A Tutorial on Naive Bayes Classification

Choochart Haruechaiyasak
(Last update: 16 August 2008)

Naive Bayes is a simple probabilistic classifier based on applying Bayes' theorem (or Bayes's rule) with strong independence (naive) assumptions. More details can be found on Wikipedia Web site : http://en.wikipedia.org/wiki/Naive_bayes

Explanation of Bayes's rule:

Bayes's rule: $$P(H \mid E) = \frac{P(E \mid H) \times P(H)}{P(E)}$$

The basic idea of Bayes's rule is that the outcome of a hypothesis or an event (H) can be predicted based on some evidences (E) that can be observed. From Bayes's rule, we have

(1) A ***priori*** probability of H or P(H): This is the probability of an event ***before*** the evidence is observed.
(2) A ***posterior*** probability of H or P(H | E): This is the probability of an event ***after*** the evidence is observed.

Example 1: To predict the chance or the probability of raining, we usually use some evidences such as the amount of dark cloud in the area.

Let H be the event of raining and E be the evidence of dark cloud, then we have

$$P(\text{raining} \mid \text{dark cloud}) = \frac{P(\text{dark cloud} \mid \text{raining}) \times P(\text{raining})}{P(\text{dark cloud})}$$

- P(dark cloud | raining) is the probability that there is dark cloud when it rains. Of course, "dark cloud" could occur in many other events such as overcast day or forest fire, but we only consider "dark cloud" in the context of event "raining". This probability can be obtained from historical data recorded by some meteorologists.
- P(raining) is the *priori* probability of raining. This probability can be obtained from statistical record, for example, the number of rainy days throughout a year.
- P(dark cloud) is the probability of the evidence "dark cloud" occurring. Again, this can be obtained from the statistical records, but the evidence is not usually well recorded compared to the main event. Therefore, sometimes the full evidence, i.e., P(dark cloud), is hard to obtain.

Explanation of Naive Bayes:

As you can see from Example 1, we can predict an outcome of some events by observing some evidences. Generally, it is "better" to have more than one evidence to support the prediction of an event. Typically, the more evidences we can gather, the better the classification accuracy can be obtained. However, the evidence must relate to the event (must make sense). For example, if you add an evidence of "earthquake" to Example 1, the above model might yield worse performance. This is since "raining" is not related to the evidence of "earthquake", i.e., if there is an earthquake, it doesn't mean that it will rain.

Suppose we have more than one evidence for building our NB model, we could run into a problem of dependencies, i.e., some evidence may depend on one or more of other evidences. For example, the evidence "dark cloud" directly depends on the evidence "high humidity". However, including dependencies into the model will make it very complicated. This is because one evidence could depend on many other evidences. To make our life easier, we make an assumption that all evidences are independent of each other (this is why we call the model "naive").

(**Note:** The complete Bayes's rule without the "independence assumption is called "Bayesian Network". If you're interested, you can read more from http://en.wikipedia.org/wiki/Bayesian_network)

Bayes's rule for multiple evidences::

$$P(H \mid E_1, E_2, ..., E_n) = \frac{P(E_1, E_2, ..., E_n \mid H) \times P(H)}{P(E_1, E_2, ..., E_n)}$$

With the *independence* assumption, we can rewrite the Bayes's rule as follows:

$$P(H \mid E_1, E_2, ..., E_n) = \frac{P(E_1 \mid H) \times P(E_2 \mid H) \times ... P(E_n \mid H) \times P(H)}{P(E_1, E_2, ..., E_n)}$$

Example 2: From Example 1, we could have the following NB model for raining,

$$P(\text{raining} \mid \text{dark cloud, wind speed, humidity}) = \frac{P(\text{dark cloud} \mid \text{raining}) \times P(\text{wind speed} \mid \text{raining}) \times P(\text{humidity} \mid \text{raining}) \times P(\text{raining})}{P(\text{dark cloud, wind speed, humidity})}$$

