Implementation Assignment #2

Alireza Mostafizi

Introduction

I coded this assignment in MATLAB. Attached to this report, you can find the MATLAB codes. The assignment is to implement Naïve Bayes Classifier to the case of document classification, using both Bernoulli and Multinomial Model. The objective is to predict the class of the given test documents with the optimized Naïve Bayes classifier using the train data. The data set is the classic 20-newsgroup data set, and has the following key characteristics represented in Table 1.

	Train Data	Test Data
Number of Documents	11269	7505
Number of Classes	20	20
Vocabulary	61188	61188

Table 1. Data Characteristics

For each document, either test document or train document, the count of used words is presented in the data files. The spit of the documents to different classes is represented in Table 2, which shows that the documents are divided to fairly equal groups so the learning process and optimization won't be biased towards one or a set of the classes.

#	Classes	Test Data		Train Data	
		Count	P	Count	P
1	alt.atheism	480	0.043	318	0.042
2	comp.graphics	581	0.052	389	0.052
3	comp.os.ms-windows.misc	572	0.051	391	0.051
4	comp.sys.ibm.pc.hardware	587	0.052	392	0.052
5	comp.sys.mac.hardware	575	0.051	383	0.051
6	comp.windows.x	592	0.053	390	0.052
7	misc.forsale	582	0.052	382	0.051
8	rec.autos	592	0.053	395	0.053
9	rec.motorcycles	596	0.053	397	0.053
10	rec.sport.baseball	594	0.053	397	0.053
11	rec.sport.hockey	598	0.053	399	0.053
12	sci.crypt	594	0.053	395	0.053
13	sci.electronics	591	0.052	393	0.052
14	sci.med	594	0.053	393	0.052
15	sci.space	593	0.053	392	0.052
16	soc.religion.christian	599	0.053	398	0.053
17	talk.politics.guns	545	0.048	364	0.049
18	talk.politics.mideast	564	0.050	376	0.050
19	talk.politics.misc	464	0.041	310	0.041
20	talk.religion.misc	376	0.033	251	0.033

Table 2. Split of the documents of the data set over the classes

Methodology

The methodology is briefly explained in this section. The first parameter estimation is the estimation of p_y using MLE estimation without Laplace smoothing since none of the classes have zero documents represented the data. Doing so, the parameters can be calibrated using the training data set as following:

$$p_y = \frac{Number\ of\ documents\ in\ class\ y}{Total\ number\ of\ document\ in\ the\ data\ set}$$

Bernoulli Model

As explained in the assignment, each document is explained by a set of binary variables which represent the presence, or the absence, of each word of vocabulary set in the document. The probability of observing a specific document x in the class of y is defined as following:

$$p(x|y) = \prod_{i=1}^{|V|} p_{i|y}^{x_i} \left(1 - p_{i|y}\right)^{(1-x_i)}$$

Where $p_{i|y}$ represents the probability of a document of class y containing the ith word of vocabulary. Using MAP estimation or Laplace Smoothing on the training data set, $p_{i|y}$ can be calibrated as following:

$$p_{i|y} = \frac{Number\ of\ document\ in\ class\ y, containing\ the\ ith\ word + 1}{Number\ of\ total\ documents\ in\ class\ y + 2}$$

As shown in the above equation, the order one Laplace smoothing has been implemented in the case of Bernoulli model, which is equivalent to $\alpha = 2$, and $\beta = 2$. The parameters have been chosen based on their effect on the distribution of probabilities, and the mentioned set of parameters will force a range of 0.0017 to 0.9832 to the probabilities which is fairly reasonable and far from 0 and 1.

