Report for CIS 530 Homework 2

Yixuan Meng, Xi He

September 2021

1 Basics

1.1 Result Summary

Our results for the basic models are as follows:

	Train			Dev		
Classifier	Precision	Recall	F-score	Precision	Recall	F-score
All Complex	0.433	1.000	0.604	0.418	1.000	0.590
Word-length Baseline	0.601	0.844	0.702	0.605	0.866	0.713
Word-frequency Baseline	0.566	0.816	0.668	0.557	0.844	0.671
Naive Bayes	0.495	0.980	0.658	0.469	0.969	0.632
Logistic Regression	0.725	0.658	0.690	0.727	0.694	0.710

1.2 Word Length Baseline Model

For the word-length baseline model, we have tried every integer threshold value between [0,19]. We also drew the Precision-Recall curve as shown in Fig. 1. The best threshold is 7. We can see that precision and recall are inversely related.

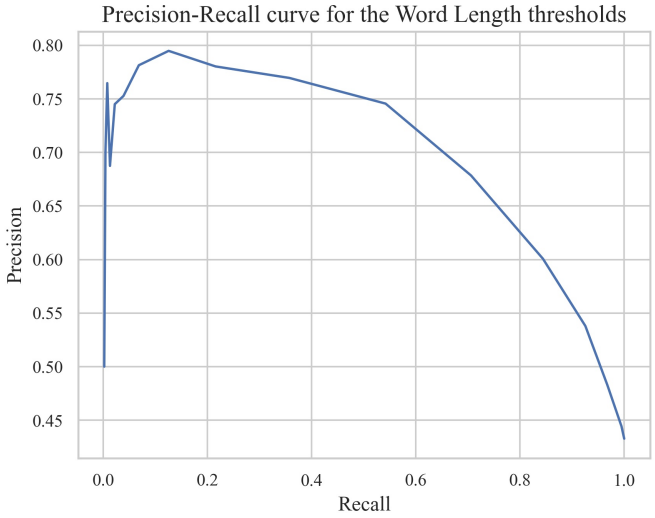

Figure 1: P-R curve for word-length baseline

1.3 Word Frequency Baseline Model

For the word-frequency baseline model, we have tried threshold value between [0,156514608] with 1000 as step, and did more fine-grained search to find the best threshold. 156514608 is the maximum word-frequency after removing top 5 percent most frequent words in the training dataset. We also drew the Precision-Recall curve as shown in Fig. 2. The best threshold is 19900171. We can see that precision and recall are inversely related.

1.4 Comparison between Word Length and Word Frequency Models

The Fig. 3 shows the Prevision-Recall curve for both word-length and word-frequency baseline models. By comparing the areas under the curve, it seems that the word-length baseline model works better on average

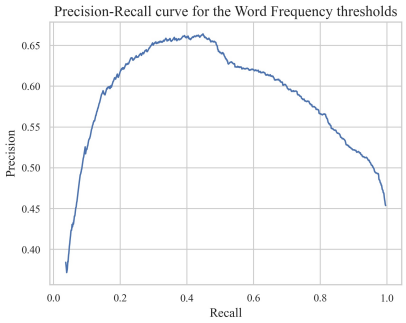

Figure 2: P-R curve for word-frequency baseline

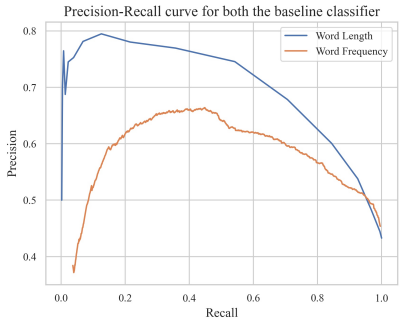

Figure 3: P-R curve for 2 baseline models

2 Classifiers

The performance of Naive Bayes and Logistic Regression models are reported in Table 1.1.
The performance of Naive Bayes model is worse than the Logistic Regression model on both the train dataset and development dataset.
The difference between the two models is caused by their different prediction mechanisms. The Naive Bayes model maximize the joint probability of input and output and assume independence between each feature while it is not the case (e.g. shorter words generally have higher frequency). In comparison, the Logistic Regression model minimizes error by modeling conditional probability, learning the mapping from input to output, it gives good performance despite correlation between some features.

3 My Own Models

3.1 Brief Introduction

In addition to the previous 2 machine learning models (Naive Bayes and Logistic Regression), we also tried 6 other machine learning models. The new models we tried are Decision Tree, SVM, Random Forest, AdaBoost, Stochastic Gradient Descent (with regularization term L1, L2, and elastic-net), Gradient Boosting and Voting Classifiers. We also applied some features and text cleaning methods in order to improve the accuracy of the prediction to previous machine learning models(Naive Bayes and Logistic Regression) and own models(new).

