CS6041 HW2, Noah Gardner, 000843905

1 Assignment 2

1. You are given sample documents with corresponding classes. Documents are annotated as A, or B. These documents are splitted as training and testing set shown in the below table. Assign the most probable class to the test sentence given below using naïve bayes classification approach. Please mention each step clearly (i.e., prior probability, conditional probability, etc.). **[Points 25]**

$$P(c|X) = \frac{P(X|c)P(c)}{P(X)} \tag{1}$$

$c(prior)$	$P(c)$
A	$P(A) = 8/11 = 0.72$
B	$P(B) = 3/11 = 0.27$
$x(predictor)$	$P(X)$
$Chinese$	$P(Chinese) = 6/11 = 0.55$
$Japan$	$P(Japan) = 1/11 = 0.27$
$Tokyo$	$P(Tokyo) = 1/11 = 0.27$
$X|c(likelihood)$	$P(X|c)$
$Chinese|A$	$P(Chinese|A) = 1/3 = 0.33$
$Japan|A$	$P(Japan|A) = 1/3 = 0.33$
$Tokyo|A$	$P(Tokyo|A) = 1/3 = 0.33$
$Chinese|B$	$P(Chinese|B) = 5/8 = 0.625$
$Japan|B$	$P(Japan|B) = 0/8 = 0.0$
$Tokyo|B$	$P(Tokyo|B) = 0/8 = 0.0$
$probability$	$P(c|X)$
$A|Chinese$	$(0.33) * (0.72)/(0.55) = 0.432$
$A|Japan$	$(0.33) * (0.72)/(0.27) = 0.88$
$A|Tokyo$	$(0.33) * (0.72)/(0.27) = 0.88$
$B|Chinese$	$(0.625) * (0.27)/(0.55) = 0.307$
$B|Japan$	$(0.0) * (0.27)/(0.27) = 0.0$
$B|Tokyo$	$(0.0) * (0.27)/(0.27) = 0.0$

Figure 1: Naive Bayes Classification calculations based on input training set.

Answer Since there are no instances of *Japan* or *Tokyo* in the training set for class B, the output of the model will necessarily be 0% for class B. Therefore, the class of the input text will be classified as A.

There exist some methods that can handle the problem of 0% likelihood for Naive Bayes classifiers. One such method is to ignore any words that did not appear in the training set. If this method is applied to the input text, the probability of each class is calculated as follows:

$$\begin{aligned} P(A|'test') &= P(A|Chinese)^3 * P(A|Japan) * P(A|Tokyo) \\ &= 0.432^3 * 0.88 * 0.88 = 0.0624 \end{aligned} \tag{2}$$

$$\begin{aligned} P(B|'test') &= P(B|Chinese)^3 \\ &= 0.307^3 = 0.0289 \end{aligned} \tag{3}$$

The probability that the test text is higher in class A than in class B. Therefore, the class of the input text will be classified as A.

2. What is cross-validation? Would you please give an example and explain how cross-validation work? When to use cross-validation during an experiment? [**Points 15**]

Answer Cross-validation is a technique that splits the data into different sections (or *folds*) and tests the algorithm on each fold. This method allows one to prevent overfitting by using different data for each training set. Then, after training the algorithm once on each fold, the models are compared to see which one performs better or can also be average to evaluate the performance of the algorithm.

An example of cross-validation is *k-fold cross-validation*. For example, if examine an algorithm using 2-fold cross-validation, we would split the dataset into the training and testing sets. Then, we would split the training set into two folds. We would train the algorihtm twice (once on each fold) and use the fold that wasn't used for training for testing or validation.

3. Explain how gradient descent algorithm works? Please explain the effect of learning rate on the learning algorithm while updating parameters using the following equations. Please show differences among different types of gradient descent – mini-batch, batch, and stochastic gradient descent. [**Points 15**]

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \frac{d}{dw} f(x, \mathbf{w}) \tag{4}$$

Answer The goal of the gradient descent algorithm is to find a valley in the function. It is often compared to climbing down a mountain. If one was climbing down a mountain, they would not want to climb past the lowest point. Instead, their goal would be to move as much as possible towards the bottom (to reach the bottom as soon as possible), without overshooting (which would require the climber to climb back down).

The 'rate' at which our climber climbs down the mountain is similar to the learning rate (η) in the gradient descent algorithm. We can increase the learning rate to make it to the bottom of the expression (*minima*, a possible solution for the equation). $\mathbf{w}^{(t)}$ represents the current value of the parameter and $\mathbf{w}^{(t+1)}$ represents the new value of the parameter after the next update step. The parameter

$\frac{d}{dw}f(x, \mathbf{w})$ is the derivative of the function $f(x, \mathbf{w})$ with respect to the parameter w, which gives us the slope of loss at the current point.