Multinomial Model

Using Multinomial model, each document can be described as a set of integer-valued variables representing the number that each word of the dictionary has been used in the document. Therefore, the probability of observing document x in the class of y can be formulated as follows:

$$p(x|y) = \prod_{i=1}^{|V|} p_{i|y}^{x_i}$$

Where $p_{i|y}$ equals the probability of a randomly selected word from the whole class y being the ith word of the vocabulary. With this in mind, the mentioned probability can be calibrated using training data set with the following approach of MAP estimation (Laplace Smoothing):

$$p_{i|y} = \frac{Number\ of\ times\ the\ ith\ word\ is\ used\ in\ class\ y + 1}{The\ total\ number\ of\ words\ used\ in\ class\ y + 61188}$$

The Laplace smoothing considered here is equivalent to have $\alpha = 1$ for the Drichlet Distribution. The number 61188 is the number of words in the vocabulary which equals to $\Sigma\alpha$, or K. In addition, as noted in the assignment: $\forall y: \quad \sum_{i=1}^{|V|} p_{i|y} = 1.$

Part I: Basic Implementation

1. Using Logarithm of Probabilities

As suggested in the assignment, since the probabilities are normally small values, product of these small values to calculate $p(x|y)$ might result of extremely negligible values that leads to underflow issues. To fix this problem, we can use the logarithm of probabilities as bellow.

$$y^* = \arg\max_y p(y|\boldsymbol{x})$$

$$= \arg\max_y \frac{p_y \,.\, p(\boldsymbol{x}|y)}{p(\boldsymbol{x})}$$

$$= \arg\max_y p_y \,.\, p(\boldsymbol{x}|y)$$

Therefore the decision criteria for classification is this product of probabilities, $p(y) \,.\, p(\boldsymbol{x}|y)$. The class with the highest product should be selected. Thus, taking logarithm of this product, we have:

$$\log\left(p_y \,.\, p(\boldsymbol{x}|y)\right) = \log(p_y) + \log(p(\boldsymbol{x}|y))$$

In case of **Bernoulli** model, we have:

$$\log(p_y) + \log(p(\boldsymbol{x}|y)) = \log(p_y) + \log(\textstyle\prod_{i=1}^{|V|} p_{i|y}^{x_i} \left(1 - p_{i|y}\right)^{(1-x_i)})$$

$$= \log(p_y) + \textstyle\sum_{i=1}^{|V|} x_i \log p_{i|y} + (1 - x_i) \log(1 - p_{i|y})$$

Therefore, instead of using probabilities, we use logarithm functions of them, and since the log function is injective and increasing, the following equation is valid:

$$y^* = \arg\max_y p(y|\boldsymbol{x}) = \arg\max_y \left[\log(p_y) + \textstyle\sum_{i=1}^{|V|} x_i \log p_{i|y} + (1 - x_i) \log(1 - p_{i|y})\right]$$

Similarly, in the case of **Multinomial** model, we have:

$$\log(p_y) + \log(p(\boldsymbol{x}|y)) = \log(p_y) + \log(\textstyle\prod_{i=1}^{|V|} p_{i|y}^{x_i})$$

$$= \log(p_y) + \textstyle\sum_{i=1}^{|V|} x_i \log p_{i|y}$$

With the same logic, we can state that:

$$y^* = \arg\max_y p(y|\boldsymbol{x}) = \arg\max_y \left[\log(p_y) + \textstyle\sum_{i=1}^{|V|} x_i \log p_{i|y}\right]$$

2. Overall Testing Accuracy

The overall testing accuracy of the models are presented in Table 3. As expected, since Bernoulli model neglects part of the information of the training data set regarding the number of words being used in the document, the Bernoulli model leads to lower accuracy rate rather than Multinomial model does.

Testing Accuracy Rate	
Bernoulli Model	% 62.4
Multinomial Model	% 78.1

Table 3. Accuracy Rates of Different Models

The detail of false predictions is presented in the next section.

