Handmade Naive Bayes

Álvaro Orgaz Expósito

THEORY

The aim of the *Naive Bayes* is to predict the categorical variable y with K possible categories (then it is a *classification* problem) knowing the explanatory variables $x = \{x_1, \ldots, x_p\}$.

Firstly, you need to understand the statistical concept of the *Bayes theorem* used in the *Naive Bayes* model to compute the probability $P(y = k|x)$ with $1 \leq k \leq K$. This is the formula of the theorem

$$P(y|x) = \frac{P(y \cap x)}{P(x)}$$

where $P(y|x)$ is the conditional probability of y knowing x, $P(x)$ is the probability of x, and $P(y, x)$ is the probability of y and x.

Secondly, you need to understand the difference between *Bayesian* and *frequentist* statistical modelling. Basically, *frequentist* statistics assume a probability distribution of y with concrete parameters for predicting. For example, if y has a normal distribution with mean 5 and deviation 2 we would get

$$P(y|\mu, \sigma) \sim Normal(\mu = 5, \sigma = 2)$$

But *Bayesian* statistics assume a probability distribution of y and a distribution for each parameter instead of a concrete value. Then, it computes the final probability distribution of y using the explained *Bayes theorem* as follows

$$P(y) = \int_{\Omega} P(y|\mu, \sigma)P(\mu, \sigma) \ \ d\mu \ \ d\sigma$$

where $P(\mu, \sigma)$ is the assumed distribution for parameters (known as *prior*). For example, using the same case we could assume

$$P(y|\mu, \sigma) \sim Normal \quad with \quad P(\mu, \sigma) \sim BivariateNormal(\mu_\mu = 5, \mu_\sigma = 2, covariance_{\mu,\sigma})$$

Note: It easy to understand the *Bayesian* statistical modelling as a weighted prediction of all candidates distributions of y using a distribution of the parameters as weights. This concept is important to know what *Bayesian* statistics provide, but do not confuse it with the *Naive Bayes* algorithm which uses the *Bayes* theorem in a differet way.

Thirdly, let's move to the classification scenario where the aim is to classify a set of points x as belonging to one of K classes. For doing that, the *Naive Bayes* computes the conditional probability $P(y = k|x)$ for each of the classes using the *Bayes theorem* and chooses the class with the highest probability as the prediction.

A straightforward application of *Bayes theorem* gives the formula of the *Naive Bayes* classifier

$$P(y = k|x) = \frac{P(y = k \cap x)}{P(x)} = \frac{P(y = k \cap x)}{\sum_{c=1}^{K} P(y = c \cap x)} = \frac{P(x|y = k)P(y = k)}{\sum_{c=1}^{K} P(x|y = c)P(y = c)}$$

where $P(y = k|x)$ is the conditional probability of y equal to class k knowing x, $P(y = k)$ is the *prior* probability of y equal to k, and $P(x|y = k)$ is the conditional probability of x knowing that y is equal to k.

But as you can see, the denominator does not depend on the class k and the model only needs to use the simplified classifier $\delta_k(x) \propto P(x|y = k)P(y = k)$ to select the class with the highest probability as the prediction.

Note: The reason for *Naive* as the name of the model is that the algorithm assumes that all p features are conditionally independent of every other feature. It simplifies a lot the definition of the conditional distribution

$$P(x|y=k) = \prod_{j=1}^{p} P(x_j|y=k)$$

Fourthly, you need to understand that the *Naive Bayers* algorithm varies depending on the distributions $P(y=k)$ and $P(x|y=k)$ that the user chooses. In this paper, we will code the *Gaussian Naive Bayes*:

