Naive Bayes Text Classification System

Wang Xintao, DC228741

1 Introduction

This project implements a text classification system based on the Naive Bayes algorithm, which is mainly used for document preprocessing, feature selection, probability calculation, classification, and performance evaluation. With this system, the complete process from data preprocessing to the evaluation of classification results can be accomplished, providing an efficient classification tool for natural language processing (NLP) tasks.

This project is divided into the following six main modules:

1. **Data Preprocessing Module**: Cleaning, tokenization, and stemming of raw data.
2. **Word Frequency Statistics Module**: Counts the frequency of occurrence of each word in different categories in the corpus.
3. **Feature Selection Module**: Selects the most important feature words according to word frequency.
4. **Probability Calculation Module**: Calculates the prior probability of each category and the posterior probability of feature words.
5. **Classification Module**: Classifies the test set based on the Naive Bayes algorithm.
6. **Performance Evaluation Module**: Evaluates the performance of the classification model using the F1 score.

With this project, we have realized a complete process from data preprocessing to classification result evaluation, which provides an efficient solution for text categorization tasks.

2 Description

2.1 Data Preprocessing Module

The main function of this module is to clean and standardize the raw data, including the following steps:

- **Remove useless symbols**: Remove punctuation marks and special characters.
- **Convert to lowercase**: Convert all letters to lowercase and standardize the format.
- **Tokenization**: Use `nltk.word_tokenize` to tokenize the text.
- **Stemming**: Use `nltk.PorterStemmer` to extract the stems of words.

The processed data is stored in JSON format and contains document ID, category, and processed text content. Below is an example:

```
{
"file_id": "training/7984",
"category": "acq",
"text": "bei ltbeih acquir iveyrowton and associ bei
hold ltd said it acquir iveyrowton and associ a nashvil
tennbas bank market firm term were not disclos"
}
```

2.2 Word Frequency Statistics Module

This module counts the frequency of occurrence of each word in different categories and outputs it as a text file. The specific steps are as follows:

1. Iterate through the preprocessed documents and count the number of documents in each category.
2. Count the number of occurrences of each word in different categories.
3. Output the statistics in the following format:

```
 359 428 535 1617 2848
shad 0 0 0 10 0
see 44 32 60 43 252
progress 1 7 6 47 26
on 528 431 770 1185 1110
insid 1 1 3 22 4
trade 81 211 450 150 107
secur 33 4 111 414 129
and 1461 1536 1684 4214 3183
......
```

2.3 Feature Selection Module

This module selects the most important feature words according to the word frequency in the following steps:

1. Read the word frequency statistics and calculate the total frequency of each word.
2. Sort the words in descending order according to the total frequency and select the first n words as feature words. In this project, $n = 10000$.
3. Output the feature words and their frequencies in each category.

```
359 428 535 1617 2848
the 4099 4464 6642 9863 6305
of 1896 2115 2656 6566 5098
to 2256 2399 2946 5842 3528
in 1729 1529 2169 3771 3741
......
```

2.4 Probability Calculation Module

This module calculates the prior and posterior probabilities required for classification:

1. **Prior Probability**: The prior probability of each category c is calculated as:

   $$P(c) = \frac{N_c}{N}$$

   where:

   - N_c: Number of documents in category c.
   - N: Total number of documents.

2. **Posterior Probability (with Laplace Smoothing)**: The posterior probability of each feature word w in category c is calculated using Laplace smoothing as:

   $$P(w|c) = \frac{\text{count}(w, c) + 1}{\sum_{w' \in V} \text{count}(w', c) + |V|}$$

   where:

   - $\text{count}(w, c)$: Frequency of word w in category c.
   - $\sum_{w' \in V} \text{count}(w', c)$: Total frequency of all words in category c.
   - $|V|$: Vocabulary size (total number of unique feature words).

   Laplace smoothing ensures that probabilities are non-zero, even for words that do not appear in a specific category.

The output format is as follows:

```
 0.06203559702782098 0.07395887333678935
0.0924485916709867 0.279419388284085 0.492137549680318
the 11.41782729805014 10.429906542056075
12.414953271028038 6.0995670995671 2.213834269662921
of 5.281337047353761 4.941588785046729
4.964485981308411 4.0606060606060606 1.7900280898876404
to 6.284122562674095 5.605140186915888
5.506542056074767 3.6128633271490416 1.2387640449438202
in 4.816155988857939 3.572429906542056
4.054205607476636 2.3320964749536177 1.3135533707865168
said 4.194986072423398 3.2616822429906542
3.499065420560748 2.9696969696969697 0.9736657303370787
and 4.069637883008356 3.588785046728972
3.147663551401869 2.6060606060606060 1.117626404494382
a 3.9275766016713094 2.630841121495327
3.2803738317757007 2.697588126159555 1.0916432584269662
...
```

2.5 Classification Module

This module classifies test documents using the Naive Bayes algorithm. The probability of a document d belonging to category c is calculated as:

$$P(c|d) \propto P(c) \prod_{w \in d} P(w|c)$$

where:

- $P(c|d)$: Probability of document d belonging to category c.
- $P(c)$: Prior probability of category c.
- $P(w|c)$: Posterior probability of word w in category c.
- $w \in d$: Words in document d.

The predicted category $\hat{c}$ for document d is:

$$\hat{c} = \arg\max_{c} P(c|d)$$

The output format is as follows:

```
 test/14975 earn
test/21067 crude
test/20081 money-fx
test/21040 acq
test/16228 earn
test/18689 crude
test/21462 crude
test/16833 crude
test/15255 crude
......
```

2.6 Performance Evaluation Module

This module evaluates the classification model using the F1 score. The following metrics are calculated:

1. **Precision**:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

2. **Recall**:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

3. **F1 Score**:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

4. **Average F1 Score**:

$$\text{Average } F_1 = \frac{1}{C} \sum_{c=1}^{C} F_1(c)$$

where C is the total number of categories.The final calculation shows that:he F1 score of the classification result is: 0.611130110064772

3 Conclusion

This project successfully implements a Naive Bayes-based text classification system, achieving the following:

1. Preprocessed raw text data into structured format.
2. Counted word frequencies and selected important features.
3. Calculated prior and posterior probabilities for classification.
4. Classified test documents and evaluated performance using the F1 score.

Key Learnings:

- Learned text preprocessing techniques, including cleaning, tokenization, and stemming.
- Gained experience in feature selection and probability calculation.
- Understood the Naive Bayes algorithm and its application in text classification.
- Learned to evaluate classification models using the F1 score.

Future Work:

- Improve preprocessing by adding stopword removal.
- Explore advanced feature selection methods, such as information gain.
- Extend the system to support multilingual text classification.
- Optimize the implementation for better efficiency.

This project provides a complete workflow for text classification and lays the foundation for future NLP tasks.