3. Confusion Matrix

I address this question using the K by K matrix as suggested in the assignment. In addition, I calculated the percentage of a class being correctly predicted, and the percentage of a class being confused by the other classes. The statistics can be formulated as following:

$$Prediction\ index_i = \frac{Number\ of\ Class\ i\ test\ documents\ being\ correctlt\ predicted}{Total\ number\ of\ documents\ in\ class\ i} * 100$$

$$Confusion\ index_i = \frac{Number\ of\ documents\ correctly\ predicted\ to\ be\ in\ class\ i}{Total\ number\ of\ documents\ predicted\ to\ be\ in\ class\ i} * 100$$

With this in mind, we can state that:

- The **higher** the **Prediction Index** is, the more **discriminable** the class is by the applied classifier
- The **lower** the **Confusion Index**, the more **confusable** the class is by the applied classifier

The confusion matrix for the Bernoulli model and the Multinomial model, along with the proposed statistics, is presented in Table 4 and 5, respectively. The discussion part is followed by each matrix.

		PREDICTED Class Labels - **BERNOULLI**																				Prediction Index
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
TRUE Class Labels	1	139	4	1	5	18	0	39	5	15	10	0	1	6	7	3	57	1	4	2	1	43.7
	2	0	204	11	19	15	13	99	3	0	0	0	12	5	1	4	2	0	1	0	0	52.4
	3	0	10	221	58	14	9	54	1	1	1	0	11	2	0	2	5	0	0	1	1	56.5
	4	0	1	12	279	16	2	59	1	0	0	0	4	18	0	0	0	0	0	0	0	71.2
	5	0	1	4	22	269	0	72	2	0	0	0	3	7	0	3	0	0	0	0	0	70.2
	6	0	45	34	16	12	206	68	1	0	0	0	5	2	1	0	0	0	0	0	0	52.8
	7	0	1	1	13	1	1	349	8	0	0	0	1	5	0	0	0	2	0	0	0	91.4
	8	0	2	1	4	10	0	76	280	7	0	0	0	13	0	1	0	1	0	0	0	70.9
	9	0	1	0	0	4	0	35	11	341	1	0	1	0	0	0	0	1	1	1	0	85.9
	10	1	1	0	0	1	0	53	0	0	331	0	0	1	0	0	6	1	0	2	0	83.4
	11	0	1	0	1	2	0	51	0	2	27	308	1	2	0	0	4	0	0	0	0	77.2
	12	0	15	3	14	34	1	59	1	3	1	1	244	14	0	1	1	2	0	1	0	61.8
	13	0	11	5	29	14	0	87	6	2	0	0	15	220	3	0	0	1	0	0	0	56.0
	14	0	6	1	7	34	0	96	4	15	1	0	0	22	189	1	11	1	1	4	0	48.1
	15	0	16	1	1	11	3	55	10	3	0	0	3	30	2	251	2	2	0	2	0	64.0
	16	1	5	0	6	19	0	51	0	2	7	0	0	4	0	0	301	1	0	1	0	75.6
	17	0	0	1	1	15	0	50	22	15	10	0	3	12	1	2	4	228	0	0	0	62.6
	18	1	2	1	1	20	0	46	5	15	16	0	2	3	1	0	23	5	233	2	0	62.0
	19	3	3	1	7	20	0	38	20	13	16	0	2	10	4	10	15	65	0	83	0	26.8
	20	26	3	2	9	17	0	37	5	9	10	0	1	3	4	3	91	21	2	1	7	2.8
Confusion Index		81.3	61.4	73.7	56.7	49.3	87.7	23.7	72.7	77.0	76.8	99.7	79.0	58.0	88.7	89.3	57.7	68.7	96.3	83.0	77.8	

Table 4. Confusion Matrix of Bernoulli Classifier

Discussion

As expected, the performance is way worse than that of Multinomial classifier. The worst class in terms of confusion is class 7, which confuses predictions of almost all other classes. In addition, classifier failed to predict almost all documents of class 20 which has the lowest prediction index of 2.8. Only 7 documents where predicted correctly. Basically we can interpret that class 7 it is a pool of other classes, and class 20 has no discriminative words in it.

