IFT3335 TP2 Report

Wenhao Xu, 20150702
Mingze Li, 20150696
Yu Deng, 20151659

1. Experimental design

To deal with the word meaning disambiguation problem, we aim to classify a word used in a context into the appropriate meaning class. The implementation process mainly includes data preprocessing, analysis of multiple classification algorithms, parameter sensitivity analysis, and draws some experimental conclusions and prospects for future optimization directions.

2. Data preprocessing

Data preprocessing mainly includes four parts: data denoising, feature extraction, feature selection and data set division.

2.1 Data denoising

There may be meaningless words or punctuation marks in each sentence, so we start from punctuation marks (such as "/", ":", "\n", etc.), part of speech ('/IN', '/DT' , '/CC') and meaningless vocabulary (stoplist.txt) to denoise the original data.

2.2 Feature Extraction

Feature extraction includes these ways: Grammatical classification extraction (gc) , n-words feature extract (nw) and nominal group extract (ng).
The Grammatical classification extraction method is to obtain the part-of-speech of each n adjacent words before and after the predicted word, such as VB VBG NNS IN, and n-words feature extract is to obtain the word stems of the n adjacent words before and after the predicted word, the word stem is restored by the NLTK package, such as working->work. Nominal group extract is It is to extract the stemming words in the same normal group in "interest" as features.

2.3 Feature Selection

Since a word can appear several times in the text, and its importance may vary according to its frequency of occurrence. Therefore, consider extracting features from part-of-speech or stemming words by using CountVectorizer and TfidfVectorizer for feature selection, and using "frequency" to measure the importance of a word in the text. Among them, CountVectorizer uses the frequency of each word or part of speech in the entire training corpus as a feature, and TfidfVectorizer uses the word frequency-inverse text frequency of each word or part of speech in the entire corpus as its feature.

2.4 Dataset division

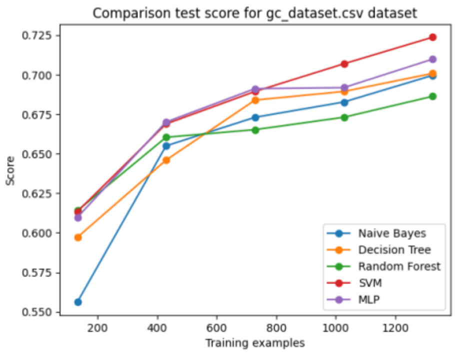
This dataset includes 2369 sentences, 6 semantics about "interest" or "interests". Detailed data can be found at: http://www.d.umn.edu/~tpederse/data.html, using the training set to train the model.

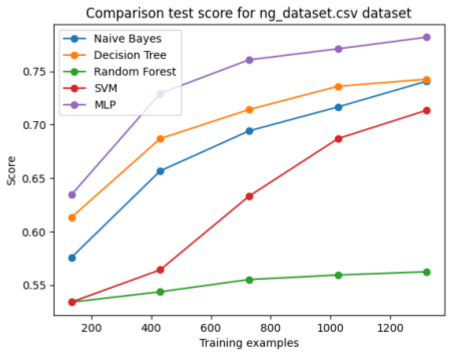
3. Performance Analysis and Comparison of Classification Algorithms

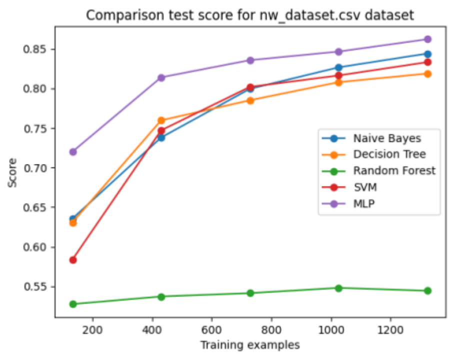
This section first makes a horizontal comparison of the performance of different classifiers under the same feature conditions, and then compares and analyzes different features and parameter settings.

