# Naive Bayes classifier

A **Naive Bayes classifier** is a simple probabilistic classifier based on applying Bayes' theorem (from Bayesian statistics) with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model".

In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 4" in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple.

Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods.

In spite of their naive design and apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, analysis of the Bayesian classification problem has shown that there are some theoretical reasons for the apparently unreasonable efficacy of naive Bayes classifiers.[1] Still, a comprehensive comparison with other classification methods in 2006 showed that Bayes classification is outperformed by more current approaches, such as boosted trees or random forests.[2]

An advantage of the naive Bayes classifier is that it only requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

## The naive Bayes probabilistic model

Abstractly, the probability model for a classifier is a conditional model

$$p(C|F_1,\ldots,F_n)$$

over a dependent class variable $C$ with a small number of outcomes or *classes*, conditional on several feature variables $F_1$ through $F_n$. The problem is that if the number of features $n$ is large or when a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable.

Using Bayes' theorem, we write

$$p(C|F_1,\ldots,F_n) = \frac{p(C)\ p(F_1,\ldots,F_n|C)}{p(F_1,\ldots,F_n)}.$$

In plain English the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}.$$

In practice we are only interested in the numerator of that fraction, since the denominator does not depend on $C$ and the values of the features $F_i$ are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model

$$p(C, F_1,\ldots,F_n)$$

which can be rewritten as follows, using repeated applications of the definition of conditional probability:

$$p(C, F_1,\ldots,F_n)$$

$$
\begin{aligned}
&= p(C)\ p(F_1, \ldots, F_n|C) \\
&= p(C)\ p(F_1|C)\ p(F_2, \ldots, F_n|C, F_1) \\
&= p(C)\ p(F_1|C)\ p(F_2|C, F_1)\ p(F_3, \ldots, F_n|C, F_1, F_2) \\
&= p(C)\ p(F_1|C)\ p(F_2|C, F_1)\ p(F_3|C, F_1, F_2)\ p(F_4, \ldots, F_n|C, F_1, F_2, F_3) \\
&= p(C)\ p(F_1|C)\ p(F_2|C, F_1)\ p(F_3|C, F_1, F_2)\ \ldots p(F_n|C, F_1, F_2, F_3, \ldots, F_{n-1}).
\end{aligned}
$$

Now the "naive" conditional independence assumptions come into play: assume that each feature $F_i$ is conditionally independent of every other feature $F_j$ for $j \neq i$. This means that

$$p(F_i|C, F_j) = p(F_i|C)$$

for $i \neq j$, and so the joint model can be expressed as

$$
\begin{aligned}
&p(C, F_1, \ldots, F_n) = p(C)\ p(F_1|C)\ p(F_2|C)\ p(F_3|C)\ \cdots \\
&= p(C) \prod_{i=1}^{n} p(F_i|C).
\end{aligned}
$$

This means that under the above independence assumptions, the conditional distribution over the class variable $C$ can be expressed like this:

$$p(C|F_1, \ldots, F_n) = \frac{1}{Z} p(C) \prod_{i=1}^{n} p(F_i|C)$$

where $Z$ (the evidence) is a scaling factor dependent only on $F_1, \ldots, F_n$, i.e., a constant if the values of the feature variables are known.

Models of this form are much more manageable, since they factor into a so-called *class prior* $p(C)$ and independent probability distributions $p(F_i|C)$. If there are $k$ classes and if a model for each $p(F_i|C = c)$ can be expressed in terms of $r$ parameters, then the corresponding naive Bayes model has (*k* − 1) + *n r k* parameters. In practice, often $k = 2$ (binary classification) and $r = 1$ (Bernoulli variables as features) are common, and so the total number of parameters of the naive Bayes model is $2n + 1$, where $n$ is the number of binary features used for classification and prediction

## Parameter estimation

All model parameters (*i.e.*, class priors and feature probability distributions) can be approximated with relative frequencies from the training set. These are maximum likelihood estimates of the probabilities. A class' prior may be calculated by assuming equiprobable classes (i.e., priors = 1 / (number of classes)), or by calculating an estimate for the class probability from the training set (i.e., (prior for a given class) = (number of samples in the class) / (total number of samples)). To estimate the parameters for a feature's distribution, one must assume a distribution or generate nonparametric models for the features from the training set. [3] If one is dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a Gaussian distribution.

For example, suppose the training data contains a continuous attribute, $x$. We first segment the data by the class, and then compute the mean and variance of $x$ in each class. Let $\mu_c$ be the mean of the values in $x$ associated with class *c*, and let $\sigma_c^2$ be the variance of the values in $x$ associated with class *c*. Then, the probability of some value given a class, $P(x = v|c)$, can be computed by plugging $v$ into the equation for a Normal distribution parameterized by $\mu_c$ and $\sigma_c^2$. That is,

$$P(x = v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}}\, e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$

Another common technique for handling continuous values is to use binning to discretize the values. In general, the distribution method is a better choice if there is a small amount of training data, or if the precise distribution of the data is known. The discretization method tends to do better if there is a large amount of training data because it will

learn to fit the distribution of the data. Since naive Bayes is typically used when a large amount of data is available (as more computationally expensive models can generally achieve better accuracy), the discretization method is generally preferred over the distribution method.

