K-nearest-neighbours

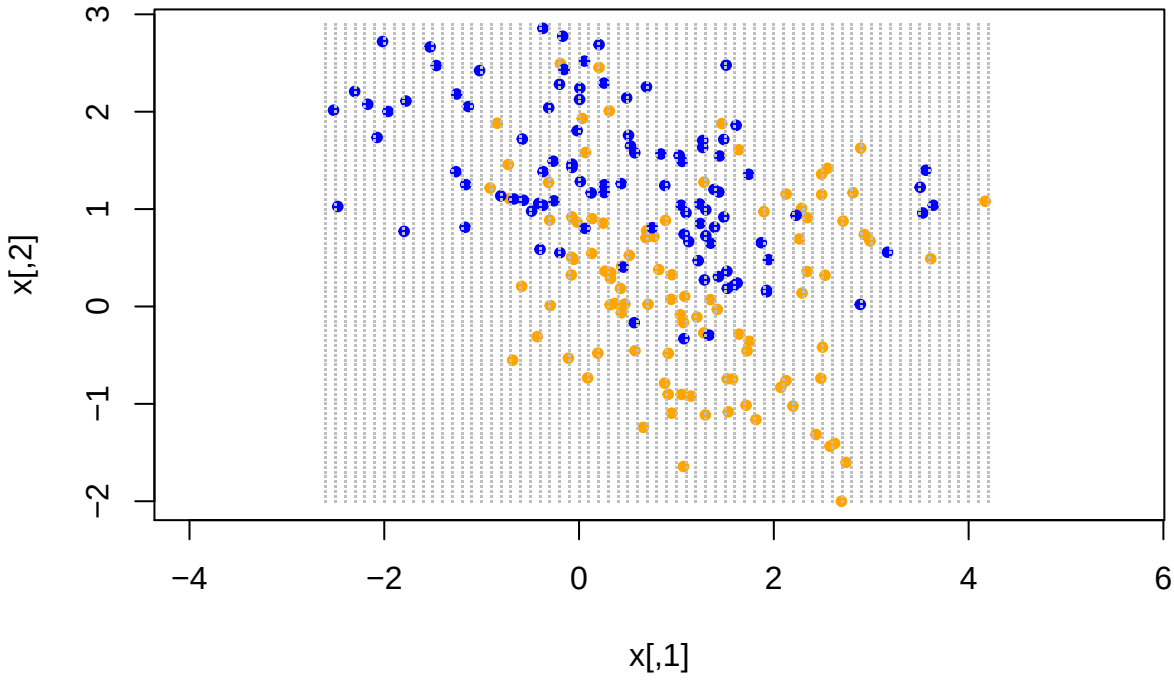
Let's have a look at a non-parametric tool such as Knn for classification.

```
#Downloading the data
website <- "https://web.stanford.edu/~hastie/ElemStatLearn/datasets/ESL.mixture.rda"
load(file(website))
summary(ESL.mixture) # Gaussian mixture data
```

```
##          Length Class  Mode
## x          400  -none- numeric
## y          200  -none- numeric
## xnew     13662  matrix numeric
## prob      6831  -none- numeric
## marginal  6831  -none- numeric
## px1         69  -none- numeric
## px2         99  -none- numeric
## means       40  -none- numeric
```

```
attach(ESL.mixture)
```

```
plot(x,pch=20, col=ifelse(y==0, "orange", "blue"), asp = 1)
# px1 and px2 define the limits of the input space
grid <- expand.grid(ESL.mixture$px1, ESL.mixture$px2)
points(grid, col="gray", pch=".")
```



Even though we know that the data comes from a Gaussian mixture, we want to use knn to classify the points.

```
#install.packages("FNN")
library(FNN)
```