Example 3: To understand how to build an NB model, let's use the example from our lecture slides (Chapter 3 of the data mining book). Given the weather data set for predicting play condition. There are 14 instances (or examples) and 5 attributes. All attributes are nominal.

	outlook	temperature	humidity	windy	play
1	sunny	hot	high	false	no
2	sunny	hot	high	true	no
3	overcast	hot	high	false	yes
4	rainy	mild	high	false	yes
5	rainy	cool	normal	false	yes
6	rainy	cool	normal	true	no
7	overcast	cool	normal	true	yes
8	sunny	mild	high	false	no
9	sunny	cool	normal	false	yes
10	rainy	mild	normal	false	yes
11	sunny	mild	normal	true	yes
12	overcast	mild	high	true	yes
13	overcast	hot	normal	false	yes
14	rainy	mild	high	true	no

We need to build the NB model from the given above data set. The result are shown below:

Outlook			Temperature			Humidity			Windy			Play	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

The top part of the table contains the frequency of different evidences. For example, there are 2 instances (examples) from the data set showing (outlook=sunny) when (play=yes). Once you have finished counting all frequency, we need to build the NB model by calculating all P(E | H) and P(H). For example,

P(outlook=sunny | play=yes) = 2/9
P(play=yes) = 9/14

Once we have the NB model, we can use it to predict the event "play" based on different set of evidences. For example, if we observe (outlook=sunny), (temperature=cool), (humidity=high) and (windy=true), then we can estimate the posterior probability as follows:

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

← *Evidence E*

$$\Pr[yes \mid E] = \Pr[Outlook = Sunny \mid yes] \times \Pr[Temperature = Cool \mid yes] \times \Pr[Humidity = High \mid yes] \times \Pr[Windy = True \mid yes] \times \frac{\Pr[yes]}{\Pr[E]}$$

Probability of class "yes"

$$= \frac{\frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{9}{14}}{\Pr[E]}$$

We can ignore Pr(E) because we only need to "relatively" compare the value to other class. Therefore we have the following results:

Likelihood of the two classes

For "yes" = 2/9 × 3/9 × 3/9 × 3/9 × 9/14 = 0.0053

For "no" = 3/5 × 1/5 × 4/5 × 3/5 × 5/14 = 0.0206

Conversion into a probability by normalization:

P("yes") = 0.0053 / (0.0053 + 0.0206) = 0.205

P("no") = 0.0206 / (0.0053 + 0.0206) = 0.795

Solving "zero-frequency" problem with smoothing technique:

If you observe from the previous NB model table, P(outlook=overcast | play=no) = 0/5. This will create a problem when we calculate for P("no"), since the result will be equal to zero. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.

For example, for attribute "outlook" when play=no, we can apply Laplace estimation as follows:

$$P(\text{outlook=sunny} \mid \text{play=no}) = \frac{3 + \mu p_1}{5 + \mu}$$

$$P(\text{outlook=overcast} \mid \text{play=no}) = \frac{0 + \mu p_2}{5 + \mu}$$

$$P(\text{outlook=overcast} \mid \text{play=no}) = \frac{2 + \mu p_3}{5 + \mu}$$

where $(p_1 + p_2 + p_3) = 1.0$

By assuming that all evidences are equally distributed, $p_1 = p_2 = p_3 = 1/3$

$$P(\text{outlook=sunny} \mid \text{play=no}) = \frac{3 + \mu/3}{5 + \mu} = \frac{3 + 3/3}{5 + 3} = 4/8$$

$$P(\text{outlook=overcast} \mid \text{play=no}) = \frac{0 + \mu/3}{5 + \mu} = \frac{0 + 3/3}{5 + 3} = 1/8$$

$$P(\text{outlook=raining} \mid \text{play=no}) = \frac{2 + \mu/3}{5 + \mu} = \frac{2 + 3/3}{5 + 3} = 3/8$$

Naive Bayes on WEKA's Explorer:

We can use WEKA (http://www.cs.waikato.ac.nz/ml/weka/) machine learning software tool to generate and test NB model automatically. To generate the model, follow these steps.