## Sample correction

If a given class and feature value never occur together in the training set then the frequency-based probability estimate will be zero. This is problematic since it will wipe out all information in the other probabilities when they are multiplied. It is therefore often desirable to incorporate a small-sample correction in all probability estimates such that no probability is ever set to be exactly zero.

## Constructing a classifier from the probability model

The discussion so far has derived the independent feature model, that is, the naive Bayes probability model. The naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the *maximum a posteriori* or *MAP* decision rule. The corresponding classifier is the function $\text{classify}$ defined as follows:

$$\text{classify}(f_1, \ldots, f_n) = \underset{c}{\operatorname{argmax}}\ p(C = c) \prod_{i=1}^{n} p(F_i = f_i | C = c).$$

## Discussion

Despite the fact that the far-reaching independence assumptions are often inaccurate, the naive Bayes classifier has several properties that make it surprisingly useful in practice. In particular, the decoupling of the class conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution. This in turn helps to alleviate problems stemming from the curse of dimensionality, such as the need for data sets that scale exponentially with the number of features. Like all probabilistic classifiers under the MAP decision rule, it arrives at the correct classification as long as the correct class is more probable than any other class; hence class probabilities do not have to be estimated very well. In other words, the overall classifier is robust enough to ignore serious deficiencies in its underlying naive probability model. Other reasons for the observed success of the naive Bayes classifier are discussed in the literature cited below.

## Examples

### Sex Classification

Problem: classify whether a given person is a male or a female based on the measured features. The features include height, weight, and foot size.

#### Training

Example training set below.

| sex | height (feet) | weight (lbs) | foot size(inches) |
|---|---|---|---|
| male | 6 | 180 | 12 |
| male | 5.92 (5'11") | 190 | 11 |
| male | 5.58 (5'7") | 170 | 12 |
| male | 5.92 (5'11") | 165 | 10 |
| female | 5 | 100 | 6 |
| female | 5.5 (5'6") | 150 | 8 |
| female | 5.42 (5'5") | 130 | 7 |
| female | 5.75 (5'9") | 150 | 9 |

The classifier created from the training set using a Gaussian distribution assumption would be:

| sex | mean (height) | variance (height) | mean (weight) | variance (weight) | mean (foot size) | variance (foot size) |
|---|---|---|---|---|---|---|
| male | 5.855 | 3.5033e-02 | 176.25 | 1.2292e+02 | 11.25 | 9.1667e-01 |
| female | 5.4175 | 9.7225e-02 | 132.5 | 5.5833e+02 | 7.5 | 1.6667e+00 |

Let's say we have equiprobable classes so P(male)= P(female) = 0.5. There was no identified reason for making this assumption so it may have been a bad idea. If we determine P(C) based on frequency in the training set, we happen to get the same answer.

### Testing

Below is a sample to be classified as a male or female.

| sex | height (feet) | weight (lbs) | foot size(inches) |
|---|---|---|---|
| sample | 6 | 130 | 8 |

We wish to determine which posterior is greater, male or female.

posterior (male) = P(male)*P(height|male)*P(weight|male)*P(foot size|male) ~~/ evidence~~

posterior (female) = P(female)*P(height|female)*P(weight|female)*P(foot size|female) ~~/ evidence~~

The evidence (also termed normalizing constant) may be calculated since the sum of the posteriors equals one.

evidence = P(male)*P(height|male)*P(weight|male)*P(foot size|male) + P(female)*P(height|female)*P(weight|female)*P(foot size|female)

The evidence may be ignored since it is a positive constant. (Normal distributions are always positive.) We now determine the sex of the sample.

P(male) = 0.5

P(height|male) = 1.5789 (A probability density greater than 1 is OK. It is the area under the bell curve that is equal to 1.)

P(weight|male) = 5.9881e-06

P(foot size|male) = 1.3112e-3

posterior numerator (male) = their product = 6.1984e-09

P(female) = 0.5

P(height|female) = 2.2346e-1

P(weight|female) = 1.6789e-2

P(foot size|female) = 2.8669e-1

posterior numerator (female) = their product = 5.3778e-04

Since posterior numerator (female) > posterior numerator (male), the sample is female.

## Document Classification

Here is a worked example of naive Bayesian classification to the document classification problem. Consider the problem of classifying documents by their content, for example into spam and non-spam E-mails. Imagine that documents are drawn from a number of classes of documents which can be modelled as sets of words where the (independent) probability that the i-th word of a given document occurs in a document from class *C* can be written as

$$p(w_i|C)$$

(For this treatment, we simplify things further by assuming that words are randomly distributed in the document - that is, words are not dependent on the length of the document, position within the document with relation to other words, or other document-context.)

