Natural Language Processing  
Homework and Programming Assignment 2

Parker Smith

9/29/2022

1. Assign the most probable class to the test sentence given below using naïve bayes classification. Mention each step clearly.

|  |  |  |  |
| --- | --- | --- | --- |
| Doc # | words | Class |  |
| d1 | Chinese Beijing Chinese | B | Training |
| d2 | Chinese Chinese Shanghai | B | Training |
| d3 | Tokyo Japan Chinese | A | Training |
| d4 | Chinese Macao | B | Training |
| **d5** | **Chinese Chinese Chinese Tokyo Japan** | ? | **Testing** |

1. Extract vocabulary from the training corpus:

V = {Chinese, Beijing, Shanghai, Tokyo, Japan, Macao}

1. Calculate the probability for each class

P(A) = 1/4 P(B) = 3/4

1. Calculate the probability of each type given the class (With Laplacian Smoothing)

P(Chinese | A) = 1 + 1 / 3 + 6 = 2 / 9 | P(Beijing | A) = 0 + 1 / 3 + 6 = 1 / 9

P(Shanghai | A) = 0 + 1 / 3 + 6 = 1 / 9 | P(Tokyo | A) = 1 + 1 / 3 + 6 = 2 / 9

P(Japan | A) = 1 + 1 / 3 + 6 = 2 / 9 | P(Macao | A) = 0 + 1 / 3 + 6 = 1 / 9

P(Chinese | B) = 5 + 1 / 8 + 6 = 6 / 14 | P(Beijing | B) = 1 + 1 / 8 + 6 = 2 / 14

P(Shanghai | B) = 1 + 1 / 8 + 6 = 2 / 14 | P(Tokyo | B) = 0 + 1 / 8 + 6 = 1 / 14

P(Japan | B) = 0 + 1 / 8 + 6 = 1 / 14 | P(Macao | B) = 1 + 1 / 8 + 6 = 2 / 14

1. Determine the probability of the class given the features using bayes rule.

P(A | Chinese Chinese Chinese Tokyo Japan)

= P(A) \* P(Chinese | A) \* P(Chinese | A) \* P(Chinese | A) \* P(Tokyo | A) \* P(Japan | A)

= 1/4 \* 2/9 \* 2/9 \* 2/9 \* 2/9 \* 2/9

= 1.35 \* 10-4

P(B | Chinese Chinese Chinese Tokyo Japan)

= P(B) \* P(Chinese | B) \* P(Chinese | B) \* P(Chinese | B) \* P(Tokyo | B) \* P(Japan | B)

= 3/4 \* 6/14 \* 6/14 \* 6/14 \* 1/14 \* 1/14

= 3.01 \* 10-4

Using Naïve Bayes classification, the testing document is identified as B.

1. What is cross-validation? Give an example and explain how cross-validation works. When do you use cross-validation?

Cross validation is a method of testing and training models where multiple iterations of testing and training occur over separate subsets of data. The fitness results from these iterations are averaged to create the final fitness result of the model. For instance, if a dataset has 100 datapoints, a single instance of training and testing may not be sufficient to cover all classes within the data. Typically, cross-validation would split the dataset into 10 subsets and run 10 iterations of training and testing with the test set in a different subset of the dataset each iteration. The results from each iteration are then averaged to create the final result for the model. Cross-validation is primarily used when the size of the dataset is small, or when there is a heavy variance between datapoints within the dataset.

1. Explain how the gradient descent algorithm works. Explain the effect of learning rate on the learning algorithm while updating parameters using the following equations.

wt+1 = wt -

Please show differences among different types of gradient descent – mini-batch, batch, and stochastic gradient descent.

Gradient descent is an algorithm which finds the least amount of loss in a model. It begins by selecting a random starting location. Next, it determines the gradient at that location, multiplies the gradient by the learning rate, then subtracts the calculated gradient from the current loss location. It then repeats these steps until the maximum number of iterations are reached, or the amount of movement each iteration is smaller than a given tolerance.

