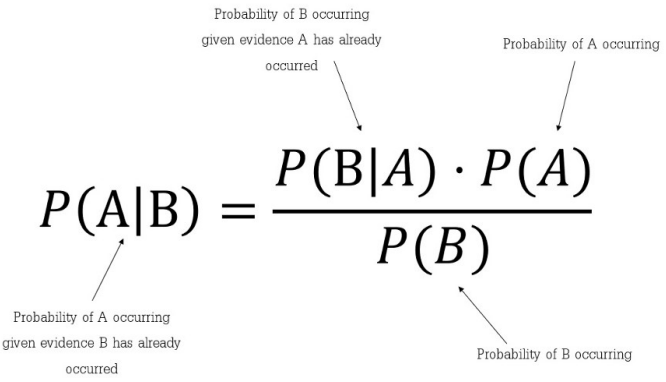
S21CSE471: Statistical Methods in AI **Spring 2021**

Naive Bayes and Multinomial Text Classification(4/Mar/2021)

Prepared by: SANYAM : 2020201006, NAVYA : 2020201033, SACHIN : 2020201073

1 Introduction to Bayes' Theorem

Bayes Theorem is a method to determine conditional probabilities i.e, the probability of one event occurring given that another event has already occurred. It is the principled way of calculating a conditional probability without the joint probability. This is useful either when the joint probability is challenging to calculate (as it is most of the time), or when the reverse conditional probability is available or easy to calculate.



It is often the case that we do not have access to the denominator directly, e.g. P(B), which is calculated in alternative way,

$$P(B) = P(B|A) * P(A) + P(B|\bar{A}) * P(\bar{A})$$

This gives a formulation of Bayes Theorem that we can use that uses the alternate calculation of P(B), described below:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B|A) * P(A) + P(B|\bar{A}) * P(\bar{A})}$$

Now, Let's Name the Terms:

- $P(A|B)$: Posterior Probability
- P(A) : Prior Probability
- $P(B|A)$: Likelihood
- P(B) : Evidence

This allows Bayes Theorem to be restated as:

Posterior = Likelihood * Prior / Evidence

1.1 Applications of Bayes' Theorem

Bayes Decision Theory is a Statistical Approach to the problem of Classification. Here, It is assumed that the underlying probability distribution for the categories is known in prior. Thus, we obtain an ideal Bayes Classifier against which all other classifiers are judged for performance. The three main Applications are:

- Naive Bayes' Classifiers
- Discriminant Functions and Decision Surfaces
- Bayesian Parameter Estimation

Let us Discuss Naive Bayes' Classifier in Detail.

2 Naive Bayes' Classifier

It is a machine learning algorithm which is used to solve Classification Problems.

Suppose you have to solve a classification problem and have created the features and generated the hypothesis, but your superiors want to see the model. You have numerous data points (lakhs of data points) and many variables to train the dataset. The best solution for this situation would be to use the Naive Bayes classifier, which is quite faster in comparison to other classification algorithms.

Naive Bayes' Theorem assumes that all features or predictors are independent.In other words, this classifier assumes that the presence of one particular feature in a class doesn't affect the presence of another one.

2.1 Types

- Gaussian Naive Bayes Classifier : This follows Gaussian normal distribution and supports continuous data.
- Multinomial Naive Bayes Classifier: It is used when features follow a multinomial distribution.
- Bernoulli Naive Bayes classifier: It is used when features are of the boolean type.

2.2 Derivation

As we already know that, Equation for Bayes Theorem would be:

$$P(class|X) = P(X|class)P(class)/P(X)$$

A class variable is something that the classifier is trying to classify. For instance, when trying to classify an email as spam or not, "is spam" is the class variable.

In the equation above, *class* is the class variable and X is the set of features. $X = x_1, x_2, ...x_n$ The above formula can be rewritten as:

$$P(class|x_1...x_n) = P(x_1|class)...P(x_n|class)P(class)/P(x_1)...P(x_n)$$

$$P(class|x_1...x_n) \propto P(x_1|class)...P(x_n|class)P(class)$$

Notice that for all entries in the given dataset, the denominator will not change. Hence, the denominator can be ignored.