- $P(y=k)$ as the percentage of observations with class $y=k$ in the data.
- $P(x|y=k)$ as a p-multivariate normal distribution for each class

  $$P(x|y=k) = \frac{exp\big(-\frac{1}{2}(x-\mu_k)\Sigma_k^{-1}(x-\mu_k)^T\big)}{\sqrt{(2\pi)^p|\Sigma_k|}}$$

  where

  $$\mu_k = \left[\frac{\sum_{\{i|y_i=k\}} x_{i1}}{N_k} \quad \dots \quad \frac{\sum_{\{i|y_i=k\}} x_{ip}}{N_k}\right]$$

  and with the *naive* conditional independent assumption the covariance between features is 0, then

  $$\Sigma_k = \begin{bmatrix} \sigma^2_{x_1\{i|y_i=k\}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma^2_{x_p\{i|y_i=k\}} \end{bmatrix}$$

Then in this paper, the simplified classifier will be

$$\delta_k(x) \propto log\big(P(x|y=k)P(y=k)\big) = log\big(P(x|y=k)\big) + log\big(P(y=k)\big) =$$

$$-\frac{1}{2}(x-\mu_k)\Sigma_k^{-1}(x-\mu_k)^T - log\big(\sqrt{|\Sigma_k|}\big) - log\big(\sqrt{(2\pi)^p}\big) + log\big(P(y=k)\big)$$

Finally, let's see the generalized steps of this *machine learning* algorithm *Naive Bayes.*

1. Define the *prior* distribution $P(y=k)$ for each K classes.
2. Define the conditional distribution $P(x|y=k)$ for each K classes.
3. For every observation in the target data x^{target}, compute the classifier $\delta_k(x) \propto P(x|y=k)P(y=k)$ as explained for all K classes.
4. For every observation in the target data x^{target}, select the class k with the highest value as the prediction.

CODE

The data used is the popular dataset *iris.* Let's predict the categorical variable *Species* (then it is a *classification* problem with 3 categories) with the explanatory features: *Petal.Length* and *Petal.Width.*

Let's define the inputs x and y. Also, let's define x^{target} by creating artificial data with all combinations of both features from the minimum to the maximum values in x by 0.05, it will be interesting for observing the *Naive Bayes decision boundaries* in the following plots.

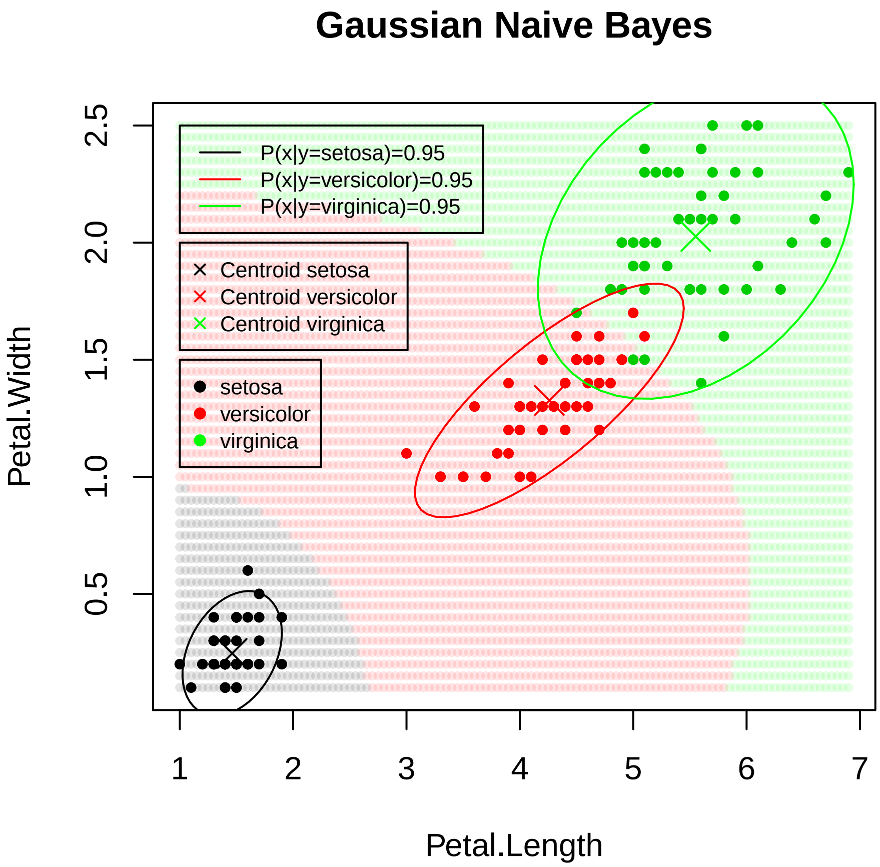
```
x <- iris[,c("Petal.Length","Petal.Width")]
y <- iris[,"Species"]
x_target <- expand.grid(list(Petal.Length=seq(min(x[,1]),max(x[,1]),0.05),
                             Petal.Width=seq(min(x[,2]),max(x[,2]),0.05)))
```