Conclusion: The prediction accuracy of the mlp model is significantly better than other classification models, and the model works best when using n-words features. Therefore, in order to analyze the impact of model parameters on the effect more deeply, the following paragraphs will conduct parameter analysis based on the mlp model.
The mlp model has stronger scalability (the number of neural network layers, the number of hidden layer nodes, optimization methods, etc.), and stronger fitting capabilities, so its performance is also better.

Here are the images for comparison:







4. Parameter sensitivity analysis

4.1 Analysis of removing stop words

Removing stop words includes removing punctuation marks (such as "/", ":", "\n", etc.), parts of speech ('/IN', '/DT', '/CC') and meaningless vocabulary (stoplist .txt) has three parts, no stop words are removed, only punctuation marks and parts of speech are removed. Through experiments, if the words contained in stoplist.txt are removed, the effect of all classifiers will be reduced. We think it is because stoplist.txt contains a lot of words and names, Words related to the user's emotional color, whether to delete stop words depends largely on the tasks we are performing and the goals we want to achieve, and we are training a model that can recognize the meaning of "interest", and keep stop words make the classification effect better.

The detailed experimental training parameters and the effects of various classifiers are as follows:

(1) Remove stop words:

```
{'filename': 'gc_dataset.csv', 'n_words': 2, 'vectorizer': 'tfidf', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}}
Dataset: gc
naive_bayes Accuracy: 0.6484
decision_tree Accuracy: 0.5556
random_forest Accuracy: 0.6414
svm Accuracy: 0.6498
mlp Accuracy: 0.6174
{'filename': 'nw_dataset.csv', 'n_words': 2, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}}
Dataset: nw
naive_bayes Accuracy: 0.8368
decision_tree Accuracy: 0.8186
random_forest Accuracy: 0.5668
svm Accuracy: 0.8368
mlp Accuracy: 0.8481
{'filename': 'ng_dataset.csv', 'n_words': 2, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}}
Dataset: ng
naive_bayes Accuracy: 0.7563
decision_tree Accuracy: 0.7465
random_forest Accuracy: 0.5380
svm Accuracy: 0.7268
mlp Accuracy: 0.8113
```

(2) Do not remove stop words:

```
{'filename': 'gc_dataset.csv', 'n_words': 2, 'vectorizer': 'tfidf', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}}
Dataset: gc
naive_bayes Accuracy: 0.7004
decision_tree Accuracy: 0.7159
random_forest Accuracy: 0.7131
svm Accuracy: 0.7314
mlp Accuracy: 0.7187
{'filename': 'nw_dataset.csv', 'n_words': 2, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}}
Dataset: nw
naive_bayes Accuracy: 0.8495
decision_tree Accuracy: 0.8087
random_forest Accuracy: 0.6976
svm Accuracy: 0.8411
mlp Accuracy: 0.8481
{'filename': 'ng_dataset.csv', 'n_words': 2, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}}
Dataset: ng
naive_bayes Accuracy: 0.7606
decision_tree Accuracy: 0.7451
random_forest Accuracy: 0.5352
svm Accuracy: 0.7310
mlp Accuracy: 0.7803
```

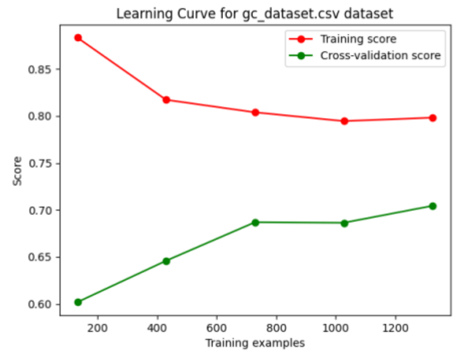
4.2 Analysis of feature extraction method

We analyze the three models one by one, let them train five algorithms respectively, and we can get conclusions through comparison.

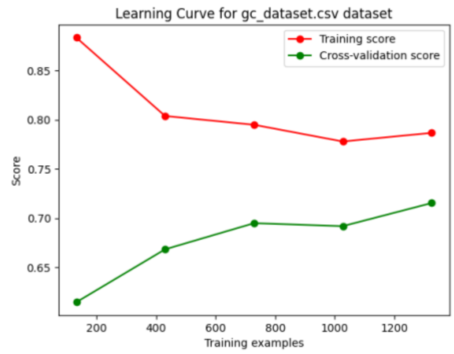
Conclusion: The model trained using stemmed word features (nw) outperforms the other two methods.
Stemming is to extract the stem or root form of a word, which can reduce the noise interference caused by tense, singular and plural, deformation, etc., and can obtain more useful information in sentences than other methods.