For all outcomes of the class variable, the class variable with the maximum probability needs to be found using:

$$class = argmax(P(x_1|class)...P(x_n|class)P(class))$$

2.3 Applications

This algorithm is commonly applied to Text Classification.

- (Automatic) Classification of emails in folders, so incoming email messages go into folders such as: "Family", "Friends", "Updates", "Promotions", etc.
- (Automatic) Tagging of job listings. Given a job listing in raw text format, we can assign it tags such as: "software development", "design", "marketing", etc.
- (Automatic) Categorization of products. Given a product description, we can assign it into categories such as: "Books", "Electronics", "Clothing", etc.

Let us Consider an Example Now:

We need to Classify E-mails into two classes, i.e Spam and Not-Spam.The only prerequisite is to have an existing training set of e-mails known to be Spam and a training set of emails known to be Not-Spam.

The Below Images contain Spam and Not-Spam Email Content which are considered for training the dataset.
Step 1: We need to Computer Term-Document Matrix for each Class.
1. Term-Document Matrix consists of a list of word frequencies appearing in a set of documents.
2. The TDM matrix is a sparse rectangular matrix of n words and m documents. And it's said to be sparse because it contains mostly zeros. The entry (i,j) of the TDM matrix represents the frequency of word "i" in document "j".
The below images **(Fig.9)** are an excerpt of the TDM matrix for the Not-Spam and Spam classes:

Step 2: Compute Frequencies
Once the TDM matrices are computed for each class, the next step is to compute the frequency and occurrence of each term. Refer to **Fig.10** for Frequency and Occurrence.
1. From raw frequency count, you can infer that Spam messages contain in high frequency terms such as "free", whereas Ham or Not-SPAM messages contain in high frequency terms such as "exam". This is how the classifier will be able to tell apart one class from the other.
2. The "frequency" column is the number of times each term appeared in all documents, that is, the sum of the columns of the TDM matrix. The "occurrence" column is the percentage of time each term appeared on all the documents.
3. Both the "frequency" and the "occurrence" columns could be used to compute probability estimates,

```
Dear Professor Hanks,

I am a student in your class.

After getting the results of last week's exam, I realized that I am struggling
with more than one topic in the course. I want to do my best in this class, and
would like to review my exam with you. Is there a time we could meet later this
week to talk further?

I look forward to your reply.
Best regards,
Soo Kim
```

Figure 1: Not Spam Email

```
Dear Professor Alice,

I'm writing to follow up on my email I sent earlier this week regarding a
question I have about the topic of assignment and the exam. I look forward to
hearing from you.

Best regards,
Michael Kumar
```

Figure 2: Not Spam Email

```
Dear Professor Foo,

I am a student in your machine learning class.

I have a question about the second term project and I was not able to find the
answer on the syllabus. Should our project be only about the topics listed on
the second part of the syllabus, or can I incorporate topics from the whole
course, as long as it fits with the subject of the class?

I look forward to hearing from you.
Best regards,
Bar
```

Figure 3: Not Spam Email

```
After many attempts of trying to make money on the Internet I decided to
give this one a try!

I have been through this program once already.  You have no idea how
profitable it is!  It is definitely the fastest thing available.  We are
talking about thousands of dollars in 2 weeks' time...send it to as many
people as you can (even though it says 20...trust me).
GOOD LUCK...you can't lose!!
```

Figure 4: Spam Email

```
We will Run a FREE Mortgage Analysis For you.
You are under NO Obligation for anything.
The Analysis Will tell you exactly how many yrs you can take off
your Mortgage.
And how much equity you can build and in what period of time.
Just Return an email With These Questions Completed
and your Free Analysis will be E-mailed to you within 5 business
days.
```

Figure 5: Spam Email

```
We will give you $1,000 for sending an e-mail to your friends.  AB Mailing,
Inc. is proud to announce the start of a new contest.  Each day until
January, 31 1999, one lucky Internet or AOL user who forwards our
advertisement to their friends will be randomly picked to receive $1,000!
You could be the winner!

Thank you for your time.
```

Figure 6: Spam Email

however, the "occurrence" column is preferred as it is bounded between zero and one.