(1) Run WEKA, click on "Explorer" mode button..
(2) Click on "Open files ..." tab and select the file "weather.nominal.arff" from subdirectory "data".
(3) Click on "Classify" tab and click on "choose" button. Then select the algorithm "weka/classifiers/bayes/NaiveBayes/Simple".
(4) To generate the model and test, click on the "Start" button.
(5) The tool automatically generates model as follows:

```
=== Classifier model (full training set) ===
Naive Bayes (simple)

Class yes: P(C) = 0.625

Attribute outlook
sunny  overcast     rainy
0.25         0.41666667   0.33333333
Attribute temperature
hot    mild   cool
0.25         0.41666667   0.33333333
Attribute humidity
high   normal
0.36363636   0.63636364
Attribute windy
TRUE   FALSE
0.36363636   0.63636364

Class no: P(C) = 0.375

Attribute outlook
sunny  overcast     rainy
0.5          0.125        0.375
Attribute temperature
hot    mild   cool
0.375        0.375        0.25
Attribute humidity
high   normal
0.71428571   0.28571429
Attribute windy
TRUE   FALSE
0.57142857   0.42857143
```

Note: NaiveBayesSimple uses Laplace estimation technique to avoid the zero frequency problem. The result matches the one in the example.

Naive Bayes for Text Classification:

Although Naive Bayes uses the independence assumption, the model is widely used in many applications. Some of the interesting applications are text classification and information filtering (such as spam filtering). One of the main reasons that NB model works well for text domain because the evidences are "vocabularies" or "words" appearing in texts and the size of the vocabularies is typically in the range of thousands. The large size of evidences (or vocabularies) makes NB model work well for text classification problem.

For text domain, we can build the NB model in the similar fashion as in the previous example. Let's take a look at the following example.

Example 4: Text classification using NB model.

Consider the following data set. We have 6 documents D0 ... D5 as the training data set. Suppose we extract and consider only 6 vocabularies from all documents. There are two classes (categories) of documents: "terrorism" and "entertainment". Documents are preprocessed and shown in the following table. The numbers are the frequency of the word in the documents. For example, the word "kill" occurs twice in the document D0.

Training Doc	kill	bomb	kidnap	music	movie	TV	C
D0	2	1	3	0	0	1	Terrorism
D1	1	1	1	0	0	0	Terrorism
D2	1	1	2	0	1	0	Terrorism
D3	0	1	0	2	1	1	Entertainment
D4	0	0	1	1	1	0	Entertainment
D5	0	0	0	2	2	0	Entertainment

Phase 1: Building the NB model

The NB model for the above data set is as shown below:

|V|	C	P(Ci)	ni	P(kill |Ci)	P(bomb |Ci)	P(kidnap |Ci)	P(music |Ci)	P(movie |Ci)	P(TV |Ci)
6	Terrorism	0.5	15	0.238095238	0.19047619	0.333333333	0.047619048	0.095238095	0.095238095
	Entertainment	0.5	12	0.055555556	0.111111111	0.111111111	0.333333333	0.277777778	0.111111111

|V| = the number of vocabularies

$P(c_i)$ = the priori probability of each class = number of documents in a class / number of all documents

P (Terrorism) = 3/6 = 0.5

P (Entertainment) = 3/6 = 0.5

ni = the total number of word frequency of each class

$n_{Terrorism}$ = 2+1+3+1+1+1+1+1+1+2+1 = 15

$n_{Entertainment}$ = 1+2+1+1+1+1+1+2+2 = 12

$P(w_i \mid c_i)$ = the conditional probability of keyword occurrence given a class

