CS - 529 : Introduction to Machine learning
Project 2: Naive Bayes Document Classifier

Muntasir Al Kabir, Pankaz Das

October 20, 2017

1 Introduction

Naive Bayes classifiers are among the most successful known algorithms for classifying text based documents. In some domains, the performance of a Naive Bayes learner is comparable to that of neural network and decision tree learning. The most distinctive feature of Naive Bayes classifier is that it has no explicit search through the space of possible hypotheses. Instead, the hypothesis is formed by counting the frequency of various data combinations within the training examples. Because of this distinctive feature, Naive Bayes classifier is an attractive candidate for screening/classifying documents.

2 Theory

Here we give a brief summary of the Naive Bayes classification method and provide necessary equations.

Bayes Theorem: Bayes theorem is a famous equation that allows us to make predictions based on data, which is given as below:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}. \tag{1}$$

Let us define A = class and B = Data, then we can rewrite the equation as

$$P(class|data) = \frac{P(data|class)P(class)}{P(data)}. \tag{2}$$

where class is a particular class (e.g. male), data is an observation's data, P(class|data) is called the posterior, P(data|class) is called the likelihood, P(class) is called the prior, and P(data) is called the marginal probability.

Naive Bayes Assumption: In the earlier example given in equation (2), if data means many events such as $E_1, E_2, ..., E_n$, then the equation becomes:

$$P(class|E_1, E_2, ..., E_n) = \frac{P(E_1, E_2, ..., E_n|class)P(class)}{P(E_1, E_2, ..., E_n)}. \tag{3}$$

However in Naive Bayes method, it is assumed that the events $E_1, E_2, ..., E_n$ are conditionally independent of each other given the class (which may not be realistic for many real world experiments). Therefore we can rewrite (3) as

$$P(class|E_1, E_2, ..., E_n) = \frac{P(E_1|class)P(E_2|class), ..., P(E_n|class)P(class)}{P(E_1, E_2, ..., E_n)}. \quad (4)$$

Now the decision rule for the Naive Bayes is:

$$class^* = h_{NB}(E_1, E_2, ..., E_n) = \underset{class}{\text{argmax}} P(class) \prod_i P(E_i|class). \quad (5)$$

Multinomial Distribution: For the multinomial distribution we usually assume the Dirichlet distribution for the priors, which can be given by for the k class

$$P(\theta) = \frac{\prod_{i=1}^{k} \theta_{i=1}^{\alpha_i - 1}}{B(\alpha_1, ..., \alpha_k)} \tilde{=} Dirichlet(\alpha_1, ..., \alpha_k). \quad (6)$$

where $\alpha_i = \beta + 1$ and $\beta = \frac{1}{|V|}$, $|V|$ = length of the vocabulary list. So MAP equation for the Naive Bayes would be

$$P(X_i|Y_k) = \frac{(\text{count of } X_i \text{ in } Y_k) + (\alpha - 1)}{(\text{total words in } Y_k) + |(\alpha - 1)(\text{length of vocabulary list})|} \quad (7)$$

where $\alpha = (\beta + 1) - 1 = (\frac{1}{|V|} + 1) - 1$.

Then the classification rule becomes

$$Y^{new} = \underset{Y_k}{\text{argmax}} [log(P(Y_k)) + \sum_i (\text{number of } X_i^{new}) log(P(X_i|Y_k))]. \quad (8)$$

3 Data description

There are four different data files available for the project, which are described as below:

vocabulary.txt is a list of the words that may appear in documents. The line number is words d in other les. That is, the rst word (archive) has wordId 1, the second (name) has wordId 2, etc.

newsgrouplabels.txt is a list of newsgroups from which a document may have come. Again, the line number corresponds to the labels id, which is used in the .label les. The rst line (alt.atheism) has id 1, etc.

training.csv Species the counts for each of the words used in each of the documents. Each line contains 61190 elements. The first element is the document id, the elements 2 to 61189 are word counts for a vectorized representation of words (refer to the vocabulary for a mapping). The last element is the class of the document (20 different classes). All word/document pairs that do not appear in the le have count 0.

testing.csv The same as training.csv except that it does not contain the last element.

4 Implementation

We Implement the Naive Bayes Document Classifier by using python. We started with importing dataset. At first, we calculate the prior probabilities from class variables. We use very simple function in association with very popular python's dictionary, which is fast enough and user friendly. We start our code initially in R, use of data-frame took a lot of time in searching and calculating probabilities from the large dataset. Then we calculate MAP to get every word's probability with all the class of the training set. And finally we classify validation set by another simple function. The total time taken to run the whole code 10000 second.