Step 3: Apply Naive Bayes' Rule
Now, The Probability we want to compute is:

$$P(Span/\mathbf{x}) = \frac{P(\mathbf{x}/Spam) * P(Spam)}{P(\mathbf{x})}$$

Where $\mathbf{x}$ is a feature vector containing the words coming from the Spam (or Ham) emails:

$$\mathbf{x} = [w_1, w_2, w_3, ..., w_n]$$

The "Naive" assumption that the Naive Bayes classifier makes is that the probability of observing a word is independent of each other. The result is that the "likelihood" is the product of the individual probabilities of seeing each word in the set of Spam or Ham emails. We calculated these probabilities in Step 2 and stored them in the "occurrence" column. Formally:

$$P(Spam/w_1, ..., w_n) \propto P(Spam)\Pi_{i=1}^{n} P(w_i/Spam) = P(Spam)*P(w_1/Spam)*P(w_2/Spam)*P(w_3/Spam)....$$

1. Notice that the "evidence" is a constant factor depending only on the features, thus the introduction of the proportionality symbol for the "likelihood" computation in the previous equation.
2. Furthermore, since our dataset has 3 Spam emails, and 3 Ham emails, we know that the "prior" P(Spam) is 50%.
3. In summary, since we have all necessary terms in the Bayes theorem ("likelihood", "prior", and "evidence") we can proceed to compute "posterior" probabilities. Remember that, the main assumption of the Naive Bayes classifier is that each feature (word) is independent of each other.

Step 4. Compute the probability of an incoming email being Spam or Not-Spam
The formal decision rule is:

$$\hat{y} = argmax_{k\epsilon\{Spam,Ham\}} p(C_k)\Pi_{i=1}^{n} p(x_i|C_k)$$

It means that,for every incoming e-mail we have to compute the probability of such e-mail being Spam and Ham/Not-Spam (i.e. for each class), and our final verdict will be given by the largest probability.

Figure 7: Test Email for Classification

The probabilities for each class, assuming an arbitrarily small probability of 1e-2 for words that are not in our training set, are:

$$\hat{(y)}_{Spam} = 5.5e - 64$$

$$\hat{(y)}_{Ham} = 9.1e - 48$$

which indicates that Given Email is Not-Spam.

2.4 Gaussian Naive Bayes' Classifier

When working with continuous data, an assumption often taken is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. The likelihood of the features is assumed to be:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution.

An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all what is needed to define such a distribution.

The below illustration **(Fig.8)** indicates how a Gaussian Naive Bayes (GNB) classifier works. At every data point, the z-score distance between that point and each class-mean is calculated, namely the distance from the class mean divided by the standard deviation of that class.

Thus, we see that the Gaussian Naive Bayes has a slightly different approach and can be used efficiently.

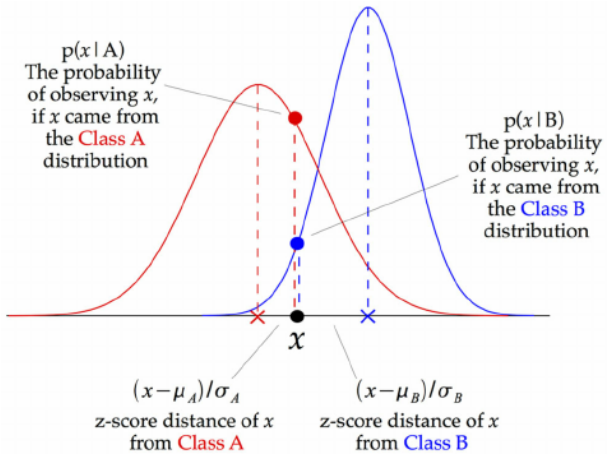


Figure 8: Gaussian Naive Bayes' Classifier

Terms	ham_train1.txt	ham_train2.txt	ham_train3.txt
able	1	0	0
answer	1	0	0
bar	1	0	0
best	1	1	2
can	1	0	0
...			
struggling	0	0	1
talk	0	0	1
time	0	0	1
want	0	0	1
weeks	0	0	1