The learning rate modifies the gradient descent algorithm by determining how much the current loss location moves each iteration. A higher learning rate can sometimes overshoot the minimum loss, however it reaches the minimum loss faster. A lower learning rate takes much longer to reach the minimum loss, but it rarely overshoots the target.

|  |  |  |
| --- | --- | --- |
| Mini Batch | Batch | Stochastic Gradient Descent |
| Mini Batch splits the dataset into N number of batches and calculates/updates the gradient after each small batch. | Batch calculates/updates the gradient after iterating through the entire dataset. | Stochastic Gradient Descent calculates/updates the gradient after each sample within the dataset. |

1. Answer the following questions:

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.

1. Calculate the TF-IDF vector for each token. Show each step.

V = {it, is, going, to, rain, today, ., i, am, not, outside, nlp, an, interesting, topic, includes, ml, ,, dl, topics, too, complete, homework}

|V| = 23

Term Frequency (Count of word in document divided by the length of the document):

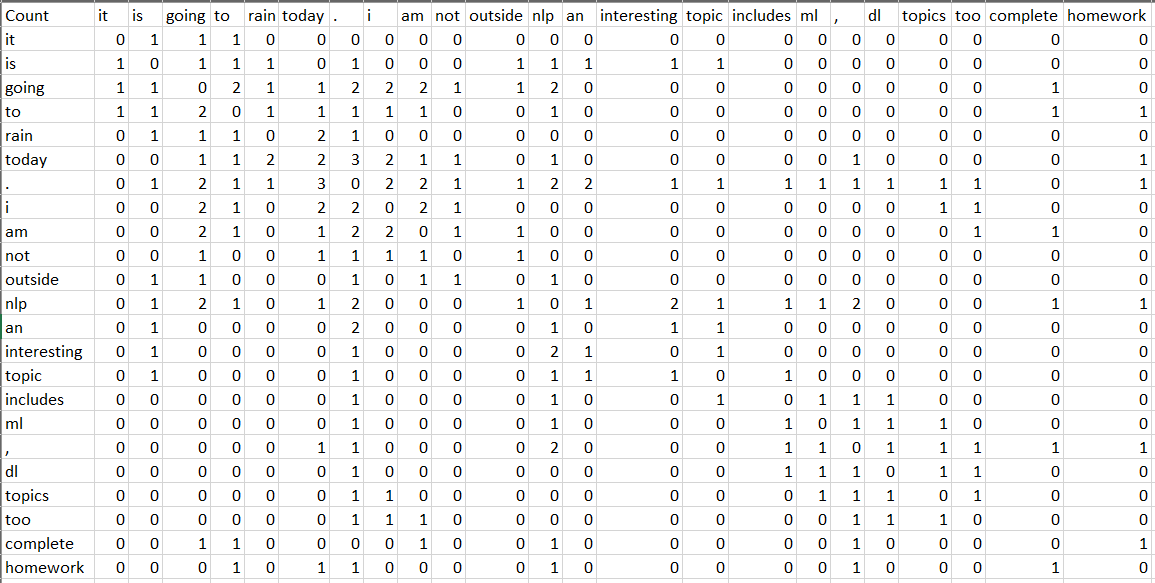
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Word | Document 1 | Document 2 | Document 3 | Document 4 | Document 5 |
| it | 1/7 | 0/7 | 0/6 | 0/8 | 0/10 |
| is | 1/7 | 0/7 | 1/6 | 0/8 | 0/10 |
| going | 1/7 | 1/7 | 0/6 | 0/8 | 1/10 |
| to | 1/7 | 0/7 | 0/6 | 0/8 | 1/10 |
| rain | 1/7 | 0/7 | 0/6 | 0/8 | 0/10 |
| today | 1/7 | 1/7 | 0/6 | 0/8 | 1/10 |
| . | 1/7 | 1/7 | 1/6 | 1/8 | 1/10 |
| i | 0/7 | 1/7 | 0/6 | 0/8 | 1/10 |
| am | 0/7 | 1/7 | 0/6 | 0/8 | 1/10 |
| not | 0/7 | 1/7 | 0/6 | 0/8 | 0/10 |
| outside | 0/7 | 1/7 | 0/6 | 0/8 | 0/10 |
| nlp | 0/7 | 0/7 | 1/6 | 1/8 | 1/10 |
| an | 0/7 | 0/7 | 1/6 | 0/8 | 0/10 |
| interesting | 0/7 | 0/7 | 1/6 | 0/8 | 0/10 |
| topic | 0/7 | 0/7 | 1/6 | 0/8 | 0/10 |
| includes | 0/7 | 0/7 | 0/6 | 1/8 | 0/10 |
| ml | 0/7 | 0/7 | 0/6 | 1/8 | 0/10 |
| , | 0/7 | 0/7 | 0/6 | 1/8 | 1/10 |
| dl | 0/7 | 0/7 | 0/6 | 1/8 | 0/10 |
| topics | 0/7 | 0/7 | 0/6 | 1/8 | 0/10 |
| too | 0/7 | 0/7 | 0/6 | 1/8 | 0/10 |
| complete | 0/7 | 0/7 | 0/6 | 0/8 | 1/10 |
| homework | 0/7 | 0/7 | 0/6 | 0/8 | 1/10 |

