Discrete Mathematics Programming Assignment 3

21500185 PyeongGang Kim
21500359 SueGeun Song
21500539 InSeok Lee
21700163 JooHyun Kim
21800727 SaeAm Choi

1. Introduction

summary

The sentiment of text we used are negative or non-negative. Each word has influence to the text classification. Predict the sentiment of input text is our purpose.To solve the problem, predict whether the message is Negative or Non-Negative, and train the frequency with which certain words in the message take up the entire dataset.

Problem analysis

The main purpose is to predict that the new message will be 'negative' or 'non-negative'. To understand this, apply 'Bayes' theorem' that posterior probabilities can be calculated with prior probabilities.

Based on the spam filter problem, the prior probability is the probability that a word constituting each message is negative or non-negative, and the posterior probability is the probability that a new message is negative or non-negative.

Whether each word is negative or non-negative is determined by the existing specified data. The nature of a word is determined according to how many specific words are included in a message determined to be negative or non-negative.

The probability that the word A is included in the negative message is the negative probability of A, and the probability that the word A is included in the non-negative message is the non-negative probability of A.

Since message is made up of words, the elements that determine the negative and non-negative probability of message are the words that make up the message. When words are used frequently in fraudulent messages, they become negative, and when they are used frequently in positive messages, they become positive.

Assuming that the case where each word appears in the message is independent of each other, the nature of the message itself (negative / positive) is determined by the product of the probability that each word is negative.

Therefore, negative and non-negative messages are predicted depending on which product of negative probabilities each word has.

2. Approach

Top-down description of solution

The program can be broadly divided into a training process for obtaining the prior probability and a process for implementing a predictor for obtaining the posterior probability.

First, dividing the message into words, and performing preprocessing such as lowercase and redundant case. After that, using stemmers, even if the word is made in a circle, words that appear in a small number (such as the name of a person) (be verbs, surveys, etc.) are excluded from the target for which the probability is calculated. Generalized and stores words in the hash table with frequency of appearance.

The following is counting which is the most important operation. Under the assumption that the more frequently a word appears in a negative message, the higher the negative probability, the negative and non-negative probabilities of each word are calculated using a hash table in which the frequency of the word is recorded in advance. As a result, two hash tables are generated in which negative probabilities and non-negative probabilities are recorded together with words. In order to access each probability, another hash table having a word as value and

an ordering number as key is created so that a table in which two probabilities are recorded can be accessed through one table. Save to model.csv with negative and non-negative probabilities assigned to each word through this process.

After that, from model.csv, create a hash table that includes information about bringing negative and non-negative probability information. Key is word and value is probability. Proceed with logarithm rescaling so that the probability of the table can be easily calculated by re-tokenizing the test file to determine whether it is negative.

Finally apply bayes' theorem and calculate each probability and find the ideal reference value that distinguishes negative and non-negative by repeating the above attempts many times.

3. Result

The smoothing value was set to 5 and the threshold value was set to 0.964. The accuracy was 83% for test.negative.csv and 84% for test.non-negative.csv.

4. Discussion

What number do we need for division is our primary discussion topic. Whether message number or word number. The given message number is our choice. Because our problem is about perspective of message, not perspective of word.

Estimating Threshold value is too hard to proceed project. A little bit larger than before value influences the accuracy of program. Also, less. For the proper value, we test many time, finally, get the optimal value.

The phrase, dialect and Nuance are hard to detect in sentence. The special meaning of Advanced english like double negative is hard to judge whether negative or non-negative.

The problem of Smoothing. Smoothing is the value that helps to predict probability of sentence. In our experience, if we grow up the number, the accuracy of our prediction is getting rower and even break up.

g_hash_table is very good for construct table and solve this problem. It's so beautiful that we can use function in table. When initiating the table, we make the function we want to use. Not only in Java, but also in C, hash table is useful.