Terms	spam_train1.txt	spam_train2.txt	spam_train3.txt
analysis	3	0	0
anything	1	0	0
build	1	0	0
business	1	0	0
can	2	0	1
...			
timesend	0	0	1
trust	0	0	1
try	0	0	1
trying	0	0	1
weeks	0	0	1

Term document matrix excerpts for the HAM and SPAM emails.

Figure 9: TDM Images for Not-Spam and Spam Emails

	term frequency	occurrence
best	4	1.0000000
class	4	0.6666667
dear	3	1.0000000
forward	3	1.0000000
look	3	1.0000000
professor	3	1.0000000
regards	3	1.0000000
exam	3	0.6666667
course	2	0.6666667
hearing	2	0.6666667

	term frequency	occurrence
will	5	0.6666667
analysis	3	0.3333333
can	3	0.6666667
many	3	0.6666667
email	2	0.6666667
free	2	0.3333333
mortgage	2	0.3333333
time	2	0.6666667
friends	2	0.3333333
give	2	0.6666667

Figure 10: Top 10 terms sorted by frequency for the Not-SPAM and SPAM classes.

3 Multinomial Naive Bayes Classifier

- It is one of the most popular applications of machine learning is the analysis of categorical data, specifically text data.With an ever-growing amount of textual information stored in electronic form such as legal documents, policies, company strategies, etc., automatic text classification is becoming increasingly important. This requires a supervised learning technique that classifies every new document by assigning one or more class labels from a fixed or predefined class.

  Implementation of the algorithm is as follows:-

  1. First we calculate the fraction of documents in each class:

$$\pi_{\mathrm{j}} = \frac{class_{\mathrm{j}}}{\sum_{i=1}^{n} class_{\mathrm{i}}} \tag{1}$$

  2. For calculating our probability, we will find the average of each word for a given class. For class j and word i, the average is given by:

$$P(i/j) = \frac{word_{\mathrm{ij}}}{word_{\mathrm{j}}} \tag{2}$$

  3. Combining probability distribution of P with fraction of documents belonging to each class.

$$Pr(j) \propto \pi_{\mathrm{j}} \prod_{i=a}^{|v|} Pr(i/j)^{f_i} \tag{3}$$

  In order to avoid underflow, we will use the sum of logs:

$$Pr(j) \propto log(\pi_{\mathrm{j}} \prod_{i=a}^{|v|} Pr(i/j)^{f_i}) \tag{4}$$

$$Pr(j) = log\pi_{\mathrm{j}} + \sum_{i=1}^{|V|} \mathrm{f}_i log(Pr(i/j))(5)$$

  One issue is that, if a word appears again, the probability of it appearing again goes up. In order to smooth this, we take the log of the frequency:

$$Pr(j) = log\pi_{\mathrm{j}} + \sum_{i=1}^{|V|} \log(1+\mathrm{f}_i) log(Pr(i/j))(6)$$

  Hence, our optimal model is:

$$Pr(j) = log\pi_{\mathrm{j}} + \sum_{i=1}^{|V|} \log(1+\mathrm{f}_i) log(Pr(i/j))(7)$$

