Naïve Bayes Algorithm in CLOP

Isabelle Guyon – September 2005 – Isabelle@clopinet.com

We implemented two versions of Naïve Bayesian classifiers, one for binary inputs and one for continuous inputs. Both make independence assumptions between the input variables/features. The binary version uses frequency counts to estimate probabilities. The continuous version assumes a Gaussian distribution of the samples in each class.

Naïve Bayes for binary inputs

We treat here the two-class case with binary 0/1 inputs and ±1 target values.

Bayes classifiers follow the rule: classify pattern **x** in class 1 if

$$P(\text{target}=1 \mid \mathbf{x}) > P(\text{target}=-1 \mid \mathbf{x}) \quad (1)$$

and in the other class otherwise.

According to Bayes' rule

$P(\text{target}=t \mid \mathbf{x}) = P(\mathbf{x} \mid \text{target}=t)\, P(\text{target}=t) / P(\mathbf{x})$

with $t=\pm1$.

Because $P(\mathbf{x})$ does not affect the result, (1) is also equivalent to classifying pattern **x** in class 1 if

$$P(\mathbf{x} \mid \text{target}=1)\, P(\text{target}=1) > P(\mathbf{x} \mid \text{target}=-1)\, P(\text{target}=-1) \quad (2)$$

and in the other class otherwise.

The independence assumptions allow us to write:

$$P(\mathbf{x} \mid \text{target}=t) = \prod_i P(x_i \mid \text{target}=t) \quad (3)$$

Each factor can be estimated from the training data as frequency counts:

$P(x_i =1 \mid \text{target}=1) \cong f_{11i}$= fraction of times feature i is 1 in training ex. from class 1.

$P(x_i =0 \mid \text{target}=1) \cong f_{01i}$=fraction of times feature i is 0 in training ex. from class 1.

$P(x_i =1 \mid \text{target}=-1) \cong f_{12i}$=fraction of times feature i is 1 in training ex. from class 2.

$P(x_i =0 \mid \text{target}=-1) \cong f_{02i}$=fraction of times feature i is 0 in training ex. from class 2.

By taking the log of (2) and using (3) we can create a linear discriminant function:

Classify pattern **x** in class 1 if

$F(\mathbf{x}) > 0$

and in the other class otherwise.

$$F(\mathbf{x})= \sum_i \log [P(x_i \mid \text{target}=1) / P(x_i \mid \text{target}=-1)] + b \quad (4)$$

with $b= \log P(\text{target}=1) - \log P(\text{target}=-1)$.

$\log P(\text{target}=1) \cong f_1$ =fraction of positive examples in the training data.

$\log P(\text{target}=-1) \cong f_2$ =fraction of negative examples in the training data..

We need to switch values depending on whether the actual feature observed is 0 or 1, therefore (4) becomes:

$F(\mathbf{x}) = \sum_i \big(x_i \log [P(x_i = 1|target=1) / P(x_i = 1|target=-1)] +$

$\qquad (1 - x_i) \log [P(x_i = 0|target=1) / P(x_i = 0|\, target=-1)] \big) + b$

or simply:

$F(\mathbf{x}) = \sum_i \big(x_i \log (f_{11i}/f_{12i}) + (1 - x_i) \log (f_{01i}/f_{02i}) \big) + \log(f_1 / f_2)$

Thus:

$$\boxed{\begin{array}{l} F(\mathbf{x}) = \sum_i w_i\, x_i + B \\ \text{where} \\ w_i = \log (f_{11i}/f_{12i}) - \log (f_{01i}/f_{02i}) \\ B = \sum_i \log (f_{01i}/f_{02i}) + \log(f_1 / f_2). \end{array}}$$

Notes:

- By playing on the class priors P(target=1) and P(target=-1), one varies the tradeoff precision recall by changing the bias.
- The method can also be used for feature ranking (using the absolute values of w_i as ranking criterion.
- The method can be trivially extended to the multi-class case and the categorical variable case. For the continuous case, one can consider extending it with T, Hastie's trick.
- The frequency estimations can make use of a prior. If $f_i = n_i / n$, we replace the frequency f_i by $f_i' = (n\, f_i + mean(f_i)) / (n + 1)$. Therefore, even if we have very few observations of positive features, we never get $f_i = 0$.

Gaussian classifier

We implemented a Naïve Bayes classifier for continuous that makes the assumption that the class conditional probabilities are Gaussian distributed. With the feature independence assumption, one gets the density:

$$P(\mathbf{x}|class1) = C \prod_i \exp{-0.5}\, (x_i - \mu_{1i})^2/\sigma_i^2$$

where C is a constant that is the same for all classes, μ_{1i} is the mean value of feature i for the examples of class 1, and is the "pooled" within class standard deviation of feature i (essentially the stdev of examples of class 1 averaged with the stdev of examples of class 2). We have a similar expression for class 2.

A good discriminant function F(**x**) should be positive if **x** is more likely to belong to class 1 and negative otherwise, that is if:

$P(class1 \mid \mathbf{x}) > P(class2 \mid \mathbf{x})$

or, after applying Bayes rule:

$P(\mathbf{x} \mid class1)\, P(class1) / P(\mathbf{x}) > P(\mathbf{x} \mid class2)\, P(class2) / P(\mathbf{x})$

or equivalently:

$\log P(\mathbf{x} \mid class1) - \log P(\mathbf{x} \mid class2) + \log P(class1) - \log P(class2) > 0$

This leads us to choose the following discriminant function:

$F(\mathbf{x}) = \log P(\mathbf{x} \mid class1) - \log P(\mathbf{x} \mid class2) + \log P(class1) - \log P(class2)$

Using (1), we obtain:

$F(\mathbf{x}) = -0.5 \sum_i (x_i - \mu_{1i})^2/\sigma_i^2 + 0.5 \sum_i (x_i - \mu_{2i})^2/\sigma_i^2 + \log P(class1)/P(class2)$

This can be rewritten as a linear discriminant function:

$F(\mathbf{x}) = \sum_i w_i x_i + b$

with

$$w_i = 0.5 (\mu_{1i} - \mu_{2i})/\sigma_i^2 \qquad (1)$$

$$b = \sum_i 0.5 (\mu_{2i}^2 - \mu_{1i}^2) / \sigma_i^2 + \log P(class1)/P(class2)$$

Mean and standard deviation are estimated in a standard way using training data. P(class1) and P(class2) are the class priors that can be estimated by frequency counts n_1/n and n_2/n, where n_1 and n_2 are the number of examples in class 1 and 2 and n is the total number of examples.

For feature selection, the ranking is done with the absolute value of the weights.

Reference:

Duda, R. O. & Hart, P. E. (1973). Pattern classification and scene analysis. Wiley. p.26.