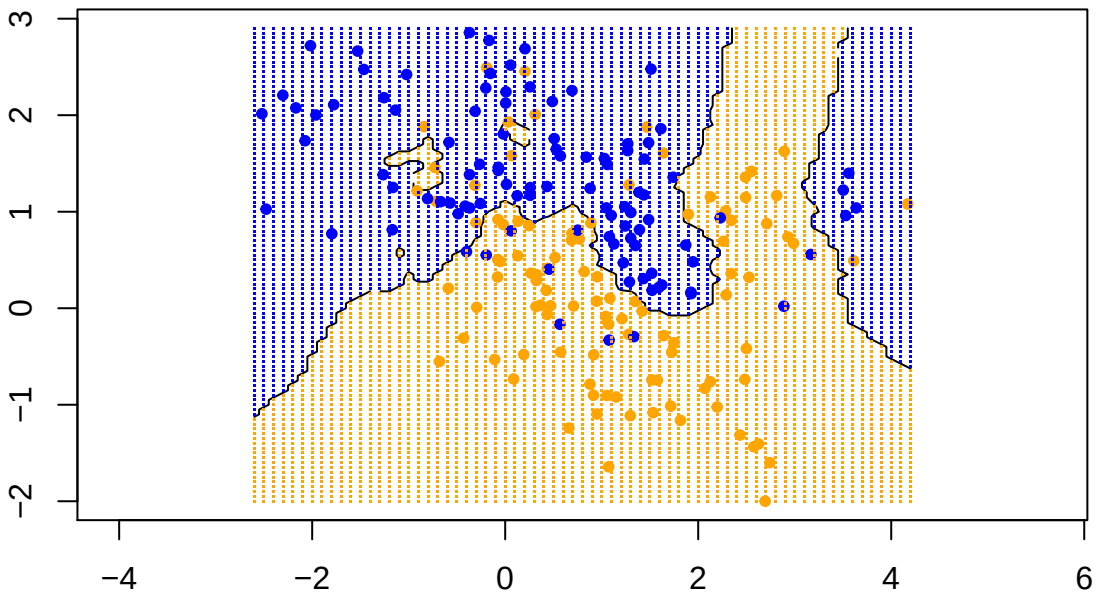
```
## Warning: package 'FNN' was built under R version 3.6.3
```

```
model <- knn(train=x,test=grid,cl=y,k=5, prob = T)
```

Let's look at the predictions of the model for the test set.

```
p <- ifelse(model=="1", attr(model, "prob"), 1- attr(model, "prob"))
```

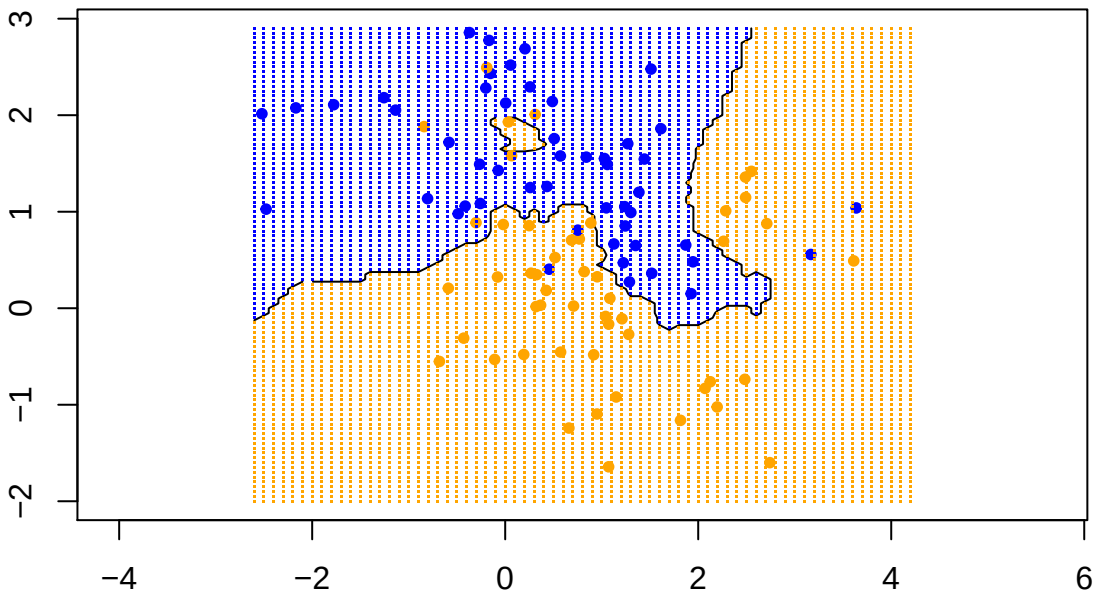
```
#let's also plot the decision boundary
prob.matrix <- matrix(p, length(px1), length(px2))
contour(px1,px2,prob.matrix, levels=0.5, labels="", xlab="", ylab="", asp=1)
# the training points
points(x,pch=20, col=ifelse(y==0, "orange", "blue"))
# and the resto of the space
grid <- expand.grid(px1, px2)
points(grid, col=ifelse(model==0, "orange","blue"), pch=".")
```