3.2 Data Preprocessing and Feature Engineering

Here is a list of data cleaning methods we tried:

1 Lemmatization: grouping together the inflected forms of a word using the NLTK library

2 Remove special characters using regular expression

3 Remove stopwords

3 Stemming: reducing inflected or derived words to their word stem using the NLTK library

Methods [1], [2] are applied in the final model.
Removing special characters can improve quality of sentence-based features, but removing stopwords makes simple and complex words less distinguishable by word length.

Here is a list of features we tried:

1 Word Length

2 Word Frequency

3 Number of Syllables

4 Number of word senses and synonyms

5 Sentence-based features (including length of the sentence, average word length, and average word frequency)

6 Number of vowels in the word

7 Number of word antonyms, hypernyms, and hyponyms

8 TF-IDF and Count Vectorization

9 POS-tag features: grouped POS tags into pronouns, nouns, adverbs, verbs, and adjectives

Features [1], [2], [3], [4], [5], [6] are applied in the final model.
They are selected because simple words generally have high frequencies, short lengths, less syllables. And complex words have wider range of senses and synonyms numbers.

3.3 Final Result

The final result are as follows:

	Train			Dev		
Classifier	Precision	Recall	F-score	Precision	Recall	F-score
Logistic Regression	0.73	0.693	0.711	0.581	0.931	0.716
Naive Bayes	0.547	0.957	0.696	0.353	0.057	0.099
SVM SVC	0.721	0.729	**0.725**	0.691	0.866	**0.769**
Decision Tree	1.0	0.999	1.0	0.591	0.467	0.521
Random Forest	1.0	0.999	1.0	0.65	0.28	0.391
Ada Boost	0.763	0.733	0.748	0.805	0.256	0.388
Stochastic Gradient Descent (L2)	0.707	0.773	0.739	0.572	0.935	0.71
Stochastic Gradient Descent (L1)	0.711	0.725	0.718	0.561	0.931	0.7
Stochastic Gradient Descent (elasticnet)	0.696	0.752	0.723	0.569	0.933	0.707
Gradient Boosting	0.77	0.804	0.787	0.797	0.234	0.362
VotingClassifier1 (SVM+SGD-net)	0.741	0.672	0.705	0.693	0.847	0.762
VotingClassifier2 (SVM+LR)	0.741	0.666	0.701	0.693	0.847	0.762
VotingClassifier3 (SVM+SGD-net+LR)	0.73	0.707	0.718	0.583	0.931	0.717

The SVM SVC model gives the best performance, and the F-score on the validation set is slightly higher than on the training set, therefore the model is not overfitted.
We chose SVM model as the final classifier, and tuned hyperparameters on the model.
The hyperparameter values are tested from start value to end value with specific steps, and the best choices of hyperparameters are also reported in Table 3.3.

Hyperparameter	Start	End	Step	Choice
Gamma	0.05	0.5	0.05	0.15
C (Regularization)	0.5	1.5	0.1	0.8
Kernel	'linear','poly','rbf','sigmoid'			'rbf'

After tuning the parameters, the precision, recall, and F-score on the development dataset of this model are 71.14%, 84.93%, and 77.43%.

3.4 Error Analysis

Here are some examples that our best model is good/not good at:

	Examples
True Complex	'renewable', 'conventional', 'surprisingly', 'plummeted', 'retailer', 'lilting', 'scholarship', 'evolving', 'toughest', 'unfounded'
True Simple	'hammer', 'showing', 'continues', 'exceed', 'combed', 'sled', 'sure', 'medium', 'seed', 'seat'
False Complex	'academy', 'go-ahead', 'nobody', 'spotify', 'hockey', 'overthrown', 'arguing', 'firefighter', 'praising', 'earliest'
False Simple	'coup', 'evaded', 'kinky', 'secure', 'concept', 'jolted', 'research', 'determined', 'subdued', 'groom'

From the examples above, it can be told that my model recognize complex words with few word senses (renewable, retailer etc.), and those with more vowels and are relatively long (conventional, toughest etc.). It also correctly classified simple words that are short (seed), frequently used (sure), or have many word senses (showing).

The SVM model also misclassifies certain kinds of words. Simple words with multiple vowels (academy), few word senses (spotify), multiple syllables (academy), or are relatively long (firefighter) would be misclassigied as complex. Some of short words annotated as complex would be classified as simple by the model.