		PREDICTED Class Labels - **MULTINOMIAL**																				Prediction Index
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
TRUE Class Labels	1	235	0	0	0	0	1	0	0	0	0	1	1	1	2	3	45	3	10	7	9	73.9
	2	3	296	6	12	7	22	1	3	2	0	0	17	4	4	7	4	0	0	1	0	76.1
	3	3	33	207	58	11	31	0	2	2	2	1	17	1	4	4	5	0	0	9	1	52.9
	4	0	8	15	305	21	2	4	6	0	0	1	6	23	0	1	0	0	0	0	0	77.8
	5	0	8	10	37	273	3	4	4	1	1	0	6	17	8	2	0	3	0	6	0	71.3
	6	0	42	7	10	2	306	1	0	2	1	0	10	0	0	3	2	1	1	2	0	78.5
	7	0	8	4	50	20	1	226	33	5	0	1	3	11	2	3	4	2	3	6	0	59.2
	8	1	1	0	2	0	1	5	356	4	2	0	1	4	0	2	1	4	2	9	0	90.1
	9	0	1	0	0	0	0	0	26	353	2	0	1	1	1	0	1	4	2	5	0	88.9
	10	4	1	0	1	1	2	3	3	1	345	17	2	2	0	0	3	1	2	9	0	86.9
	11	2	0	0	0	0	0	1	1	0	4	381	1	0	2	1	2	0	1	3	0	95.5
	12	0	4	1	1	2	1	1	0	0	0	0	361	3	2	0	2	8	0	8	1	91.4
	13	2	18	0	27	8	3	1	10	2	0	0	46	259	6	3	6	0	2	0	0	65.9
	14	10	7	1	3	0	0	0	4	0	1	0	1	3	324	3	17	3	6	10	0	82.4
	15	3	7	0	0	0	2	0	0	1	0	1	4	4	4	335	5	1	2	22	1	85.5
	16	7	2	1	0	1	2	0	0	0	0	0	1	0	1	0	377	2	2	1	1	94.7
	17	1	0	0	0	1	0	1	2	1	1	1	3	0	1	2	3	325	2	16	4	89.3
	18	12	1	0	0	0	0	0	2	1	1	1	4	0	0	0	8	3	325	18	0	86.4
	19	6	1	0	0	0	1	0	1	0	0	0	3	0	3	7	3	95	5	184	1	59.4
	20	47	3	0	0	0	0	0	0	1	0	0	1	0	3	5	70	19	5	8	89	35.5
Confusion Index		69.9	67.1	82.1	60.3	78.7	81.0	91.1	78.6	93.9	95.8	94.1	73.8	77.8	88.3	87.9	67.6	68.6	87.8	56.8	83.2	