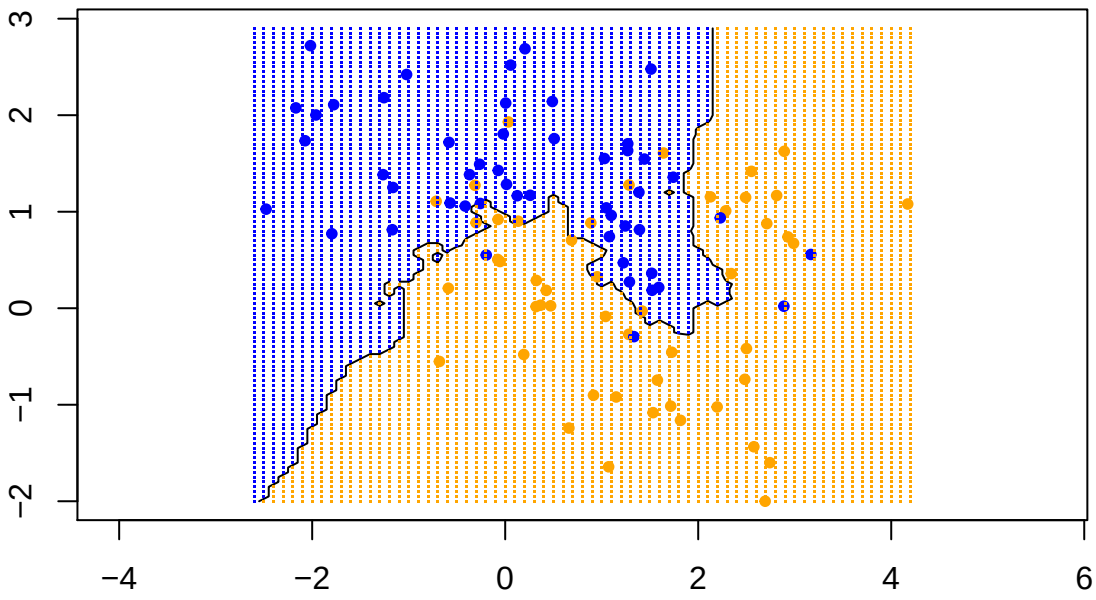


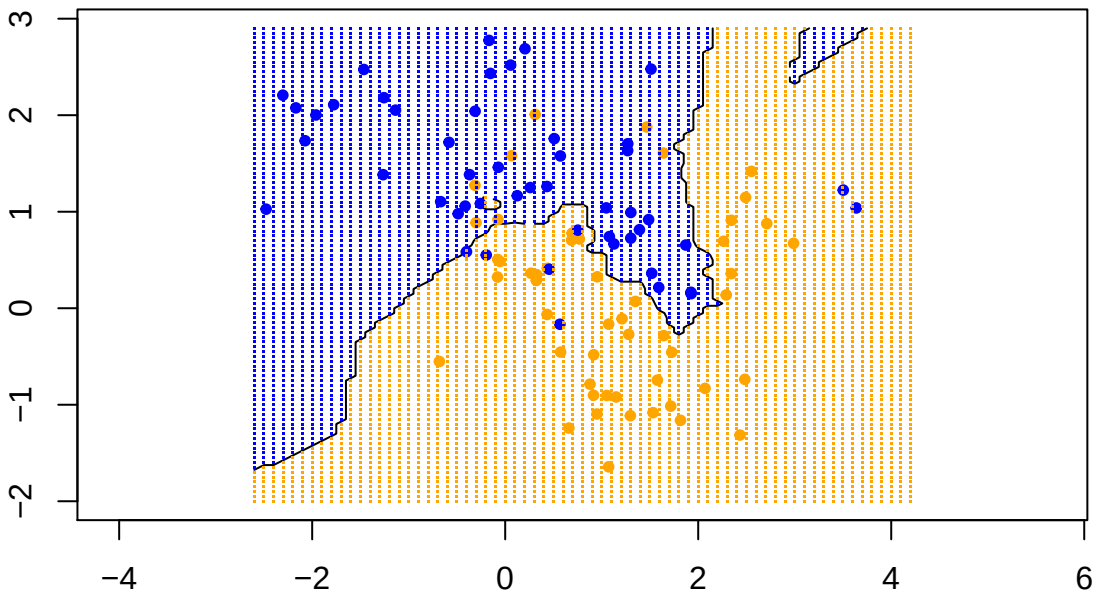
How robust is this classifier? Let's use random sampling to answer that.