For example, P (kill | Terrorism) = (2 + 1 + 1) / 15 = 4/15

P (kill | Entertainment) = (0 + 0 + 0) / 12 = 0/12

To avoid the "zero frequency" problem, we apply Laplace estimation by assuming a uniform distribution over all words as follows:

P (kill | Terrorism) = (2 + 1 + 1 + 1) / (15 + |V|) = 5/21 = 0.2380

P (kill | Entertainment) = (0 + 0 + 0 + 1) / (12 + |V|) = 1/18 = 0.0555

Phase 2: Classifying a test document

To classify a test document Dt, we have to calculate the "posterior" probabilities, P(c | W) for each class as follows:

Test Doc	kill	bomb	kidnap	music	movie	TV	C
Dt	2	1	2	0	0	1	?

$$P(c_i \mid W) = P(c_i) \times \prod_{j=1}^{V} P(wj|ci)$$

P(Terrorism | W) = P(Terrorism) x P(kill | Terrorism) x P(bomb | Terrorism) x P(kidnap | Terrorism) x P(music | Terrorism) x P(movie | Terrorism) x P(TV | Terrorism)

$= 0.5 \times 0.2380^2 \times 0.1904^1 \times 0.3333^2 \times 0.0476^0 \times 0.0952^0 \times 0.0952^1$

$= 0.5 \times 0.0566 \times 0.1904 \times 0.1110 \times 1 \times 1 \times 0.0952$

$= \underline{5.7 \times 10^{-5}}$

P(Entertainment | W) = P(Entertainment) x P(kill | Entertainment) x P(bomb | Entertainment) x P(kidnap | Entertainment) x P(music | Entertainment) x P(movie | Entertainment) x P(TV | Terrorism)

$= 0.5 \times 0.0555^2 \times 0.1111^1 \times 0.1111^2 \times 0.3333^0 \times 0.2777^0 \times 0.1111^1$

$= 0.5 \times 0.0030 \times 0.1111 \times 0.0123 \times 1 \times 1 \times 0.1111$

$= \underline{2.27 \times 10^{-7}}$

Since P(Terrorism | W) has the highest value, therefore Dt is classified into "Terrorism". This makes sense because Dt contains many words related to terrorism such as "kill", "bomb" and "kidnap".

Underflow prevention:

As you can observe from the above example, the "posterior" probability value is very small. Typically the number of conditional probabilities is in the range of thousands or more (number of words appearing in a document collection), the value will be too low for the CPU to handle. This problem is referred to as the *underflow* problem. To solve this problem, we can take logarithm on the probabilities as follows:

$P(\text{Terrorism} \mid W) = \log(0.5 \times 0.2380^2 \times 0.1904^1 \times 0.3333^2 \times 0.0476^0 \times 0.0952^0 \times 0.0952^1)$

$= \log(0.5) + 2\log(0.2380) + 1\log(0.1904) + 2\log(0.3333) + 0\log(0.0476) + 0\log(0.0952) + 1\log(0.0952)$

$= -0.3010 - 1.2468 - 0.7203 - 0.9543 + 0 + 0 - 1.0213$

$= \underline{-4.2437}$

$P(\text{Entertainment} \mid W) = \log(0.5 \times 0.0555^2 \times 0.1111^1 \times 0.1111^2 \times 0.3333^0 \times 0.2777^0 \times 0.1111^1)$

$= \log(0.5) + 2\log(0.0555) + 1\log(0.1111) + 2\log(0.1111) + 0\log(0.3333) + 0\log(0.2777) + 1\log(0.1111)$

$= -0.3010 - 2.511 - 0.9542 - 1.9085 + 0 + 0 - 0.9542$

$= \underline{-6.6289}$

Again, since P(Terrorism | W) has the higher value, therefore Dt is classified into "Terrorism". You can also observe that the final calculated values were scaled nicely so that the "underflow" problem is avoided.

Note: We can use the following property of logarithm: $\log(x*y) = \log(x) + \log(y)$ and $\log x^y = y * \log(x)$