Table 5. Confusion Matrix of Multinomial Classifier

Discussion

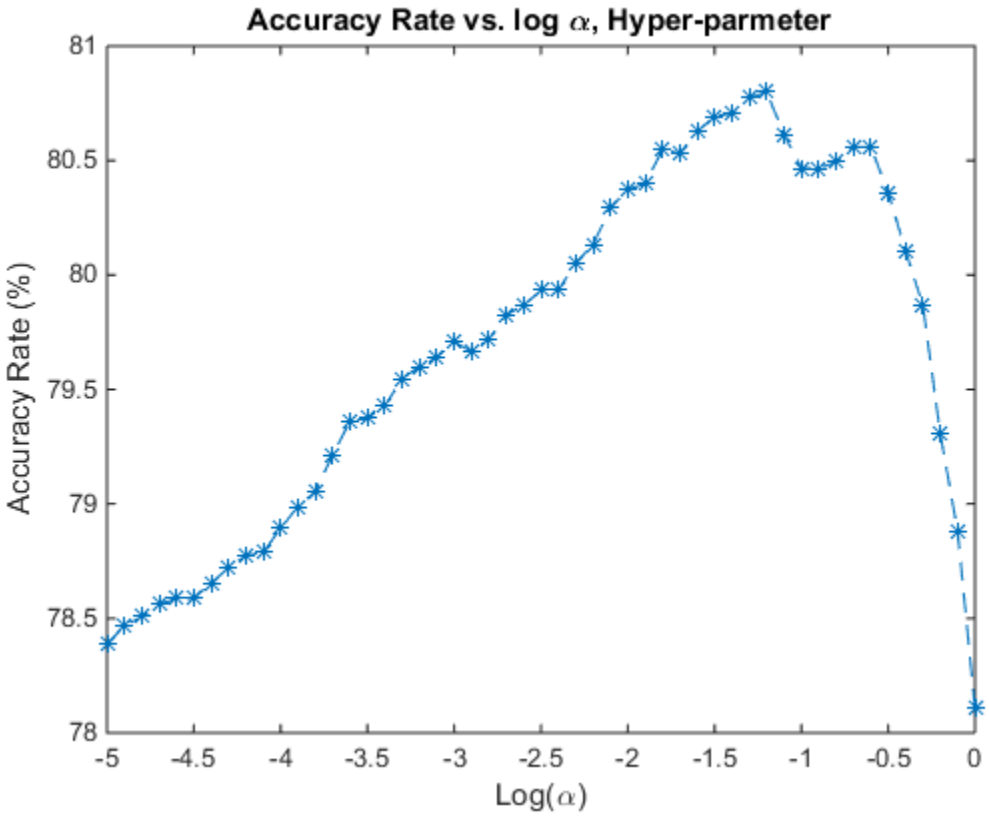
As you can see, classes with low Prediction Index and low Confusion Index are highlighted. Classifier is not performing well on predictions of classes 3, 17, 13, 19, and 20. This can be caused by sparse documents, the word list of these classes, the class type and its characteristics, or, in case of class 20, low number of training documents. In addition, classes 1, 2, 4, 16, and 19 are the ones that are the most confusable. In other words, they are the most falsely predicted ones since they have the lowest confusion index. Some of the bizarre predictions are also marked.

Part II: Priors and Overfitting

The parameter α shows up in the Laplace Smoothing for Multinomial model as following:

$$p_{i|y} = \frac{Number\ of\ times\ the\ ith\ word\ is\ used\ in\ class\ y + \alpha}{The\ total\ number\ of\ words\ used\ in\ class\ y + \alpha * |V|}$$

Where in this case, with the full vocabulary list, the |V| is 61188. $\alpha = 1$ has been used for our prior investigations. Here I, played with α ranging from 10^{-5} to 1. Figure 1 shows the trend of accuracy rate of multinomial model associated with the hyper parameter chosen for the classifier.



Discussion

As the graph shows, the accuracy rate increases with the increase of α to some extent, and then start to decrease. This phenomenon does make sense, since we were expecting the α to have an optimized value. With low values of α, the probabilities can be very close to either zero, in case the word has not been used at all, which leads to overfitting the classifier. As you increase the value of hyper parameter close to one, the over fitting problem will be resolved to some extent since the probabilities get farther from zero, even if the word has not even showed up in the class. But after this point, the more you increase α, the probabilities of the words being used in each document get clustered in a short range, and the classifier perform well with those probabilities which causes lower accuracy rates.

Part III: Identifying Important Features

I approached this part of the assignment by removing some the words from the vocabulary intuitively, and running the classifier using Multinomial Model, mainly because it performs better initially, and takes the number of words into consideration. In addition, it runs faster than Bernoulli model does. The hyper-parameter is set to 1, for comparison purposes to the main experiments which lead to 78 percent accuracy. To do that, I calculated the following statistic for each word in the vocabulary.

$$PresenceCount\ Index_i = \frac{Number\ of\ Classes\ in\ which\ Word_i\ has\ shwed\ up}{Total\ Number\ of\ Classes}$$

Which basically shows the number of classes that contain each word. I started training the classifier using the modified vocabulary set, and classify the test documents with the optimized classifier. Some selected results are shown in Table 6. The modification is expected to have the same effect on the Bernoulli classifier since I neglected the number of word counts in the modification, and selected words only based on the presence or absence in the class.