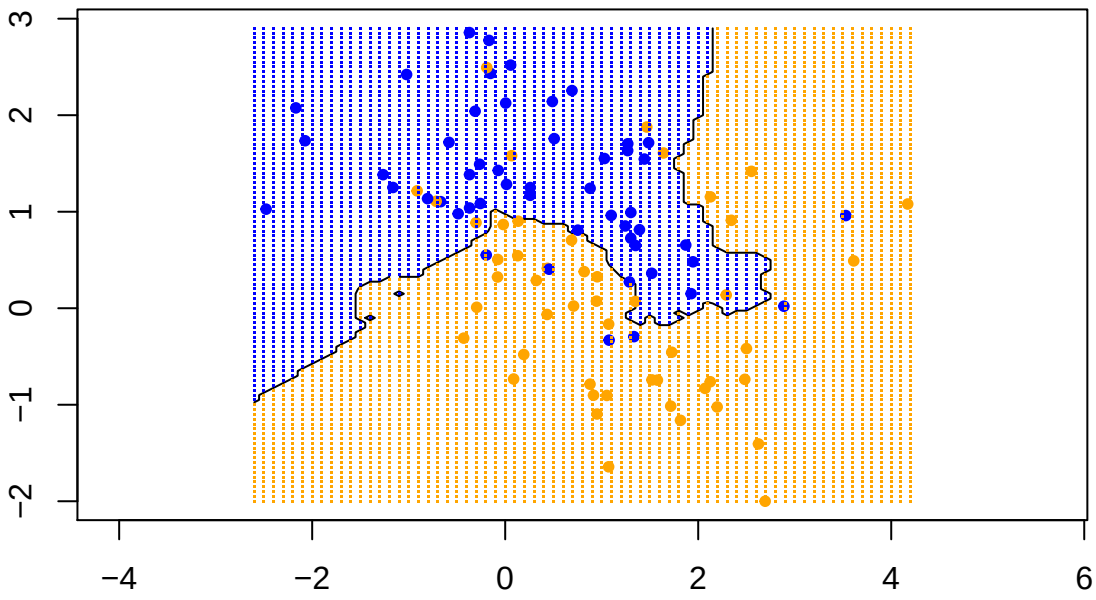
```
sampling.knn <-function(k){
  m <- 100 #sample size
  n <- length(y)
  nrep <- 4
  for(i in 1:nrep){
    indices <- sample(1:n,size = m, replace = F)
    y.sample <- y[indices]
    x.sample <- x[indices,]
    fit.sample <- knn(train=x.sample,test=grid,cl=y.sample,k=k, prob = T)
    # plotting
    p <- ifelse(fit.sample=="1", attr(fit.sample, "prob"), 1- attr(fit.sample,
                                                                   "prob"))

    prob.matrix <- matrix(p, length(px1), length(px2))
    contour(px1,px2,prob.matrix, levels=0.5, labels="", xlab="", ylab="", asp=1)
    points(x.sample,pch=20, col=ifelse(y.sample==0, "orange", "blue"))
    grid <- expand.grid(px1, px2)
    points(grid, col=ifelse(p<0.5, "orange","blue"), pch=".")
}
}
```

```
sampling.knn(5)
```