To get the confusion matrix to analyze our method we separate training set to 10001 for training and 1999 for validation set. We trained our classifier with the training set. Then we classify our validation set and found around 81 percent accuracy. We create the confusion matrix of 1999 test set to explore our method more closely.

We use very basic plot library of python matplotlib to plot the accuracy vs beta value plot. We use logarithmic scale in x axis to get a better visualization. And finally we calculate entropy of every words probability to find calculate highest ranking word where the Nave Bayes classifier rely on. To do so we use entropy function and dictionary.

5 Results

The summarized results in different setting of beta are given below.We found the best accuracy at beta value 0.00000000000001.

Table 1: Accuracy With Different Beta Value

Beta Value	Testing Accuracy
0.00000000000001	0.82728
0.00000001	0.82019
0.000001	0.81310
0.00001	0.80690
0.5	0.79716
0.6	0.58990
0.9	0.55594
1.0	0.54413

For the small β, we have the probability of unseen word tends to zero which was not present in training set. Some words in training set also occur few in training documents, which might be different class than test dataset, therefore, accuracy decreases. When β is large, the method experienced overfitting for the training set i.e. bias to training set. As a result for the unknown validate set it gives misclassification.

6 Analysis

In this section, we will analyze our project results by answering the questions below:

Question 1: In your answer sheet, explain in a sentence or two why it would be difficult to accurately estimate the parameters of this model on a reasonable set of documents (e.g. 1000 documents, each 1000 words long, where each word comes from a 50,000 word vocabulary).

Answer: In this model, if a word, say "muntasir", appears at two different positions in a single document, then their probability will be different, which is a pattern/sequence recognition problem. For a document with 1000 words there are 1000 random variables with the domain of 50000 words, which results in a very computationally huge conditional probability estimation problem of features. In other words, we are not using the occurrences of a single word in a document which is not efficient in statistics.

Question 2: In your answer sheet, report your overall testing accuracy (Number of correctly classified documents in the test set over the total number of test documents), and print out the confusion matrix (the matrix C, where c_{ij} is the number of times a document with ground truth category j was classified as category i).

Answer: To get the confusion matrix we divide the training set to 10001 for training and 1999 for validate. We got 80.44 percent accuracy. The confusion matrix is is figure 1.

Question 3: Are there any newsgroups that the algorithm confuses more often than others? Why do you think this is?

Answer: We found in the confusion matrix, the similar topics are confused frequently, for example, the group (sci.electronics, sci.space) and (talk.politics.mideast and talk.politics.misc). Newsgroup with similar topics contain similar words. That is why the misclassification rate at those groups is higher than other.

Question 4: Re-train your Naive Bayes classifier for values of β between .00001 and 1 and report the accuracy over the test set for each value of β. Create a plot with values of β on the x-axis and accuracy on the y-axis. Use a logarithmic scale for the x-axis (in Matlab, the semilogx command). Explain in a few sentences why accuracy drops for both small and large values of β.

Answer: Figure 2 shows the results of accuracy versus β. Observe that, when β is small, we have the probability of unseen word which was not present in training set tends to zero. Some words in training set also occur few in training documents which might be different class than test dataset, therefore, accuracy decreases.

For large β the method experienced overfitting for the training set i.e. bias to training set. As a result for the unknown validate set it gives misclassification.

