Kaggle: Text Classification Report

Student name - Debabrata Champati
Student ID- A0206519E

Data used for the classification exercise

The data consists of unstructured text answers posted by users in a programming forum on the internet. There are 44,183 records available for training out of which around 25,000 records are tagged as Class 1 and rest class 0. Hence, there is fair degree of balancing available already in the training data.

Data pre-processing

a. **Removing stop words**
   The general stop words in English language are available within NLTK and this set has been used to remove them, as they might not have added value in the determining the class.
b. **Highlighting R**
   The presence of the programming language R in the answers might have some additional information on the language used in the answer. This is mentioned as 'r' or 'R' in answers. While trying out some token patterns and ignoring the one lettered words, 'R' might be ignored. Hence, 'r' is replaced with 'programmingr' beforehand to avoid elimination.
c. **Lemmatization but preservation of characters**
   Textblob has been used for lemmatization of the words. However it is made sure that the characters are intact because they have a special role to play in the answers as can be seen later on.
d. **Understanding the word cloud**
   Referring to Appendix1, the wordcloud on the left is for the words present in answers with Class 1 and on the right is the words which most frequently occur for answers with Class 0. It can be inferred that the top words appearing in both the classes are similar and what might make the difference between them is actually the presence of other special characters or low frequency words.

Feature engineering

a. **Tokenization**
   Tokenization is tested by using both countVectorizer and TfIDf vectorizer and the latter performs better with the data at hand. N-gram range is gradually varied to obtain the balance between capturing syntactical and sematic information and (1,4) is decided on as it results in better accuracy. The binary option for the vectorizer has been activated which means that the classification is based on presence of the words rather than the number of times they appear in a particular answer. This boosts the accuracy by ~1.5%.
b. **Stacking the length of the answer**
   In general, the length of the answer might play an important role in the response received for it. It may happen that short and crisp answers with appropriate keywords will have more upvotes as compared to lengthy text. Normalized lengths having been stacked on top of the document matrix generated by TfIdf.

c. **Stacking the composition of the answer**
   As the text under study are answers to questions posted on a programming forum, it is possible that the answers containing specific codes will have more upvotes as compared to textual answers. Hence, the composition of the answer in terms of the alphabet to special characters ratio is calculated and stacked on top of the document matrix
d. **Stacking the Topic alignment characteristics through Latent Dirichlet Allocation**
   It is possible that the answers revolve around certain topics and that the upvotes by the users for each answer can be attributed to the amount of alignment of the specific answer to the topics. Hence, 100 topics have been identified from the answers in general using LDA and level of alignment of an answer to each topic is calculated by transforming the vectorized token on the learned topics. This results in additional 100 features for each answer which is then stacked on top of the document matrix designed. Refer Appendix 2 for example topics.

Model building

There are two categories of model chosen for the purpose of text classification- Planar models and Tree based classification problems with varying hyper parameters.For this dataset, the planar models outperform the tree-based models. The final model selected however is SVM with a 'Rbf' kernel and degree of 2.

Model validation

Grid search CV for the purpose of cross validation was the first option. However, the grid search takes considerable amount of time for certain models like SVM. Hence, the models are validated on the 25% randomized test split which was not used for fitting the model. Appendix3 can be referred for the composition of the validation. The train test split also preserves the 0-1 ratio across the splits (as shown in the following table). The test size is large enough to be attributed just to mere coincidence. Appendix 4 can be referred to for the validation accuracies of the four different models tried.

Final models proposed

Although the above descriptions are for the model which can be explained better, the predictive power seems to increase when the analyser in TfIdf vectorizer is changed to 'char' instead of 'word' and this improves the validation accuracy by around 1.6-2%. The additional feature engineering mentioned above does not have incremental benefit when the 'char' is selected as analyser. Hence, the final two models proposed are:

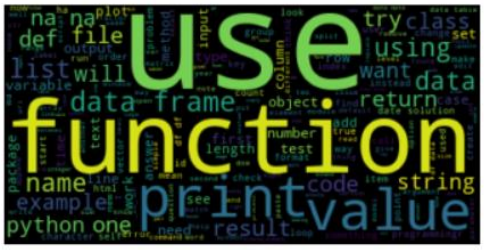
Model 1 – *better interpretability but slightly lower accuracy* (All the above mentioned data processing, feature engineering, Analyzer 'word' in TfIdf vectorizer with N Gram (1,4), binary=True followed by a SVC classification).

Model 2 – *lower interpretability but higher accuracy* (No additional feature engineering, Analyzer 'char' in TfIdf vectorizer with N Gram (1,5), binary=True followed by a SVC classification).

Models are retrained on the full training data set and used for the prediction

Appendix

Appendix1



Appendix2

```
Topic #0:
data function frame data frame column matrix true row vector length false lapply list col value

Topic #1:
python file http install package import use code com module path pip using version command

Topic #2:
django request user query form model field json objects models input sql filter template session
```

Appendix3

Train-Test split for validation used in place of Grid search CV. The composition of the split is as follow:

Split	Number of 1's	Number of 0's	0:1 ratio
Train	18743	14394	77%
Test	6249	4797	77%

Appendix4

Results of the model validation are as follows:

Algorithm	Accuracy on validation set
Multinomial Bayes	67.58%
GradientBoosting Classifier	67.41%
Logistic Regression	71.70%
SVC	72.10%