3
2
1
0
−1
−2
−4
−2
0
2
4
6

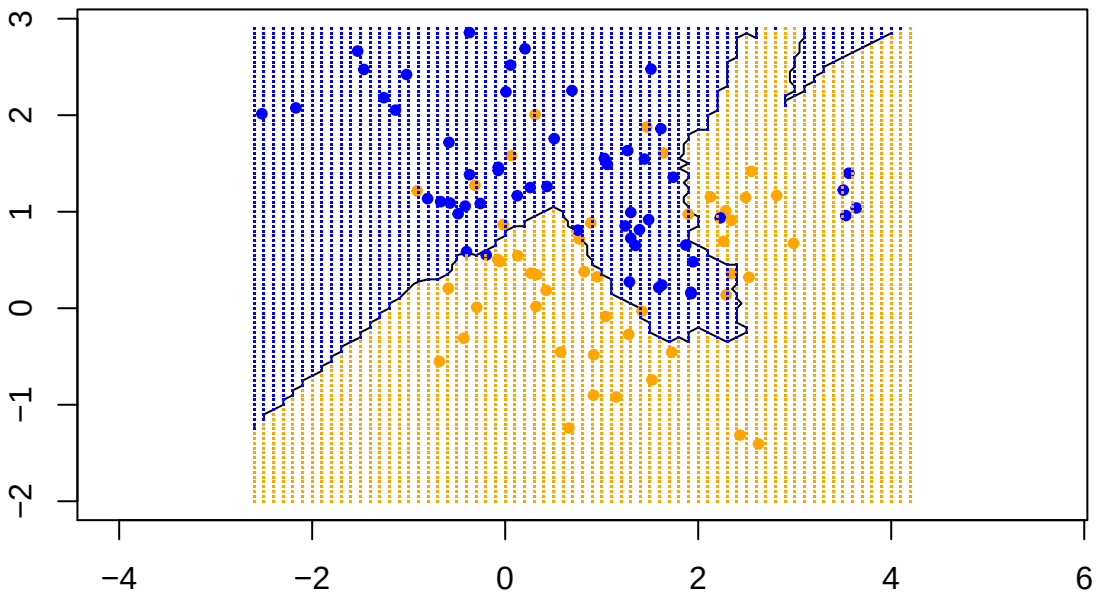

3
2
1
0
−1
−2
−4
−2
0
2
4
6

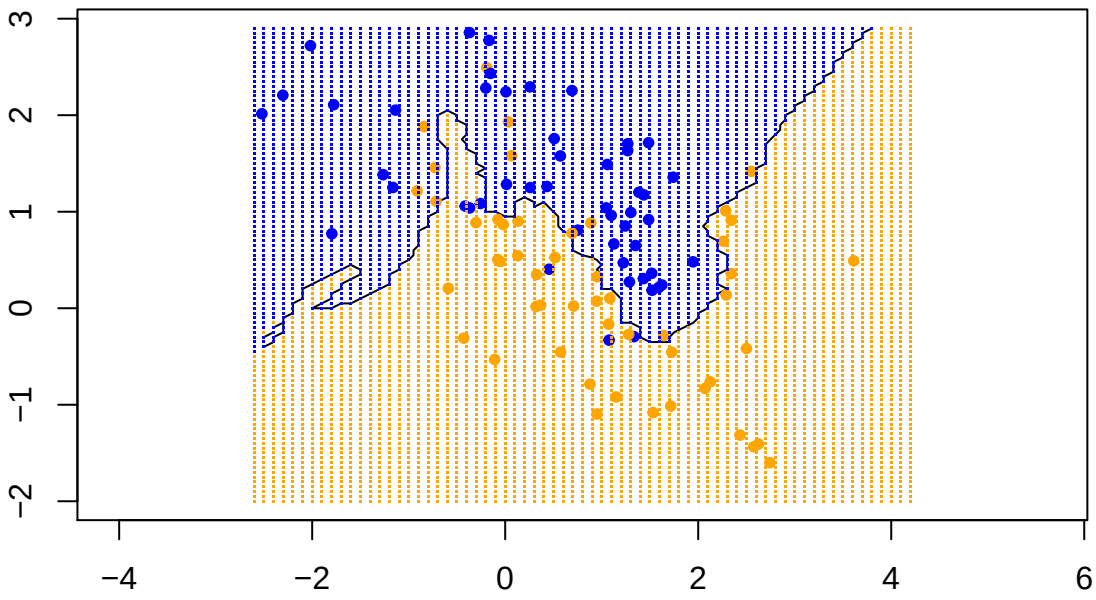

3
2
1
0
−1
−2
−4
−2
0
2
4
6

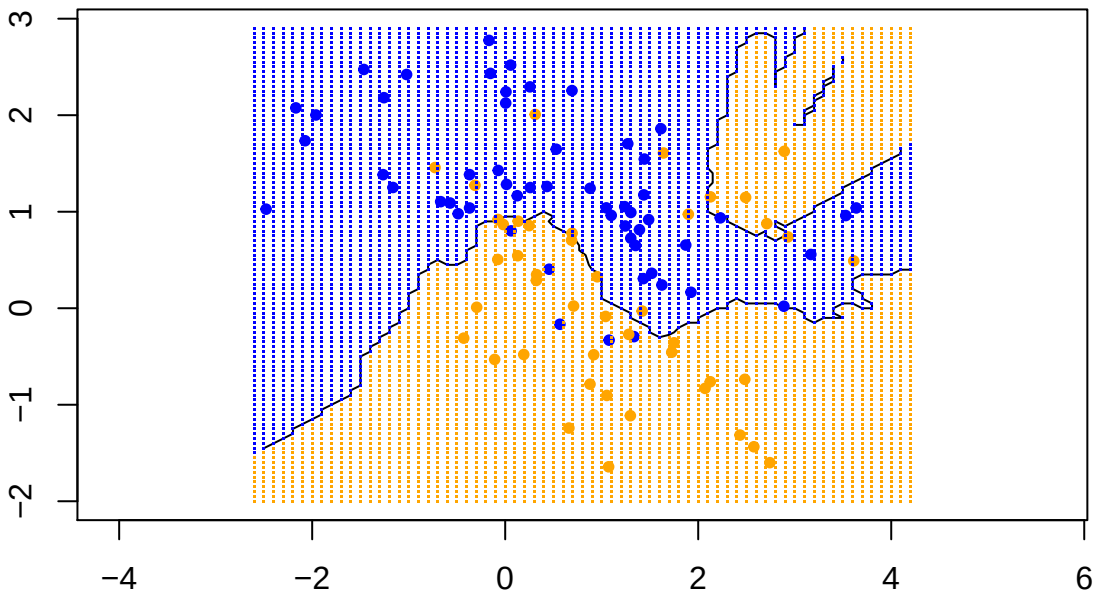