		PredictedClass																			
		1 alt.atheism	2 comp.graphics	3 comp.os.ms-windows.misc	4 comp.sys.ibm.pc.hardware	5 comp.sys.mac.hardware	6 comp.windows.x	7 misc.forsale	8 rec.autos	9 rec.motorcycles	10 rec.sport.baseball	11 rec.sport.hockey	12 sci.crypt	13 sci.electronics	14 sci.med	15 sci.space	16 soc.religion.christian	17 talk.politics.guns	18 talk.politics.mideast	19 talk.politics.misc	20 talk.religion.misc
TrueClass	1 alt.athei	67	0	0	0	0	0	0	11	0	3	0	0	2	0	0	0	0	0	0	0
	2 comp.gr	0	88	3	1	0	0	0	2	3	1	1	0	0	0	0	0	0	0	0	0
	3 comp.os	0	1	98	1	0	1	0	0	0	0	2	0	0	0	1	0	1	0	1	1
	4 comp.sy	0	0	0	108	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	5 comp.sy	0	0	0	9	68	4	2	2	0	2	1	4	0	0	3	1	1	0	0	0
	6 comp.wi	0	0	0	2	2	89	0	4	0	5	0	0	0	0	0	0	0	0	0	0
	7 misc.for	1	0	0	1	0	1	107	4	0	1	1	1	0	0	0	0	0	0	0	0
	8 rec.auto	0	0	0	0	0	0	0	85	0	2	0	1	0	0	0	0	0	0	0	0
	9 rec.mot	0	0	0	1	0	0	0	0	92	3	2	0	1	0	0	0	0	0	0	0
	10 rec.spo	0	0	0	0	0	0	0	0	0	104	0	0	0	0	0	0	0	0	0	0
	11 rec.spo	1	0	0	1	0	0	0	3	3	6	54	0	0	0	0	0	0	0	0	0
	12 sci.cryp	0	0	0	2	0	2	1	2	0	0	0	99	0	4	2	0	1	0	0	0
	13 sci.elec	1	0	0	1	0	0	2	28	2	3	4	0	27	0	0	0	0	0	0	0
	14 sci.med	0	0	0	11	0	0	1	0	0	1	0	5	0	73	11	0	8	0	0	0
	15 sci.spa	0	0	0	7	0	0	0	0	0	0	0	7	0	4	65	1	1	1	0	0
	16 soc.reli	0	0	0	11	3	2	2	4	1	1	0	8	0	0	1	61	2	0	0	0
	17 talk.po	0	0	0	1	0	1	0	2	0	2	1	6	0	0	0	0	92	0	0	0
	18 talk.po	0	0	2	13	3	3	6	6	2	3	0	11	0	3	19	5	1	43	8	1
	19 talk.po	0	0	0	9	0	0	0	0	1	3	5	0	0	0	0	0	0	1	82	2
	20 talk.rel	0	0	0	1	0	0	0	1	2	3	0	0	0	0	0	0	0	0	3	106

Figure 1: Confusion matrix of naive Bayes implementation of model 2.

Question 5: Propose a method for ranking the words in the dataset based on how much the classifier relies on them when performing its classification (hint: information theory will help). Your metric should use only the classifiers estimates of P (Y) and P($X|Y$). It should give high scores to those words that appear frequently in one or a few of the newsgroups but not in other ones. Words that are used frequently in general English (the, of, etc.) should have lower scores, as well as words that only appear extremely rarely throughout the whole dataset. Finally, in your method this should be an overall ranking for the words, not a per-category ranking.

Answer: For the document with a single word for example "pankaz", we need to calculate H(Y $|X_i =$ True). Intuitively, this value will be low if a word appears most of the time in a single class, because the distribution P(Y$|X_i =$ True) will be highly peaked. After getting all word's total conditional probability, we can get the rank. In our method we consider

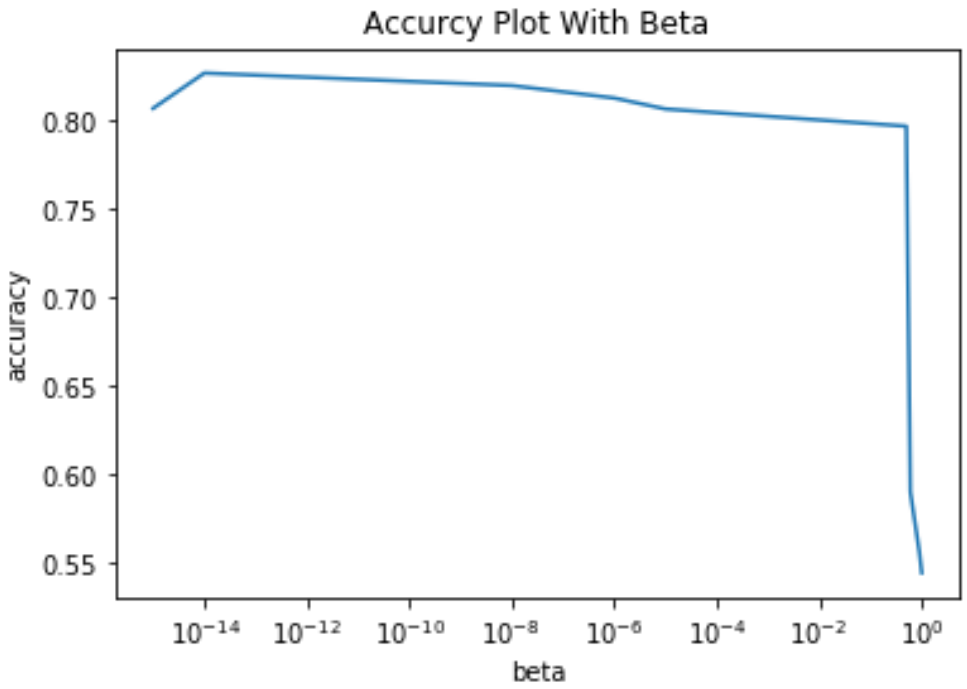

Figure 2: Accuracy versus beta (β).

