
- The total number of comments in the discussion: this feature would be useful to be used to identify announcement posts which have only one post.
- The subreddit the post came from: this feature can be used to identify the subreddits with controversial topics. So controversial topics can be identified and be distinguished from the ones which are related to announcements posts which tends to be categorized in uncontroversial subreddits.
- Average of pre-trained Glove embeddings for the post: avg of embeddings of the posts can be used as an identification of the posts in which similar words are used.

Model	Precision	Recall	Accuracy	F1
All features	0.320	0.494	0.591	0.341
All features - {punctuations}	0.262	0.435	0.519	0.288
All features - {depth of the comment}	0.285	0.445	0.564	0.302
All features - {author}	0.306	0.502	0.583	0.331
All features - {number of comments}	0.321	0.502	0.590	0.341
All features - {subreddit}	0.338	0.461	0.579	0.350
All features - {average Glove embeddings}	0.320	0.495	0.590	0.339

Figure 8: Macro results of the models