The intuition says that:

- Words that are frequently used in most of the classes are of lower significances.
  Since these words are used in most of the classes, they don't have high discriminative power.
- Words that are used in all the classes are not significant
  The filtered vocabulary shows that these words are the most common words, such as "are," "the," "it," etc.
- Words that are used in none of the classes are not significant
  The filtered vocabulary shows that these words are the technical words, like medical words which have not been used in any of the classes
- Words that are used in all the classes except one might be significant
  These words can be used to discriminate one of the classes over the other ones, where probability of these words being absent in the class is high. Focusing on these words is expected to improve the performance of Bernoulli Model to some extent.

Therefore, keeping the four points mentioned above in mind, the strategy I took was to try different sets of words and check whether the accuracy rate increases or decreases. Each Scenario is discussed following the Table 5. Figure 2 also shows the performance of different scenarios with respect to the words they have used to train the classifier.

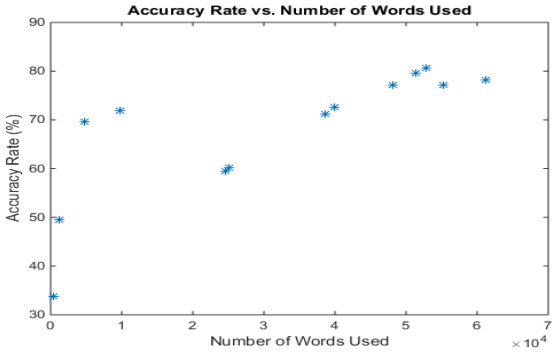


Figure 2. Accuracy Rate vs. Number of Words Used

#	Scenario		Accuracy Rate
	Words with PresenceCount Index of	Vocab Count	
1	1	24591	59.47
2	19	425	33.68
3	1, 19	25016	60.13
4	1, 2, 3	38693	71.18
5	17,18,19	1146	49.58
6	1, 2, 3, 19, 18, 17	39839	72.58
7	1, 2, … , 9	48065	77.19
8	0,1, … , 9	55278	77.13
9	10, 11, … , 19	4769	69.65
10	5, 6, … , 15	9688	71.91
11	1, 2, … , 15	51292	79.52
12	1,2, … , 19	52834	80.68
13	0, 1, … , 20 *ALL WORDS*	61188	78.15

Table 6. The scenarios tested

Discussion

1. Scenario one only consider the words that have showed up in one class.
2. Scenario two only considers the words that have showed up in all classes except one. This shows that scenario one has the words with higher discriminative characteristics.
3. Putting both scenario one and two together does not really improve the accuracy rate of scenario one.
4. Comparing scenario three to scenario one, we can state that having words that showed up in up to three classes help the accuracy rate by 12 percent.
5. Again, the words that have showed up to most of the classes do not have high discriminative characteristics.
6. Scenario six is the combination of four and five which does not really improve scenario four.
7. Having words that showed up in up to nine classes has a result almost as well as running the classifier with the full vocabulary.
8. Comparing scenario seven and eight, including the words that did not show up in any of the classes deteriorates the performance of our classifier.
9. Running the model on the other half of vocabulary spectrum has a worse performance.
10. The middle part of the words spectrum, with respect to their *PresenceSount Index*, does not show a great experience.

Conclusion

Looking at the scenario eleven and twelve, we can state that although words with high *PresenceCount index* have lower discriminative power, but keeping them in the classifier improves the results. On the other hand, comparing scenario twelve and thirteen, removing words that have never used, or the words that have been used in all the classes improves the performance of classifier.

Notes on running the code:

1. Main code is “Main.m,” and the others are functions.
2. Run the code, part by part, separated by comments.
3. Keep in mind that it takes a lot of time!