NBC project

Name: 이세형

Student Number: 2020-312145

1. Implementation:

An introduction to the functions implemented in this project is as follows.

- **get_chunk(inputfile):** This function separates the input CSV file by each item.
- **filtering(text):** For the given text area, this function removes special characters and stop words, separates it into words, and returns it in the form of a list.
- **tokenize(chunks, percentage, debug):** Preprocessing is performed on each item of the CSV file according to the assignment requirements. During this process, the 'filtering' function is used.
- **get_features(tokens, total_cnt):** Based on the preprocessing results, the 1000 most frequent words are extracted as feature vectors.
- **featuring(tokens, features):** Based on the extracted feature vectors, filtering is performed again on each item of the CSV file. In other words, only the words corresponding to the feature vectors are retained in the text area.
- **calculate_prior(tokens):** Based on the preprocessed data, the prior probabilities for each class (positive or negative) are calculated.
- **calculate_likelihood(tokens, features):** The likelihood probabilities are calculated based on the preprocessed data and feature vectors. Laplace smoothing is applied during this process.
- **predict(words, prior, likelihood):** Using the given words, along with the previously calculated prior probabilities and likelihood probabilities, class prediction is performed.
- **calculate_accuracy(tokens, prior, likelihood):** For the given test data, class predictions are compared with the actual labels to determine the overall accuracy.

(For the sake of challenge, this project was implemented purely without using any external libraries or even built-in libraries such as 'csv' and 'math'. Only the 'matplotlib' library was imported for the purpose of plotting graphs.)

2. How does NBC learn and predict?

It is assumed that the train and test data sets have been appropriately preprocessed.

According to Bayes' theorem, the following equation holds for feature vector x and class C_k.

$$P(C_k|x) = \frac{P(C_k) \times P(x|C_k)}{P(x)} \quad \Rightarrow \quad P(C_k|x) \propto P(C_k) \times P(x|C_k)$$

It is assumed that each feature x_i is independent of the others. In other words, the occurrence of the word 'good' and the occurrence of the word 'bad' in a text area are considered independent events that do not affect each other. Additionally, the probability of the evidence is independent of the class, meaning it is the same for all classes, and therefore it is not considered during the training process.

Accordingly, the process of calculating prior probabilities, likelihood probabilities, and making predictions is as follows.

a. **Prior probability:**

The prior probability, that is, the probability of the class occurring, is calculated as follows.

$$P(positive) = \frac{\#\ of\ positive\ ratings}{\#\ of\ total\ ratings}, \quad P(negative) = \frac{\#\ of\ negative\ ratings}{\#\ of\ total\ ratings}$$

b. **Likelihood probability:**

The likelihood probability is calculated as follows. Laplace smoothing was applied in this calculation.

$$\mathrm{P}(x_i|C_k) = \frac{frequency\ of\ word\ x_i\ in\ C_k\ ratings + 1}{(\#\ of\ total\ words\ in\ C_k\ ratings) + (\#\ of\ features)}$$

c. **Predict:**

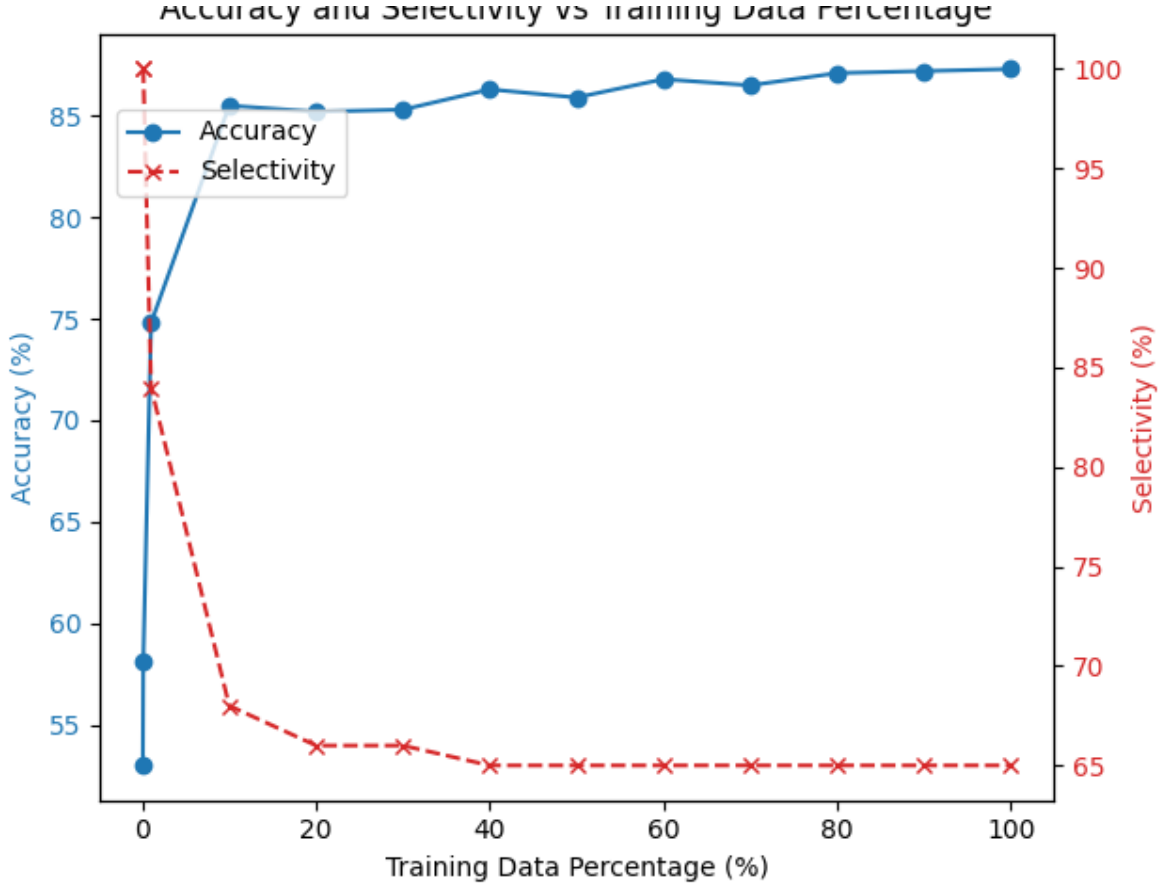
The prior probabilities and likelihood probabilities calculated from the training data are used to compute the posterior probabilities for each class. The class with the higher posterior probability is then predicted.

$$P(C_k|x) \propto P(C_k) \cdot \prod_{i=1}^{n} P(x_i|C_k) \quad (k = 1, 2)$$

cf) Since the log function increases very steeply in the range (0, 1), it is advisable to calculate probabilities using the log function when programming this process. This prevents underflow problems that can occur when calculations are performed without the log function. However, in this project, log function was not used.

3. Result analysis:

The experimental results for different training data percentages (0.05%, 0.1%, 1%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100%) are as follows.



When using 1% of the training data, the accuracy exceeded 70%, and when using 10%, it already surpassed 80%. This indicates that the accuracy increased sharply when the feature vector selection ratio (a concept used in this project for convenience, not an official term, with the calculation

method explained below) dropped below 80%. Additionally, from 10% and above, there was no significant difference in accuracy and performance.

∴ This demonstrates that NBC can achieve good performance with a small amount of data, provided the training data has appropriate characteristics.

(cf. selectivity = $\frac{sum\ of\ total\ frequency\ of\ feature\ words}{\#\ of\ total\ words\ in\ data\ set\ (before\ preprocessing)}$)