4.2.1 gc_dataset:

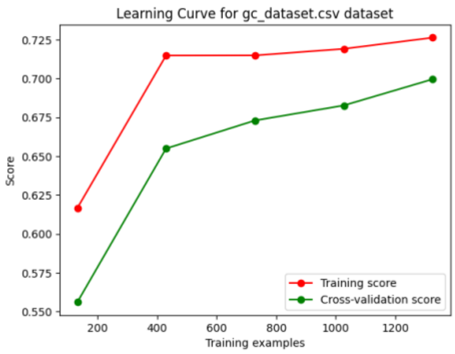
gc_dataset.csv_DecisionTreeClassifier:



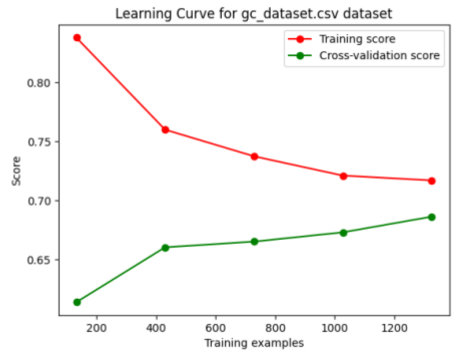
gc_dataset.csv_MLPClassifier



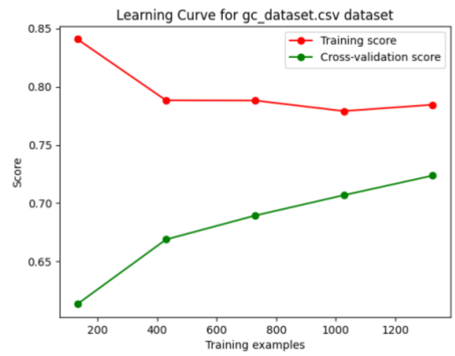
gc_dataset.csv_MultinomialNB



gc_dataset.csv_RandomForestClassifier



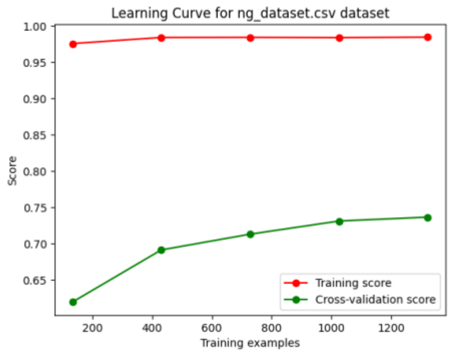
gc_dataset.csv_SVC

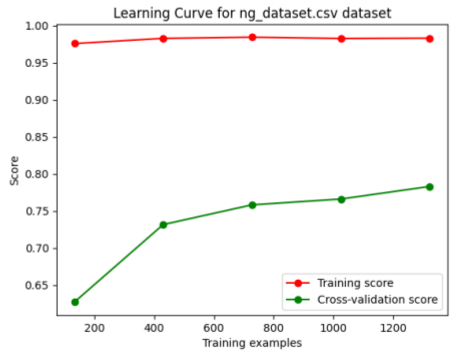


4.2.2 ng_dataset:

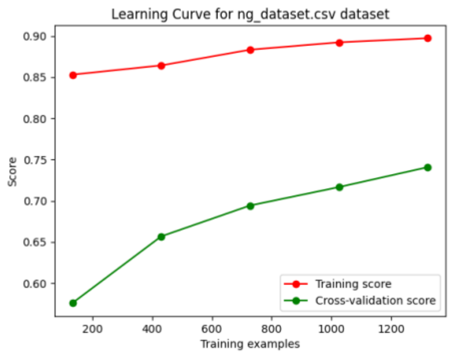
ng_dataset.csv_DecisionTreeClassifier.

