NLP 220 Assignment 3

University of California, Santa Cruz

Multilabel Classification of arXiv Paper Summarizations

In this assignment, you will process data, engineer features, and train a model to predict which category (or categories) an arXiv paper's abstract belongs in. This is a **multi-label** classification problem, so recall there can be 0 or more correct labels.

Dataset Preparation  (5pt)

Your dataset is simple, and is provided as a JSON (arxiv-data.json) in data folder under files) with two columns. In one column, you have the paper abstracts. In the second, you have a list of correct labels.

You are required to build features which help you predict a class label. For example, one feature might be the presence of the phrase `machine learning`, which could be a good indicator of the label `stat.ML`.

* Use 85% of the data for training and the remaining 15% for testing.
* Use 15% of the training data for validation (70% training, 15% validation, 15% testing).
* Print or plot the distribution of your classes/labels, and include these in the report.

Training & Feature Engineering (45pt)

For this multi-label classification problem, you should perform data preprocessing, feature engineering, and model selection based on what you have learned in previous assignments and in class. You are not restricted to any particular choices, but you should experiment with several options of models, features, and data processing.

* You can use any of the libraries we’ve been using (pandas, numpy, sklearn, spacy/nltk). If you have another library you’d like to use, ask in the slack channel.
* Ensure you report your validation scores for each model, and then report the test score for your best model. You shouldn’t be experimenting with test data!
* Try at least 5 different model choices (e.g, Naive Bayes, SVM)

Feature Engineering Deliverables (40pt)

* Briefly (1-2 sentences each) describe the features you created.
* Compare the micro and macro F1 scores across your models, and discuss how your features/model choices affected each. In particular, note any features/models which were highly impactful, whether positively or negatively. Discuss why you think that feature is useful/not useful.
* There are 88 labels in the output set. Is it worth trying to classify all 88? Discuss why or why not, and cite data from your experiments to support your theory.

Training/Inference time comparison (5pt)

Compare training and prediction/inference time of each of your classifiers. Which one would you recommend based on accuracy and train/inference time?

**Performance (15pt)**

This is an open-ended homework, but you’re still expected to experiment with many different models, data techniques, etc. to achieve a high performing model. To encourage this, you will be graded on your best performing model’s multilabel micro F1 score in comparison to the rest of the class. The highest overall F1 score will be awarded 15 points, and lower performing models will be graded compared to the best.

To keep things fair, use the following snippet of code to split your data into train/val/test data. After you run this, you can do any sort of processing you want.

Note: **do not make any modifications to the “terms” column in the test split**. You are free to do any processing you want in the train/val splits.

import pandas as pd

from sklearn.model\_selection import train\_test\_split

df = pd.read\_json('/path/to/arxiv\_data.json')

train, valtest = train\_test\_split(df, test\_size=0.30, random\_state=1234)

val, test = train\_test\_split(valtest, test\_size=0.50, random\_state=1234)

# This will give you a 70/15/15 split.

print(len(train), len(val), len(test))



Submission

* **Report** containing a full writeup of your experiments, data preparation, feature engineering, and model choices, along with your results and a discussion/analysis.
* **Code as a .py file**, which should be runnable with the following command. I will supply the correct JSON path to the data, and the output should be a txt file which contains a classification report for your best performing model on the validation and test data. **Note - your final code only needs to run your best performing model/features - not all of your experiments!**

$ python homework3.py --data "arxiv\_data.json" --output "results.txt"

$ cat results.txt

Validation Classification Report:

             precision    recall  f1-score   support

     class 0       0.50      1.00      0.67         1

     class 1       0.00      0.00      0.00         1

     class 2       1.00      0.67      0.80         3

    accuracy                           0.60         5

   macro avg       0.50      0.56      0.49         5

weighted avg       0.70      0.60      0.61         5

Test Classification Report:

             precision    recall  f1-score   support

     class 0       0.50      1.00      0.67         1

     class 1       0.00      0.00      0.00         1

     class 2       1.00      0.67      0.80         3

    accuracy                           0.60         5

   macro avg       0.50      0.56      0.49         5

weighted avg       0.70      0.60      0.61         5

* README.md containing the following information:

1. The type of your best performing model (e.g., Decision Tree)
2. A short description of any data preprocessing you did (e.g., removing stopwords, cleaning up labels, etc.)
3. A short description of the features you used for the best performing model (e.g., “length of abstract”, “existence of ‘machine learning’ in abstract”, etc)
4. Your micro and macro F1 scores for your best performing model
5. Anything else you’d like me to know about your code, such as any known bugs or mistakes, or just rationale for why you did something a certain way.