Data Wrangling Coursework 2

Sam Dixon
40056761@live.napier.ac.uk
Edinburgh Napier University
School Of Computing

April 2018

1 Introduction

This report describes the process of creating and evaluating a movie review sentiment classification system.

2 Data Processing

The following preprocessing techniques were applied to the provided data file:

- Removed capitalisation.
- Removed repeated lines.
- Shuffled the order of lines.
- Split sentiment and review into separate lists.

3 Feature Extraction

Once the data was processed, feature extraction was performed. Text frequency-inverse document frequency (tf-idf) was utilised to calculate weights for each unigram in the corpus [1]. No minimum or maximum document frequency (cut-off) was applied to remove common or uncommon corpus words - these methods could be explored further to alter the accuracy of the classification.

After features were extracted, datasets could be constructed. Both standard split, and k-folded datasets were generated to allow a comparison of techniques.

The split dataset consisted of 9:1 training/testing split - the order of the features was shuffled to prevent either the training or testing sets containing only one class of review (positive or negative) [2].

The k-folded dataset was folded 10 times, with 9 folds being used to train a classifier, and the remaining 1 fold used to test [3]. Like the split dataset, the order of the features was shuffled to ensure uniform distribution of sentiment classes.

4 Classification Model

Three different SVM kernels were tested in this project:

- Linear.
- Radial Bias Function (RBF).
- Polynomial.

The default `scikit-learn` implementation for each kernel was utilised, further experimentation and turning of kernel parameters could be investigated to improve their accuracy [4, 5].

5 Evaluation

Each SVM kernel's results was evaluated using the F1-score function [6]. F1-score is a commonly used metric for the comparison of algorithm accuracies. The F1-score of a system is calculated based off of

Kernel	Dataset	Average F1-Score
Linear	Split	0.963636363636
Linear	Fold	0.961481481481
RBF	Split	0.536026936027
RBF	Fold	0.540740740741
Polynomial	Split	0.52861952862
Polynomial	Fold	0.540740740741

Table 1: Average kernel F1-score over 10 tests.

its Precision and Recall. F1-score is typically a more accurate measure of an algorithm's performance then simply comparison the number correct and incorrect predictions.

Each of the three kernels was trained and tested with the split and k-folded datasets 10 times, and the average F1-score for each was calculated. Each dataset was randomly generated from the tf-idf feature list before training and testing. The results for each kernel and dataset is displayed in Table 1.

6 Results

As can be seen, the k-folded dataset is only marginally superior to the split alternative. The similarity in results is likely due to the limited size of the data set.

In terms of kernel performance, the linear SVM classifier obviously outperforms the alternative methods. As stated, this may be due to the lack tuning of the RBF and Polynomial kernels, which feature a wide array of variables that can be altered to improve accuracy.

7 Conclusion

From the results of testing each kernel and dataset the following observations can be made:

With a limited amount of data, the manner used to divide the dataset is of little consequence, provided numerous test are conducted.

In terms of SVM kernel choice, linear produces satisfactory results with little to no tuning requirements. The RBF and polynomial kernels may be able to produce similar or superior results to the linear kernel, provided adequate time is available to tune and test the kernel's parameters.

Project available at: `https://github.com/neaop/set11521_coursework_2`.

References

[1] "sklearn.feature_extraction.text.tfidfvectorizer." `http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html`.

[2] "sklearn.model_selection.train_test_split." `http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html`.

[3] "sklearn.model_selection.kfold." `http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html`.

[4] "1.4. support vector machines." `http://scikit-learn.org/stable/modules/svm.html`.

[5] "1.4. support vector machines." `http://scikit-learn.org/stable/modules/svm.html#svm-kernels`.

[6] "sklearn.metrics.f1_score." `http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html`.