4 Example for Multinomial Naive Bayes

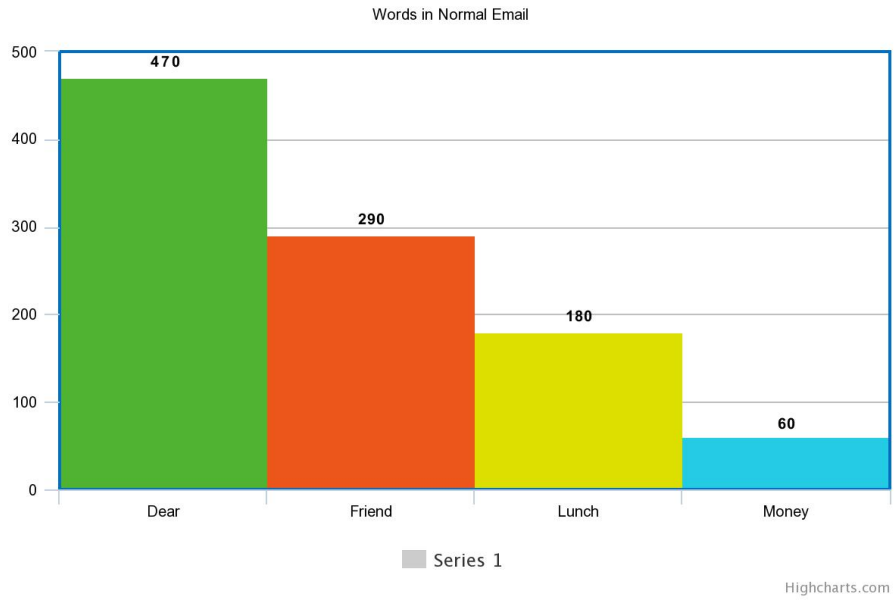
1. Text Classification

   With an ever-growing amount of textual information in electronic media, its important to classify the documents based on the textual information. This requires a supervised learning technique that classifies every new document by assigning one or more class labels from a fixed or predefined class.

   Naive Bayes, which is computationally very efficient and easy to implement, is a learning algorithm frequently used in text classification problems.

Let us take an example of spam filtering process using Multinomial Naive Bayes Classification. Now, imagine we received a lot of emails from friends, family, office and we also received spam (unwanted messages that are usually scams or unsolicited advertisements).

Let see the histogram of all the words that occur in the normal messages from family and friends. We can use the histogram to calculate the probabilities of seeing each word, given that it was a normal message.



Here we are assuming that we have 4 words in our Vocabulary {Dear,Friend,Lunch,Money} along with their frequencies {470,290,180,60} respectively.

Lets denote number of such Normal Emails as N where $N = 1000$ out of total 1500 Emails. So the probabilities of words given that we saw in normal message are-

$$P(Dear|N) = \frac{470}{1000} = 0.47$$

$$P(Friend|N) = \frac{290}{1000} = 0.29$$

$$P(Lunch|N) = \frac{180}{1000} = 0.18$$

$$P(Money|N) = \frac{60}{1000} = 0.06$$

Now let's make the histogram of all the words in spam.

Lets denote number of such Spam Emails as S where $S = 500$ out of total 1500 Emails. So the probabilities of words given that we saw in normal message are-