Then the probability that a given document *D* contains all of the words $w_i$, given a class *C*, is

$$p(D|C) = \prod_i p(w_i|C)$$

The question that we desire to answer is: "what is the probability that a given document *D* belongs to a given class *C*?" In other words, what is $p(C|D)$?

Now by definition

$$p(D|C) = \frac{p(D \cap C)}{p(C)}$$

and

$$p(C|D) = \frac{p(D \cap C)}{p(D)}$$

Bayes' theorem manipulates these into a statement of probability in terms of likelihood.

$$p(C|D) = \frac{p(C)}{p(D)} p(D|C)$$

Assume for the moment that there are only two mutually exclusive classes, *S* and ¬*S* (e.g. spam and not spam), such that every element (email) is in either one or the other;

$$p(D|S) = \prod_i p(w_i|S)$$

and

$$p(D|\neg S) = \prod_i p(w_i|\neg S)$$

Using the Bayesian result above, we can write:

$$p(S|D) = \frac{p(S)}{p(D)} \prod_i p(w_i|S)$$

$$p(\neg S|D) = \frac{p(\neg S)}{p(D)} \prod_i p(w_i|\neg S)$$

Dividing one by the other gives:

$$\frac{p(S|D)}{p(\neg S|D)} = \frac{p(S) \prod_i p(w_i|S)}{p(\neg S) \prod_i p(w_i|\neg S)}$$

Which can be re-factored as:

$$\frac{p(S|D)}{p(\neg S|D)} = \frac{p(S)}{p(\neg S)} \prod_i \frac{p(w_i|S)}{p(w_i|\neg S)}$$

Thus, the probability ratio p(*S* | *D*) / p(¬*S* | *D*) can be expressed in terms of a series of likelihood ratios. The actual probability p(*S* | *D*) can be easily computed from log (p(*S* | *D*) / p(¬*S* | *D*)) based on the observation that p(*S* | *D*) + p(¬*S* | *D*) = 1.

Taking the logarithm of all these ratios, we have:

$$\ln \frac{p(S|D)}{p(\neg S|D)} = \ln \frac{p(S)}{p(\neg S)} + \sum_i \ln \frac{p(w_i|S)}{p(w_i|\neg S)}$$

(This technique of "log-likelihood ratios" is a common technique in statistics. In the case of two mutually exclusive alternatives (such as this example), the conversion of a log-likelihood ratio to a probability takes the form of a sigmoid curve: see logit for details.)

Finally, the document can be classified as follows. It is spam if $p(S|D) > p(\neg S|D)$ (i.e., $\ln \frac{p(S|D)}{p(\neg S|D)} > 0$), otherwise it is not spam.

## References


[1] Harry Zhang "The Optimality of Naive Bayes". FLAIRS2004 conference. *(available online: PDF* (http://www.cs.unb.ca/profs/hzhang/publications/FLAIRS04ZhangH.pdf)*)*

[2] Caruana, R. and Niculescu-Mizil, A.: "An empirical comparison of supervised learning algorithms". Proceedings of the 23rd international conference on Machine learning, 2006. *(available online PDF* (http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.122.5901&rep=rep1&type=pdf)*)*

[3] George H. John and Pat Langley (1995). Estimating Continuous Distributions in Bayesian Classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. pp. 338-345. Morgan Kaufmann, San Mateo.


## External links

- Benchmark results of Naive Bayes implementations (http://tunedit.org/results?d=UCI/&a=bayes)
- Hierarchical Naive Bayes Classifiers for uncertain data (http://www.biomedcentral.com/1471-2105/7/514) (an extension of the Naive Bayes classifier).
- winnowTag (http://winnowtag.org) Create tags that run Bayesian classifiers on over 1/2 million items updated from 7,500 feeds.
- Online application of a Naive Bayes classifier (http://www.convo.co.uk/x02/) (emotion modelling), with a full explanation.
- dANN: Naive Classifier - Explanation and Example (http://wiki.syncleus.com/index.php/DANN:Naive_Classifier)
- BayesNews - Bayesian RSS Reader (http://www.bayesnews.xpg.com.br/) (useful for personal news clipping).
- Naive Bayes in PMML (http://www.dmg.org/v4-0/NaiveBayes.html)
- Simple example of document classification with Naive Bayes, implemented in Ruby (http://www.nils-haldenwang.de/computer-science/machine-learning/how-to-apply-naive-bayes-classifiers-to-document-classification-problems)

Software

- IMSL Numerical Libraries Collections of math and statistical algorithms available in C/C++, Fortran, Java and C#/.NET. Data mining routines in the IMSL Libraries include a Naive Bayes classifier.
- Orange, a free data mining software suite, module orngBayes (http://www.ailab.si/orange/doc/modules/orngBayes.htm)
- Winnow content recommendation (http://doc.winnowtag.org/open-source) Open source Naive Bayes text classifier works with very small training and unbalanced training sets. High performance, C, any Unix.