Let's create the *Naive Bayes* algorithm.

```
NaiveBayes <- function(x,y,x_target){
  # Define the prior P(y=k) and the conditional P(x|y=k) distributions
  classes <- unique(y)
  K <- length(classes)
  p <- ncol(x)
  prior <- rep(0,times=K)
  mu <- matrix(0,nrow=K,ncol=p)
  covariance <- array(0,dim=c(p,p,K))
  for(k in 1:K){
    # Compute prior of class k
    prior[k] <- mean(y==classes[k])
    for(j in 1:p){
      # Compute mu of class k and feature j
      mu[k,j] <- mean(x[y==classes[k],j])
      # Compute variance of class k and feature j
      covariance[j,j,k] <- var(x[y==classes[k],j])
    }
  }
  # Define the simplified classifier function
  classifier <- function(x,k){
    log_prior <- log(prior[k])
    log_conditional <- -1/2*sum((x-mu[k,])^2/diag(covariance[,,k]))
                       -log(sqrt(det(covariance[,,k])))-log(sqrt(2*pi)^p)
    return(log_conditional+log_prior)
  }
  # Iterate all target data, compute the classifier for each class and make prediction
  predictions <- rep(0,times=nrow(x_target))
  classifier_values <- matrix(0,nrow=nrow(x_target),ncol=K)
  for(i in 1:nrow(x_target)){
    for(k in 1:K){
      classifier_values[i,k] <- classifier(x_target[i,],k)
    }
    predictions[i] <- which.max(classifier_values[i,])
  }
  return(predictions)
}
```

Now, let's apply the *Naive Bayes* and plot the result.

```
predictions <- NaiveBayes(x,y,x_target)
transparent_colors <- scales::alpha(c("black","red","green"),0.1)
plot(x_target[,1],x_target[,2],col=transparent_colors[as.numeric(predictions)],
     pch=19,cex=0.5,xlab="Petal.Length",ylab="Petal.Width",main="Gaussian Naive Bayes")
points(x[,1],x[,2],col=y,pch=19,cex=0.6)
legend(1,1.5,legend=unique(y),col=c("black","red","green"),pch=19,cex=0.7)
# Adding the 0.95 level curves of the conditional P(x|y=k) distributions for each class
library(car)
dataEllipse(x[,1],x[,2],group=y,group.labels=NA,add=T,levels=0.95,plot.points=F,
            col=c("black","red","green"),center.cex=2,center.pch=4,lwd=1)
legend(1,2,legend=paste("Centroid",unique(y)),col=c("black","red","green"),pch=4,cex=0.7)
legend(1,2.5,legend=paste0("P(x|y=",unique(y),")=0.95"),col=c("black","red","green"),
       lwd=1,cex=0.7)
```



In conclusion, in these plots we can observe the predictions for x^{target} (transparent coloured points) and the real values y (solid coloured points). We can observe the *decision boundary* of the *Gaussian Naive Bayes* for each class in the target variable *Species* in the *iris* data. Also, we can observe the curve level of 0.95 probability for each conditional distribution $P(x|y = k)$ as well as the centroid of these distributions μ_k, remember that in this paper we assume that $P(x|y = k)$ is a p-multivariate normal distribution.