$$
\begin{aligned}
H(Y|X_i = T) &= -\sum_k P(Y = y_k|X_i = T) log(P(Y = y_k|X_i = T)) \\
&= -E_{P(Y|X_i=T)} log(P(Y = y_k|X_i = T)) \\
&= -E_{P(Y|X_i=T)} log \frac{P(X_i = T|Y = y_k)P(Y = y_k)}{(P(X_i = T))} \\
&= -E_{P(Y|X_i=T)} log \frac{P(X_i = T|Y = y_k)}{(P(X_i = T))} - E_{P(Y|X_i=T)} log P(Y = y_k)
\end{aligned}
\tag{9}
$$

Question 6: Implement your method, set b back to $\frac{1}{|V|}$, and print out the 100 words with the highest measure.

Answer: The first 100 words are:
'ffrf', 'figmo', 'haught', 'freethought', 'aah', 'rationalist', 'islington', 'freethinker', 'ibka', 'internationaler', 'konfessionslosen', 'atheisten', 'materialien', 'informationen', 'politisches', 'konfessionslosesn', 'hrsg', 'vertrieb', 'ibdk', 'ucherdienst', 'disch', 'canticle', 'leibowitz', 'doomsday', 'blueprints', 'pangborn', 'approachable', 'healer', 'craftsmen', 'earthers', 'fernwright', 'dismissively', 'handmaid', 'theocracy', 'vicars', 'enlighting', 'adulteries', 'gottes', 'diener', 'dunkle', 'seite', 'papsttums', 'droemer', 'knaur', 'contempory', 'platinga', 'swinburne', 'unsupportable', 'subtitled', 'secularization', 'ballantine', 'clarendon', 'trilogy', 'posthumous', 'anselm', 'sidgwick', 'restatements', 'theses', 'plantinga', 'lelie', 'axiarchism', 'refreshingly', 'anthology', 'unfathomable', 'mutations', 'johnsd', 'jec', 'unamerican', 'propogation', 'golen',

'kuweit', 'psycholgical', 'parallelism', 'bobbe', 'beauchaine', 'unswerving', 'blender', 'jurisprudence', 'imprision', 'probablity', 'vehicals', 'possibilty', 'wand', 'mauled', 'ssauyet', 'sauyet', 'feud', 'slayings', 'homophobe', 'latent', 'traits', 'abberation', 'procreation', 'genetically', 'legitimization', 'crushes', 'deviates', 'puma', 'intelligibly', 'leapard' .

Question 7: If the points in the training dataset were not sampled independently at random from the same distribution of data we plan to classify in the future, we might call that training set biased. Dataset bias is a problem because the performance of a classifier on a biased dataset will not accurately reflect its future performance in the real world. Look again at the words your classifier is relying on. Do you see any signs of dataset bias?

Answer: For the first few sentences of the dataset is ('ffrf', 'figmo', 'haught', 'freethought', 'aah', 'rationalist', 'islington', 'freethinker', 'ibka', 'internationaler', 'konfessionslosen'), which reflects psychological context. We don't know about the time period of the dataset. Is it usual the classifier could be rely on some word that are time constraint. The second word figmo which is mostly used in military.

7 Conclusions

After this project, we strongly believe that Naive Bayes classifiers can be used for natural language processing. It can produce promising results in document/word and resume-job matching. However, as seen in the project, noisy and similar data can affect the performance of the classifier negatively and significantly. The main reason is that Naive Bayes classifier does not do explicit search. It relies on the frequency of the keywords. If keywords are not properly represented, Naive Bayes classifier cannot properly produce the correct result. We also notice from our experiments that when using Naive Bayes classifiers, we need to choose our training data more carefully. For instance, we should correct non-common misspellings as well as somehow standardize abbreviations that are used.

References

[1] Machine Learning, Tom Mitchell. ISBN-10: 0070428077 — ISBN-13: 978-0070428072.

[2] *Expected value*, available at `https://stackoverflow.com`.