Inverse Document Frequency (log(number of documents divided by number of documents where word occurs))

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| it | Log(5/1) = .70 | am | Log(5/2) = .40 | ml | Log(5/1) = .70 |
| is | Log(5/2) = .40 | not | Log(5/1) = .70 | , | Log(5/2) = .40 |
| going | Log(5/3) = .22 | outside | Log(5/1) = .70 | dl | Log(5/1) = .70 |
| to | Log(5/2) = .40 | nlp | Log(5/3) = .22 | topics | Log(5/1) = .70 |
| rain | Log(5/1) = .70 | an | Log(5/1) = .70 | too | Log(5/1) = .70 |
| today | Log(5/3) = .22 | interesting | Log(5/1) = .70 | complete | Log(5/1) = .70 |
| . | Log(5/5) = 0 | topic | Log(5/1) = .70 | homework | Log(5/1) = .70 |
| i | Log(5/2) = .40 | includes | Log(5/1) = .70 |  |  |

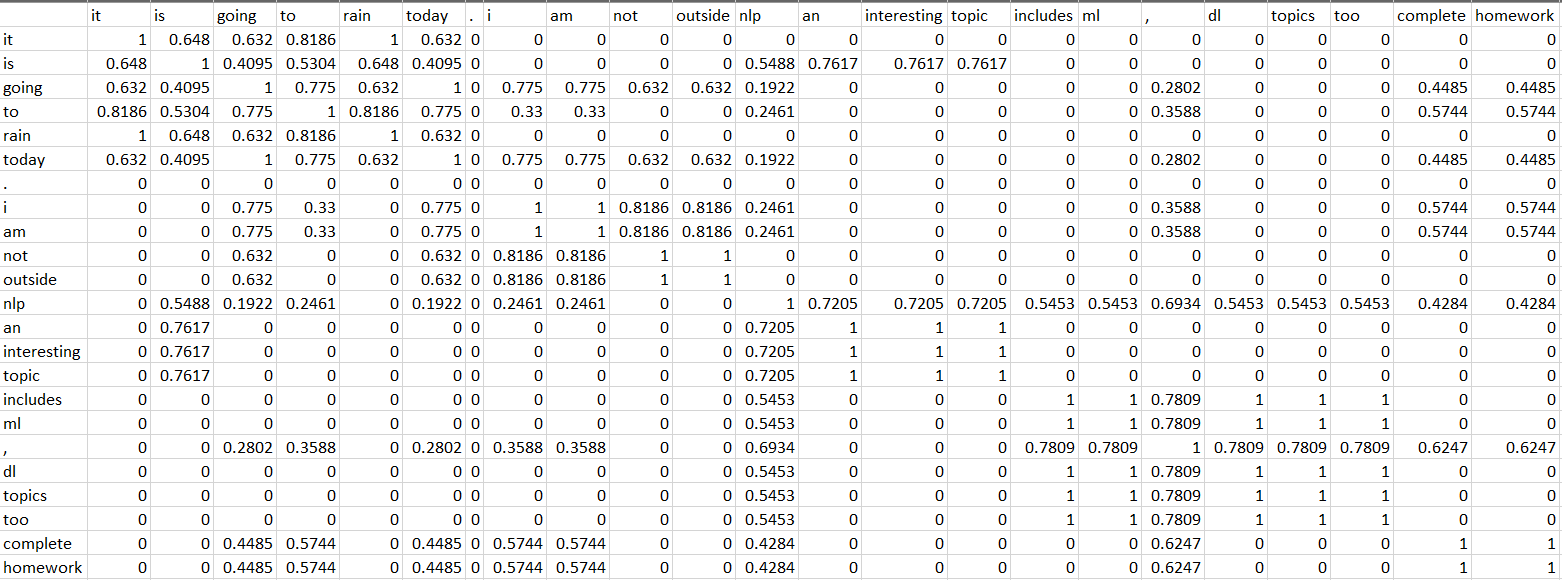
TF-IDF (Term Frequency multiplied by Inverse Document Frequency)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Word | Document 1 | Document 2 | Document 3 | Document 4 | Document 5 |
| it | .100 | 0 | 0 | 0 | 0 |
| is | .057 | 0 | .067 | 0 | 0 |
| going | .031 | .031 | 0 | 0 | .022 |
| to | .057 | 0 | 0 | 0 | .040 |
| rain | .100 | 0 | 0 | 0 | 0 |
| today | .031 | .031 | 0 | 0 | .022 |
| . | 0 | 0 | 0 | 0 | 0 |
| i | 0 | .057 | 0 | 0 | .040 |
| am | 0 | .057 | 0 | 0 | .040 |
| not | 0 | .100 | 0 | 0 | 0 |
| outside | 0 | .100 | 0 | 0 | 0 |