ng_dataset.csv_MLPClassifier

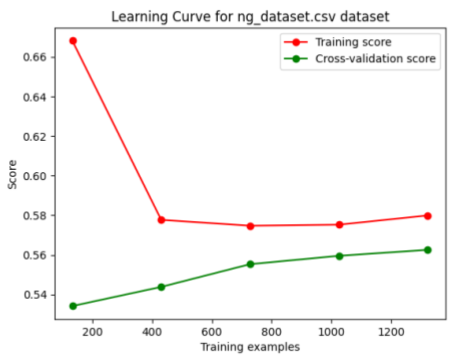




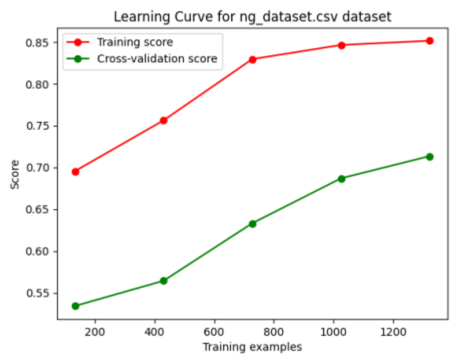
ng_dataset.csv_MultinomialNB



ng_dataset.csv_RandomForestClassifier

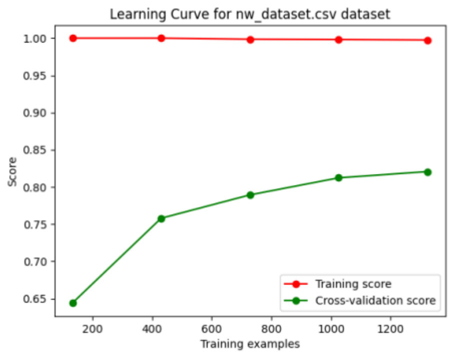


ng_dataset.csv_SVC

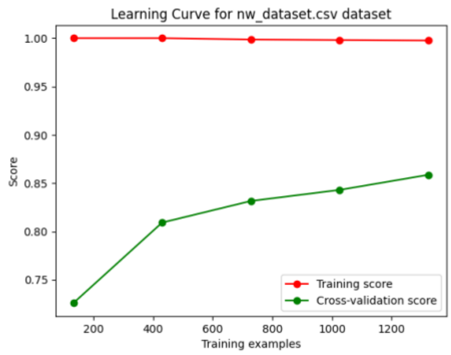


4.2.3 nw_dataset:

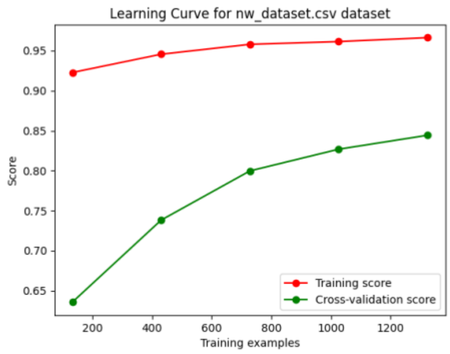
nw_dataset.csv_DecisionTreeClassifier



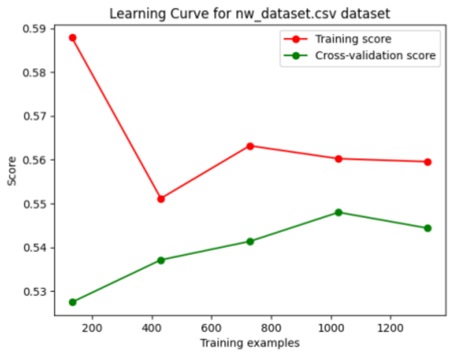
nw_dataset.csv_MLPClassifier



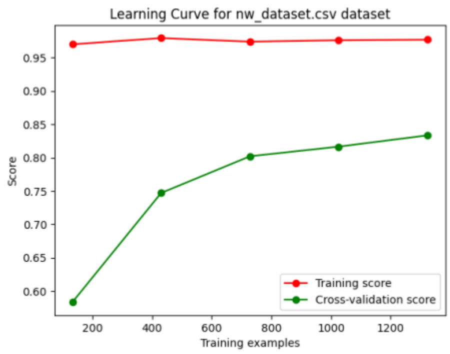
nw_dataset.csv_MultinomialNB



nw_dataset.csv_RandomForestClassifier



nw_dataset.csv_SVC



4.3 Window size

Conclusion: Among decision_tree, random_forest, svm, and mlp, we choose the mlp method that has been determined and has the best rendering effect to calculate the performance. We get the best results when the window is 2.

