HOMEWORK 6: LEARNING TO CLASSIFY TEXT

In this exercise, you are given a dataset containing 20,000 news articles and categorized into 20 newsgroups, as described in **Table 1**. Then, you need to apply the Naïve Bayes algorithm to learn a classifier to predict the label for a new instance.

I. Dataset

Table 1. List of labels of newsgroup dataset

Labels	*Labels*	*Labels*
alt.atheism	rec.autos	sci.crypt
comp.graphics	rec.motorcycles	sci.electronics
comp.os.ms-windows.misc	rec.sport.baseball	sci.med
comp.sys.ibm.pc.hardware	rec.sport.hockey	sci.space
comp.sys.mac.hardware	talk.politics.guns	soc.religion.christian
comp.windows.x	talk.politics.mideast	
misc.forsale	talk.politics.misc	
	talk.religion.misc	

II. Requirements

The algorithm uses a Naïve Bayes classifier together with the assumption that the probability of word occurrence is independent of position within the text. Notice the algorithm is quite simple. During learning, the procedure `LEARN_NAIVE_BAYES_TEXT` examines all training documents to extract the vocabulary of all words and tokens that appear in the text, then counts their frequencies among the different target classes to obtain the necessary probability estimates. Later, given a new document to be classified, the procedure `CLASSIFY_NAIVE_BAYES_TEXT` uses these probability estimates to calculate v_{NB} values. Note that any words appearing in the new document that were not observed in the training set are simply ignored by `CLASSIFY_NAIVE_BAYES_TEXT`.

With the given dataset, you are required to program a tool to learn the classifier based on Naive Bayes algorithm for this dataset. Specifically, you need to do the following tasks:

a. Understand the dataset before processing it.

b. To train a classifier, you need to split the dataset into two parts, i.e., training set and test set. For example, you can take 80% of the dataset to construct the training set, and the other will be the test set. Besides that, you also need to consider the case of shuffling the dataset before splitting it.

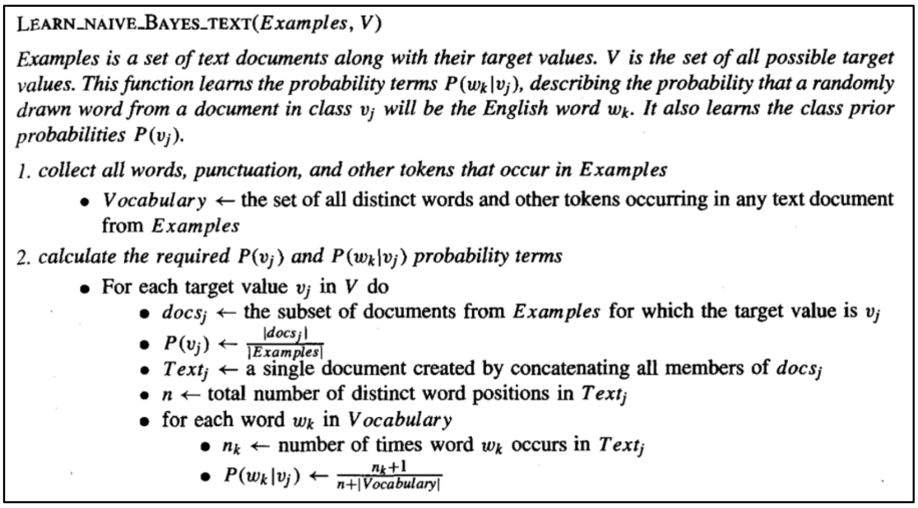
c. Report the experimental results in the following format:

No.	*Splitting ratio*	*# Shuffles*	*Accuracy*
1	80%	1	A_1
2	80%	2	A_2
3	80%	3	A_3
		...	
N	50%	$i \in \{1,2,3\}$	A_N

--- THE END ---

APENDIX A: NAÏVE BAYES ALGORITHM

Naive Bayes algorithms for learning and classifying text. In addition to the usual Naïve Bayes assumptions, these algorithms assume the probability of a word occurring is independent of its position within the text.

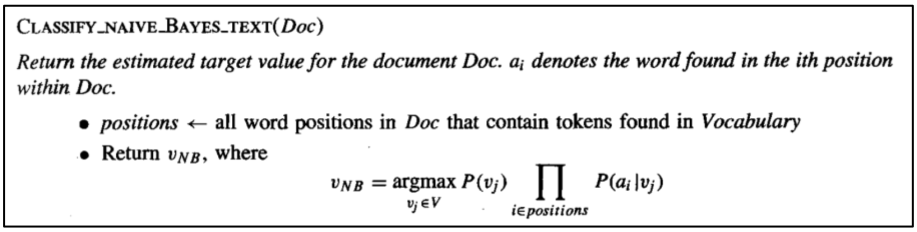


```
LEARN_NAIVE_BAYES_TEXT($Examples, V$)

Examples is a set of text documents along with their target values. $V$ is the set of all possible target
values. This function learns the probability terms $P(w_k|v_j)$, describing the probability that a randomly
drawn word from a document in class $v_j$ will be the English word $w_k$. It also learns the class prior
probabilities $P(v_j)$.

1. collect all words, punctuation, and other tokens that occur in $Examples$
    • $Vocabulary \leftarrow$ the set of all distinct words and other tokens occurring in any text document
      from $Examples$
2. calculate the required $P(v_j)$ and $P(w_k|v_j)$ probability terms
    • For each target value $v_j$ in $V$ do
        • $docs_j \leftarrow$ the subset of documents from $Examples$ for which the target value is $v_j$
        • $P(v_j) \leftarrow \frac{|docs_j|}{|Examples|}$
        • $Text_j \leftarrow$ a single document created by concatenating all members of $docs_j$
        • $n \leftarrow$ total number of distinct word positions in $Text_j$
        • for each word $w_k$ in $Vocabulary$
            • $n_k \leftarrow$ number of times word $w_k$ occurs in $Text_j$
            • $P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$
```

Figure 1 Pseudo code of LEARN_NAIVE_BAYES_TEXT



```
CLASSIFY_NAIVE_BAYES_TEXT($Doc$)

Return the estimated target value for the document Doc. $a_i$ denotes the word found in the ith position
within Doc.

    • $positions \leftarrow$ all word positions in $Doc$ that contain tokens found in $Vocabulary$
    • Return $v_{NB}$, where
        $v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_{i \in positions} P(a_i|v_j)$
```

Figure 2 Pseudo code of CLASSIFY_NAIVE_BAYES_TEXT