| nlp | 0 | 0 | .037 | .028 | .022 |
| an | 0 | 0 | .117 | 0 | 0 |
| interesting | 0 | 0 | .117 | 0 | 0 |
| topic | 0 | 0 | .117 | 0 | 0 |
| includes | 0 | 0 | 0 | .088 | 0 |
| ml | 0 | 0 | 0 | .088 | 0 |
| , | 0 | 0 | 0 | .05 | .04 |
| dl | 0 | 0 | 0 | .088 | 0 |
| topics | 0 | 0 | 0 | .088 | 0 |
| too | 0 | 0 | 0 | .088 | 0 |
| complete | 0 | 0 | 0 | 0 | .070 |
| homework | 0 | 0 | 0 | 0 | .070 |

1. Calculate the term-term co-occurrence matrix with a given context window 3.
2. Find the most similar pair of words for both TF-IDF and co-occurrence matrix based vector representation cases. To compute similarity score, use cosine similarity. Show similarity calculation for both cases.

Most similar words for TF-IDF:

As shown by the table below, the most similar words are “it” and “rain”, “going” and “today”, “I” and “am”, “not” and “outside”, “an” and “interesting”, “an” and “topic”, “topic” and “interesting”, “ml” and “includes”, “dl” and “includes”, “topics” and “includes”, “too” and “includes”, “dl” and “ml”, “topics” and “ml”, “too” and “ml”, “dl” and “topics”, “too” and “dl”, “topics” and “too”, and “complete” and “homework” with a similarity of 1.



Most similar words for co-occurrence matrix:

As shown by the table below, the most similar words are “am” and “not” with a 0.7698 similarity.

Table

Description automatically generated