There are different methods for applying the gradient descent algorithm, including mini-batch, batch, and stochastic gradient descent. Sotchastic gradient descent is an iterative method which updates the gradient at each step. The batch gradient descent is a method that updates the gradient at the end of each epoch by using a *batch* of the data (usually the whole dataset). Mini-batch gradient descent is a method that is derived from batch gradient descent and uses smaller batches of data to update the gradient in fewer steps than stochastic gradient descent but more steps than batch gradient descent.

4. You are given the following text documents. Please answer the following questions. [**Points 45**]

   Text:

   It is going to rain today.

   Today I am not going outside.

   NLP is an interesting topic.

   NLP includes ML, DL topics too.

   I am going to complete NLP homework, today.

   (a) Would you please calculate the TF-IDF vector for each of the tokens? Please show each step (i.e., tokenization, vocabulary, TF, IDF, etc.). [**Points 15**]

   Answer:

   Step 1: Split into tokens, remove punctuation, and set tokens to lower-case.

```
[['it', 'is', 'going', 'to', 'rain', 'today'],
['today', 'i', 'am', 'not', 'going', 'outside'],
['nlp', 'is', 'an', 'interesting', 'topic'],
['nlp', 'includes', 'ml', 'dl', 'topics', 'too'],
['i', 'am', 'not', 'going', 'to', 'play', 'football', 'today']]
```

   Step 2: Create a vocabulary from the tokens (I use a sorted vocabulary here).

```
['am', 'an', 'dl', 'football', 'going', 'i',
'includes', 'interesting', 'is', 'it', 'ml',
'nlp', 'not', 'outside', 'play', 'rain', 'to',
'today', 'too', 'topic', 'topics']
```

   Step 3: Create a TF vector for each token.

   (b) Please calculate term-term co-occurrence matrix with a given context window ± 3. [**Points 15**]

   (c) Please find the most similar words (a pair of words) from their vector representations for both TF-IDF and co-occurrence matrix based vector representation cases. To compute most similarity score, you may use cosine similarity score. Show details similarity calculation for both cases – (a) TF-IDF and (b) co-occurrences-based representations. [**Points 15**]

	tf_d0	tf_d1	tf_d2	tf_d3	tf_d4	idf	tf_idf_d0	tf_idf_d1	tf_idf_d2	tf_idf_d3	tf_idf_
am	0	1	0	0	1	0.398	0.0	0.398	0.0	0.0	0.398
an	0	0	1	0	0	0.699	0.0	0.0	0.699	0.0	0.0
dl	0	0	0	1	0	0.699	0.0	0.0	0.0	0.699	0.0
football	0	0	0	0	1	0.699	0.0	0.0	0.0	0.0	0.699
going	1	1	0	0	1	0.222	0.222	0.222	0.0	0.0	0.222
i	0	1	0	0	1	0.398	0.0	0.398	0.0	0.0	0.398
includes	0	0	0	1	0	0.699	0.0	0.0	0.0	0.699	0.0
interesting	0	0	1	0	0	0.699	0.0	0.0	0.699	0.0	0.0
is	1	0	1	0	0	0.398	0.398	0.0	0.398	0.0	0.0
it	1	0	0	0	0	0.699	0.699	0.0	0.0	0.0	0.0
ml	0	0	0	1	0	0.699	0.0	0.0	0.0	0.699	0.0
nlp	0	0	1	1	0	0.398	0.0	0.0	0.398	0.398	0.0
not	0	1	0	0	1	0.398	0.0	0.398	0.0	0.0	0.398
outside	0	1	0	0	0	0.699	0.0	0.699	0.0	0.0	0.0
play	0	0	0	0	1	0.699	0.0	0.0	0.0	0.0	0.699
rain	1	0	0	0	0	0.699	0.699	0.0	0.0	0.0	0.0
to	1	0	0	0	1	0.398	0.398	0.0	0.0	0.0	0.398
today	1	1	0	0	1	0.222	0.222	0.222	0.0	0.0	0.222
too	0	0	0	1	0	0.699	0.0	0.0	0.0	0.699	0.0
topic	0	0	1	0	0	0.699	0.0	0.0	0.699	0.0	0.0
topics	0	0	0	1	0	0.699	0.0	0.0	0.0	0.699	0.0

2 Appendix