At the same time, we consult Levy & Goldberg's paper "Dependency-based word embeddings" to understand the qualitative impact of window size (https://levyomer.files.wordpress.com/2014/04/dependency-based-word-embeddings-acl-2014.pdf), they found that larger windows tend to capture more information about topics and domains, and smaller windows tend to capture more information about the word itself, which further demonstrates that we want to train a machine that can recognize "interest "The goal of meaning models is better suited for smaller windows. The detailed experimental results are as follows:

{'filename': 'nw_dataset.csv', 'n_words': 1, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100),'activation': 'tanh'}, 'dt': {'depth': 500}} Dataset: nw

mlp Accuracy: 0.8242

{'filename': 'nw_dataset.csv', 'n_words': 2, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}} Dataset: nw

mlp Accuracy: 0.8467

{'filename': 'nw_dataset.csv', 'n_words': 3, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}} Dataset: nw

mlp Accuracy: 0.8425

{'filename': 'nw_dataset.csv', 'n_words': 4, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}} Dataset: nw

mlp Accuracy: 0.8397

{'filename': 'nw_dataset.csv', 'n_words': 5, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}} Dataset: nw

mlp Accuracy: 0.8326

4.4 Analysis of Feature Selection Method

TfidfVectorizer and CountVectorizer.

Conclusion: TfidfVectorizer works better than CountVectorizer on most classifiers.

CountVectorizer only considers the frequency of vocabulary in the text, which belongs to the bag of words model feature, while TfidfVectorizer not only considers the frequency of a certain vocabulary in the text, but also pays attention to the number of all texts containing this vocabulary. It can reduce the impact of high-frequency meaningless words and get more meaningful features.

The experimental training parameters and the effects of various classifiers are as follows:

(1) tf-idf

{'filename': 'nw_dataset.csv', 'n_words': 2, 'vectorizer': 'tfidf', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}} Dataset: nw

naive_bayes Accuracy: 0.7014
decision_tree Accuracy: 0.7283
random_forest Accuracy: 0.7633
svm Accuracy: 0.9167
mlp Accuracy: 0.9353

(2) count

{'filename': 'nw_dataset.csv', 'n_words': 2, 'vectorizer': 'count', 'mlp': {'solver': 'adam', 'hidden_layer_sizes': (50, 100), 'activation': 'tanh'}, 'dt': {'depth': 500}} Dataset: nw

naive_bayes Accuracy: 0.6801

decision_tree Accuracy: 0.7223

random_forest Accuracy: 0.7956
svm Accuracy: 0.8909
mlp Accuracy: 0.9298

4.5 Classifier parameters

Perform parameter sensitivity analysis on the best mlp model, including three parts: network layer number, optimizer and activation function.

Although the following parameters do not make much difference to the results of the test methods, there are only some minor differences. But some conclusions can also be drawn:

4.5.1 Number of Network Layers

Conclusion: The more hidden nodes in the network to a certain extent, the more layers, the better the fitting effect of the model.

4.5.2 Optimizer

Conclusion: adam is slightly better than sgd and lbfgs, and adam is better than sgd in terms of calculation speed.

4.5.3 Activation function

Conclusion: tanh is better than relu and logistic

5. Conclusions and expectations

Combining the above, it can be concluded that through the preprocessing of the data set, the analysis of various classification algorithms, and the sensitivity analysis of parameters, we found that the mlp model has the best and stable effect, and found that data processing has a greater impact on the subsequent model results. In the future, we will try more novel data processing, feature processing, and model experiments to discover more meaningful things.