Bayesian classifier

Yangyi Lu

1/17/2020

Simulation example: Mixture Gaussians

Step 1: Generate means for "Blue" class

Generate $\mu_1, \ldots, \mu_{10}$ from 2d Gaussian distribution $N((1,0)^T, \mathbf{I})$.

Step 2: Generate means for "Orange" class

Generate $\mu_{11}, \ldots, \mu_{20}$ from 2d Gaussian distribution $N((0,1)^T, \mathbf{I})$.

Step 3: Generate observations in each class

Pick μ_l at random with probability 0.1, then generate a sample from $N(\mu_l, \mathbf{I}/5)$, which leads to a mixture of Gaussians:

$$p(x|Y = ”blue”) = \sum_{l=1}^{10} 0.1\phi(\mu_l, \mathbf{I}/5) \tag{1}$$

$$p(x|Y = ”orange”) = \sum_{l=11}^{20} 0.1\phi(\mu_l, \mathbf{I}/5), \tag{2}$$

where $\phi(\mu, \Sigma)$ denotes the bivariate Gaussian density function.

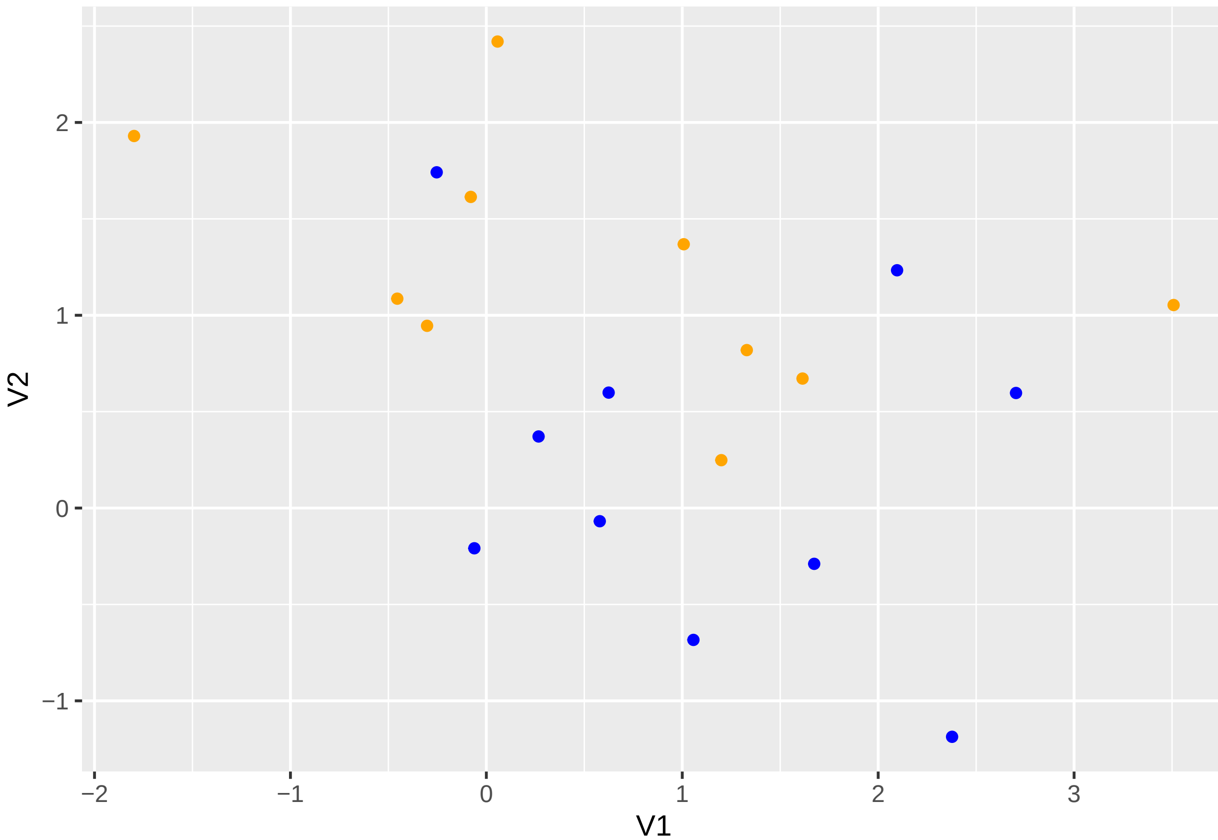
μ_l's and 6831 test points are provided in mixture.Rdata, data info is provided in misture-info.txt.

Questions:

(a) Plot the 20 μ_l's using the corresponding class color.

First, we load μ_l's from mixture.Rdata and use **geom_point** from **ggplot2** to plot them using the corresponding class color.

```r
load("mixture.Rdata")
library(ggplot2)
# means: 20 by 2 table, ordered by \mu_1,...,\mu_20.
means = as.data.frame(means)
color = c(rep("blue",10),rep("orange",10))
ggplot(means,aes(x=V1,y=V2))+geom_point(colour=color)
```



(b) For each grid point, compute $p(x|Y = "blue")$, $p(x|Y = "orange")$ and $p(x)$

$P(x|Y = "blue")$ and $P(x|Y = "orange")$ are known in the question and can be calculated using function **dmvnorm** and **mvtnorm** library. $P(Y = "blue") = P(Y = "orange") = 0.5$.

