AI Assignment#3 Report

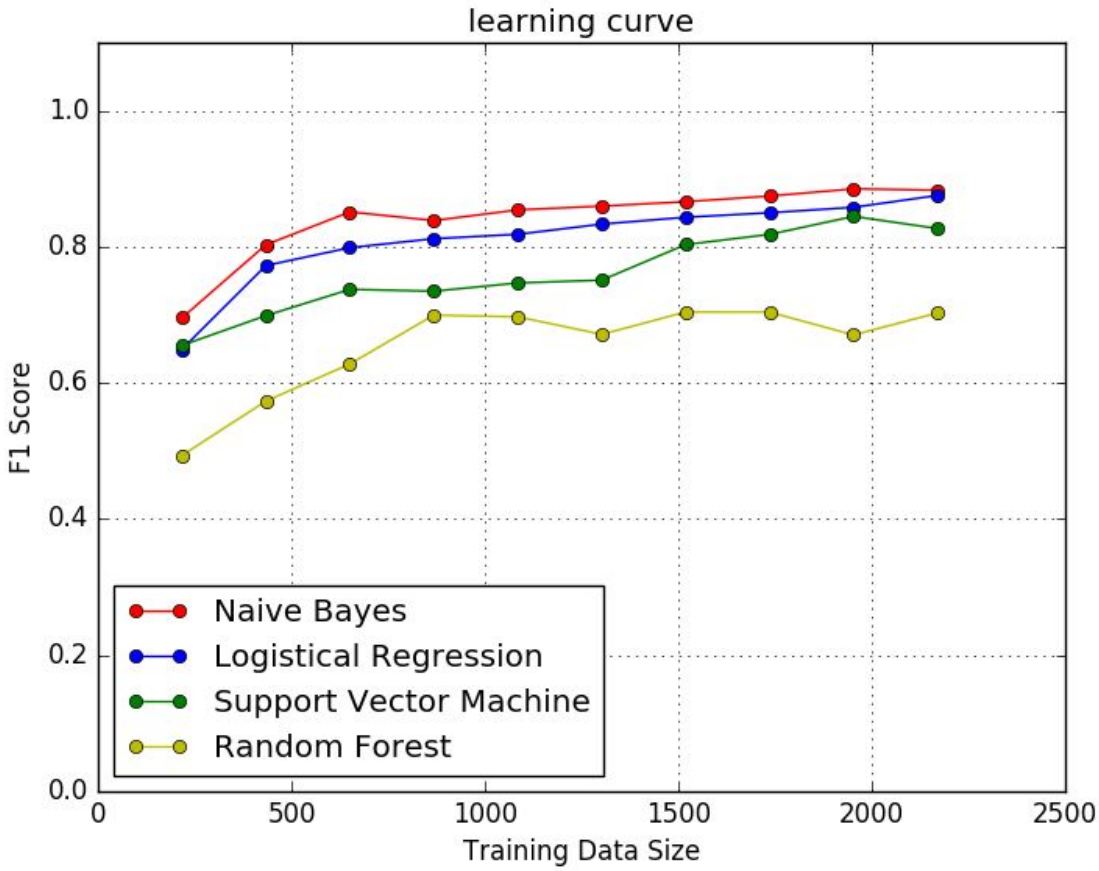
Zhichuang Sun
SBUID: 110345185

Basic Comparison with Baselines

A table of macro-average of precision/recall and F1 values

		Unigram Baseline	Bigram Baseline
Naive Bayes	Precision Score	0.92	0.89
	Recall Score	0.88	0.81
	F1 Score	0.89	0.82
Support Vector Machine	Precision Score	0.79	0.78
	Recall Score	0.79	0.75
	F1 Score	0.79	0.75
Logistic Regression	Precision Score	0.86	0.85
	Recall Score	0.85	0.81
	F1 Score	0.86	0.82
Random Forest	Precision Score	0.81	0.72
	Recall Score	0.73	0.61
	F1 Score	0.72	0.60

A learning curve result



Findings and Arguments

- Bigram Baseline is worse than unigram baseline, which is not what I expected. By inspecting the features, I find that bigram model will generate almost N^2 number of features comparing with unigram N features. That may be the problem. Because it's possible that when use bigram, number of features explode and each feature is unique, so it will be harder for the classifier to catch the real feature of a certain class.
- On either case, Naive Bayes has the best performance. Which explain it well that Naive Bayes is suitable for text classification.
- Support vector does not perform as well as expect. Maybe we need to adjust hyperparameters more carefully.
- Support Vector training needs more time than other model. While NB is fast.

My Best Configuration

The best performing classification algorithm is Naive Bayes.

Naive Bayes with Different Configuration			Select Feature	Not Select Feature
Count Vector	Stemming & Stopwords	Precision Score	0.91	**0.93**
		Recall Score	0.90	**0.90**
		F1 Score	0.90	**0.91**
	No Stemming & Stopwords	Precision Score	0.92	0.92
		Recall Score	0.90	0.88
		F1 Score	0.91	0.89
TFIDF Vector	Stemming & Stopwords	Precision Score	0.89	0.89
		Recall Score	0.80	0.74
		F1 Score	0.80	0.73
	No Stemming & Stopwords	Precision Score	0.85	0.85
		Recall Score	0.62	0.58
		F1 Score	0.59	0.54

Export the Best Configuration Training Model

To test the best configuration on new test data, please put your new_data under the directory data/ . Currently, there are two subset under data/, they are:

data/

Training/

Judge/

Test/

If you copy your new_test_data under data/ , it will looks like this:
data/
Training/
Judge/
Test/
new_test_data/

If you want to test MBC on this new data, please run
./mba.py new_test_data

Explanation

- Stemming and filter stopwords really works. It work as supposed. Stemming will reduce unnecessary extra features for same words of different forms. Filter stopwords will reduce noise and give out features of high quality
- Select features doesn't work well on NB, There are tens of thousands features, I use selectKbest to select the best 2000 features, 4000 features. And the later performs better than the former, it seems that the more features the better.
- Count Vector outperform TFIDF, it's a bit strange. It's hard to understand, TFIDF should better reflect the feature of the document, but it doesn't work well in NB model.