Jessica Lin

CSE 352 – AI

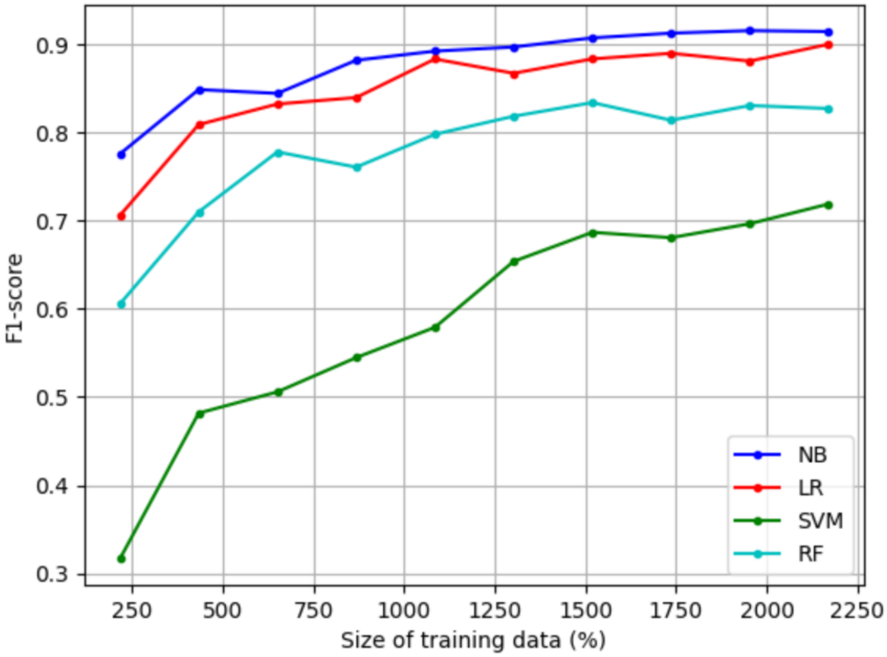
Assignment 3 – Text Classification

1. Basic Comparison with Baselines

1.1. Results

Configuration	Macro-Precision	Macro-Recall	F1-Score
NB,UB	0.9205508373360588	0.9107479823359221	0.9143565233888378
NB,BB	0.8928696506745287	0.8709828942693265	0.8771200045750571
LR,UB	0.9040365729133806	0.8973843459722857	0.9000113480911696
LR,BB	0.8693641875625819	0.8454452058271155	0.8519678193233128
SVM,UB	0.78857656681561	0.715355819501548	0.7188571916010096
SVM,BB	0.8395338875428737	0.768420587787422	0.7731370792216015
RF,UB	0.897873024914035	0.8337857976752449	0.8381326135707142
RF,BB	0.8434348490732206	0.7924708390437034	0.7986769765340437

1.2. Learning Curve



For each of the four algorithms, as the training data sized increased, the f1-score increased as well. The f1-score increases because there is more data for the learning algorithms, lowering the error rate. There are instances where increasing data could decrease accuracy due to overfitting.

However, in this case, the training data provided isn’t so complicated that it results in overfitting, and instead increases the accuracy. From most accurate to least accurate, the naïve bayes algorithm is the most accurate, followed by logistic regression, random forest, and finally SVM. Since naïve bayes typically performs better than logistic regression when the data set contains bias, it can be gathered that certain words in each data file are biased towards the class they are in. Logistic regression works well when there’s already identified independent variables, applying to our current data set. The random forest algorithm does well when the prediction trees have low correlation to each other, so it may have performed lower compared to the other two algorithms because of this. SVM which works well when there’s unstructured data, which may be the reason why is performs worse when compared to the other algorthms.

2. My Best Configuration

2.1. Exploration

Configuration	Macro-Precision	Macro-Recall	F1-Score
NB,CV+UB	0.9378856741181141	0.9400232983097304	0.9388907454064539
LR,TFIDF+UB	0.925487373574001	0.9200169534541394	0.9223386613628897
SVM,TFIDF+UB	0.929929906542056	0.9170674585046443	0.9219129479420174
RF,CV+UB	0.9056997260760111	0.8530986751941526	0.8576384286433136

2.2. Best Configuration

The configuration that resulted in the highest accuracy was the naïve bayes algorithm, using CountVectorizer as feature extractor with unigram baseline. Prior to extracting the features, the data was preprocessed by setting all the letters to lowercase, filtering out stop words, and applying stemming. Once the features were extracted, then the number of features were reduced by selecting a third of the best features using SelectKBest. Finally, when training the learning algorithm, the hyperparameters were tuned such that alpha was 0.1.

Filtering out stop words increased the accuracy because in this case, those words do not add new information to the classification of the datafiles. Since this is a classification problem, stop words aren’t that important because removing them will still give us the general idea. Similarly, applying stemming will produce the root of the words in the data. Filtering out some features increased the accuracy of the naïve bayes algorithm because some of the redundant information would have been reduced. This would have increased the probability for the correct class since irrelevant predictors are removed. The alpha hyperparameter is the additive smoothing parameter, as to prevent probabilities from becoming 0. However, given the size and number of classifications, it is unlikely for any probabilities to be 0, and thus having a lower alpha value would increase the accuracy.

3. Citations

[1] https://www.kdnuggets.com/2015/06/machine-learning-more-data-better-algorithms.html

[2] https://medium.com/axum-labs/logistic-regression-vs-support-vector-machines-svm-c335610a3d16

[3] https://towardsdatascience.com/feature-selection-for-machine-learning-in-python-filter-methods-6071c5d267d5

[4] https://medium.com/@limavallantin/why-is-removing-stop-words-not-always-a-good-idea-c8d35bd77214