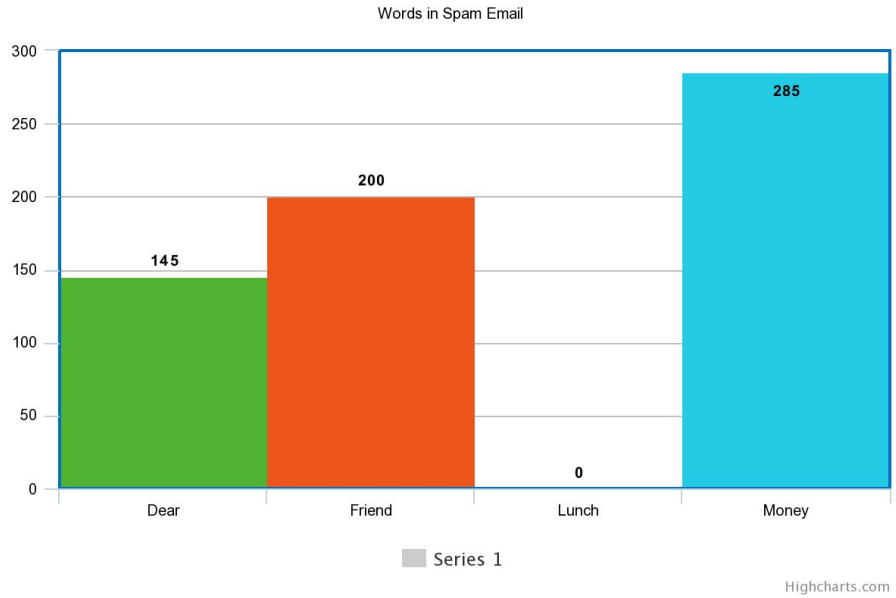
$$P(Dear|S) = \frac{145}{500} = 0.29$$

$$P(Friend|S) = \frac{200}{500} = 0.4$$

$$P(Lunch|S) = \frac{0}{500} = 0$$

$$P(Money|S) = \frac{285}{500} = 0.57$$

These Probabilities are also called **Likelihoods**.



Now, let's say we have received a normal message as **Dear Friend** and we want to find out if it's a normal message or spam. Therefore, the prior Probabilities are:

$$P(N) = 1000/1500 = 0.67$$

$$P(S) = 500/1500 = 0.33$$

We multiply this prior probability with the probabilities of Dear Friend that we have calculated earlier.

$$P(N|\textbf{Dear Friend}) = \frac{P(\textbf{Dear}|N) * P(\textbf{Friend}|N) * P(N)}{P(\textbf{Dear Friend})}$$

$$P(S|\textbf{Dear Friend}) = \frac{P(\textbf{Dear}|S) * P(\textbf{Friend}|S) * P(S)}{P(\textbf{Dear Friend})}$$

We can ignore the denominator term as they both are same for Normal as well as Spam classification. Hence our equation becomes

$$P(N|\textbf{Dear Friend}) \propto P(\textbf{Dear}|N) * P(\textbf{Friend}|N) * P(N)$$

$$P(N|\textbf{Dear Friend}) \propto 0.47 * 0.29 * 0.67 = 0.09$$

$$P(S|\textbf{Dear Friend}) \propto P(\textbf{Dear}|S) * P(\textbf{Friend}|S) * P(S)$$

$$P(S|\textbf{Dear Friend}) \propto 0.29 * 0.14 * 0.33 = 0.01$$

The probability score of **Dear Friend** being a *normal message* is greater than the probability score of **Dear Friend** being *spam*. We can conclude that **Dear Friend** is a *normal message*.

Lets take another message as example, x =**Lunch Money Money Money**.

$$P(N|\mathbf{x}) \propto P(\textbf{Lunch}|N) * P(\textbf{Money}|N)^3 * P(N) = 0.000002$$

$$P(S|\mathbf{x}) \propto P(\textbf{Lunch}|S) * P(\textbf{Money}|S)^3 * P(S) = 0$$

As we can see that we got 0 probability for the message being classified as Spam, so to avoid that we assign a minimum value (Let say $\alpha = 0.01$) to all the words in our vocabulary being classified as spam. Therefore,

$$P(Dear|S) = \frac{145}{500} = 0.29 + 0.01 = 0.3$$

$$P(Friend|S) = \frac{200}{500} = 0.4 + 0.01 = 0.4$$

$$P(Lunch|S) = \frac{0}{500} = 0 + 0.01 = 0.01$$

$$P(Money|S) = \frac{285}{500} = 0.57 + 0.01 = 0.58$$

Now,

$$P(\textbf{Lunch}|S) * P(\textbf{Money}|S)^3 * P(S) = 0.01 * (0.58)^3 * 0.33 = 0.000643$$

The probability score of **Lunch Money Money Money** being a *spam message* is greater than the probability score of **Lunch Money Money Money** being *normal*. We can conclude that **Lunch Money Money Money** is a *spam message*.

References

[1] https://machinelearningmastery.com/naive-bayes-for-machine-learning

[2] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[3] https://towardsdatascience.com/multinomial-naive-bayes-classifier-for-text-analysis-python-8dd6825ece67

[4] https://www.mygreatlearning.com/blog/multinomial-naive-bayes-explained