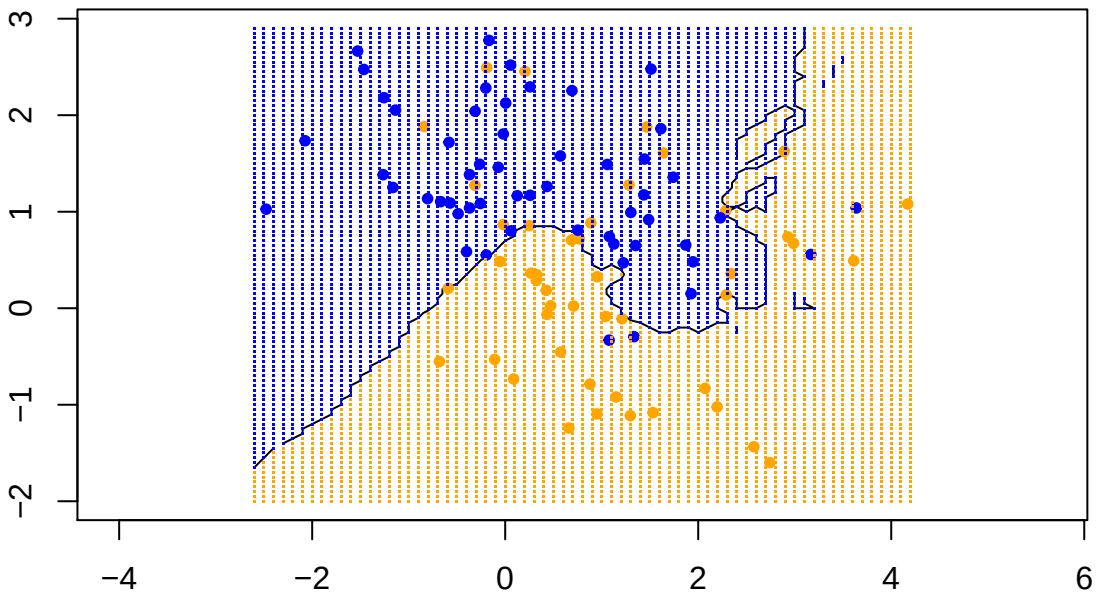
From the above plots the decision boundary seems to be highly unstable. Let's try changing k to see its effect on the robustness of the classifier.

```
sampling.knn(k=10)
```