The marginal density can be written as:

$$p(x) = \sum_y p(x, Y = y) = p(x|Y = "blue")P(Y = "blue") + p(x|Y = "orange")P(Y = "orange")$$

.

```
library(mvtnorm)
# mixture Gaussian for blue class
density_blue = function(x){
den = 0
for(i in 1:10){
den = den + dmvnorm(x,as.numeric(means[i,]),0.2*diag(2))
}
return(den/10)
}
# mixture Gaussian for blue class
density_orange = function(x){
den = 0
for(i in 11:20){
den = den + dmvnorm(x,as.numeric(means[i,]),0.2*diag(2))
}
```

```
return(den/10)
}
x_marginal_blue = apply(xnew,1,density_blue)
x_marginal_orange = apply(xnew,1,density_orange)
x_marginal = 0.5*x_marginal_blue+0.5*x_marginal_orange
```

(c) Based on (b), for each grid point, compute $P(Y = blue|x)$ and decide a color for each grid point

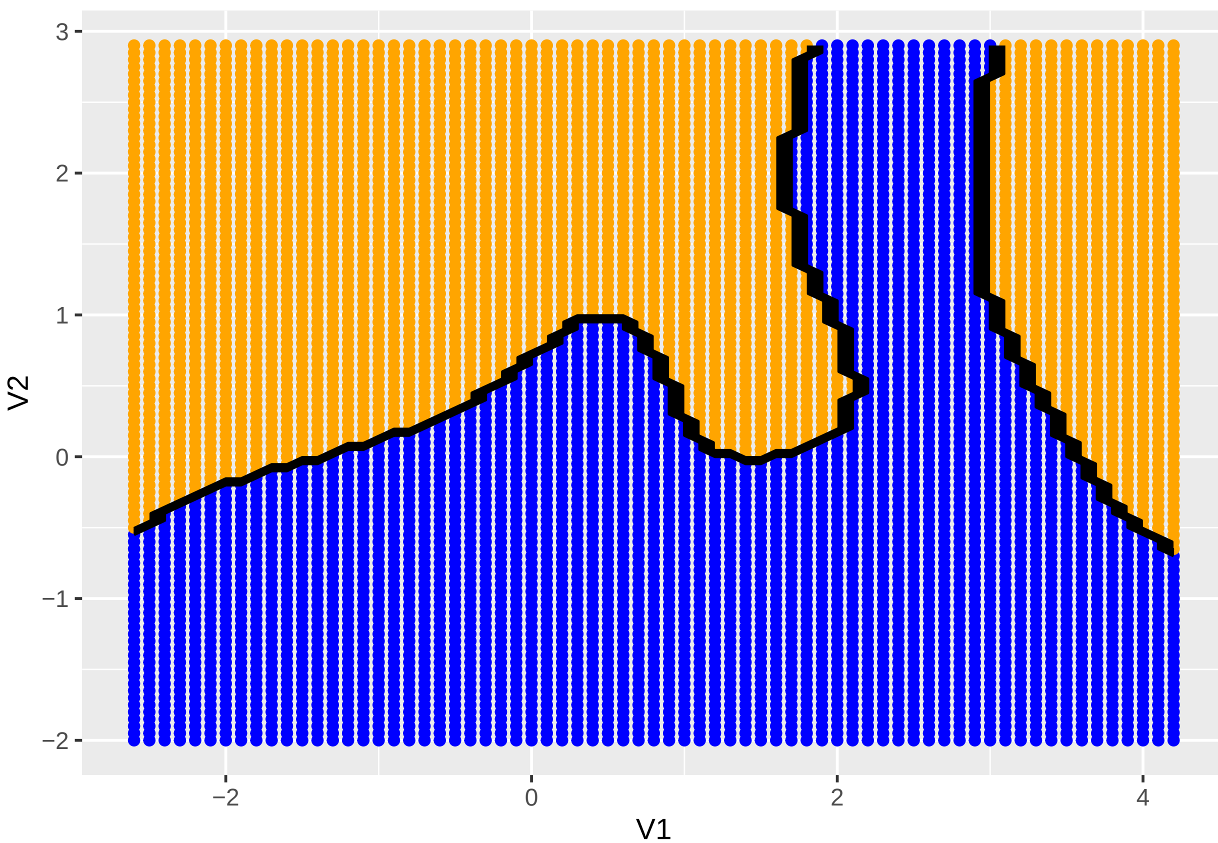
$$p(Y = "blue"|x) = p(x|Y = "blue")P(Y = "blue")/p(x)$$

. The color of point x should be blue if $P(Y = "blue"|x) > 0.5$ otherwise red according to Bayes optimal classifier.

```
conditional_blue = x_marginal_blue*0.5/x_marginal
color = rep("orange",nrow(xnew))
color[conditional_blue>0.5] = "blue"
```

(d) Plot the test points using the so decided color in (c), and also plot the decision boundary

```
xnew = data.frame(xnew)
ggplot(xnew,aes(x=V1,y=V2,z = as.numeric(conditional_blue>0.5)))+
geom_point(color=color)+geom_contour(color='black')
```



(e) Compute the Bayes error rate

The Bayes error can be calculated as follows:

$$P(Y \neq C(X)) = \int_x P(Y \neq C(x))p(x)dx \approx \frac{\sum_x P(Y \neq C(x))p(x)}{\sum_x p(x)}$$

. For every point x,

$$\begin{aligned} P(Y \neq C(x)) &= P(Y = Blue, C(x) = Orange) + P(Y = Orange, C(x) = Blue) && (3) \\ &= P(Y = "Blue"|x)\mathbf{1}_{\{P(Y="Blue"|x)<0.5\}} + (1 - P(Y = "Blue"|x))\mathbf{1}_{\{P(Y="Blue"|x)>0.5\}} && (4) \end{aligned}$$

```
bayes_error=sum(x_marginal*(conditional_blue*as.numeric(conditional_blue<0.5)+
(1-conditional_blue)*as.numeric(conditional_blue>0.5)))/sum(x_marginal)
bayes_error
```

```
## [1] 0.2101192
```