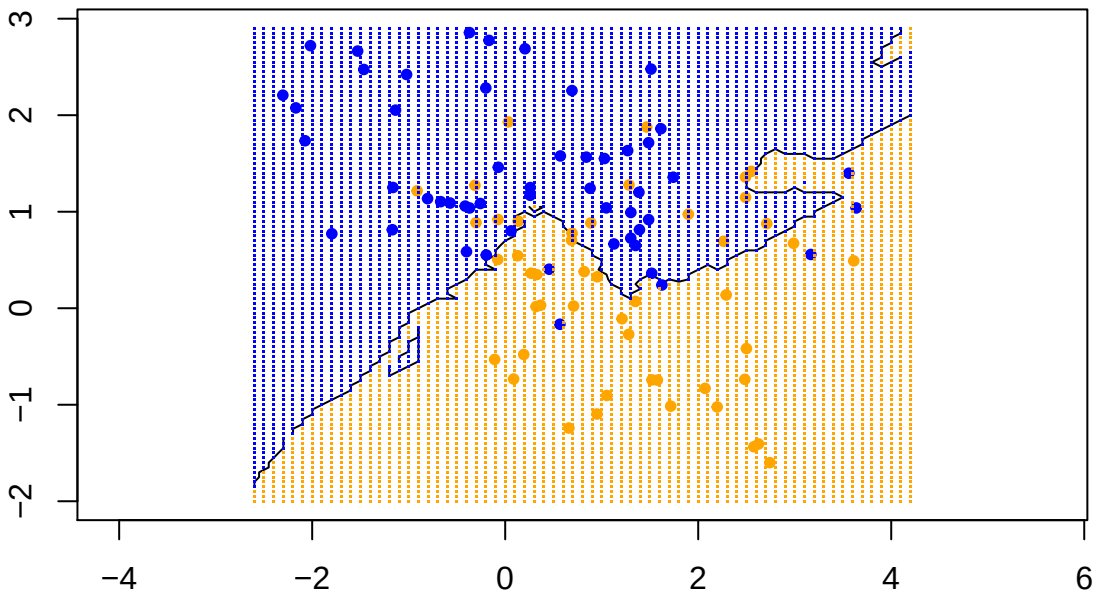

3
2
1
0
−1
−2
−4
−2
0
2
4
6

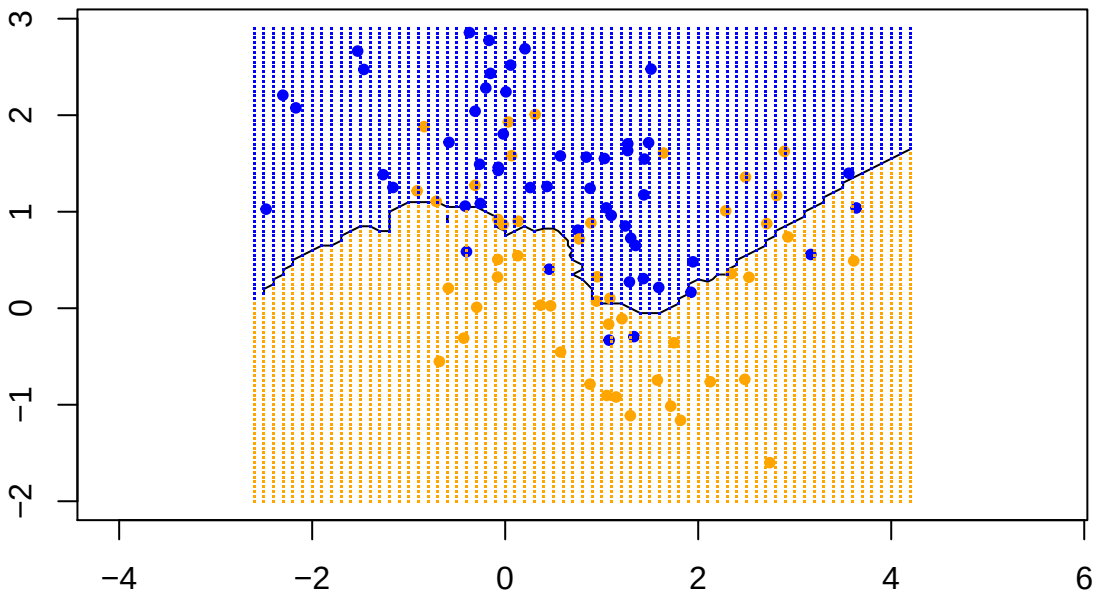

3
2
1
0
−1
−2
−4
−2
0
2
4
6

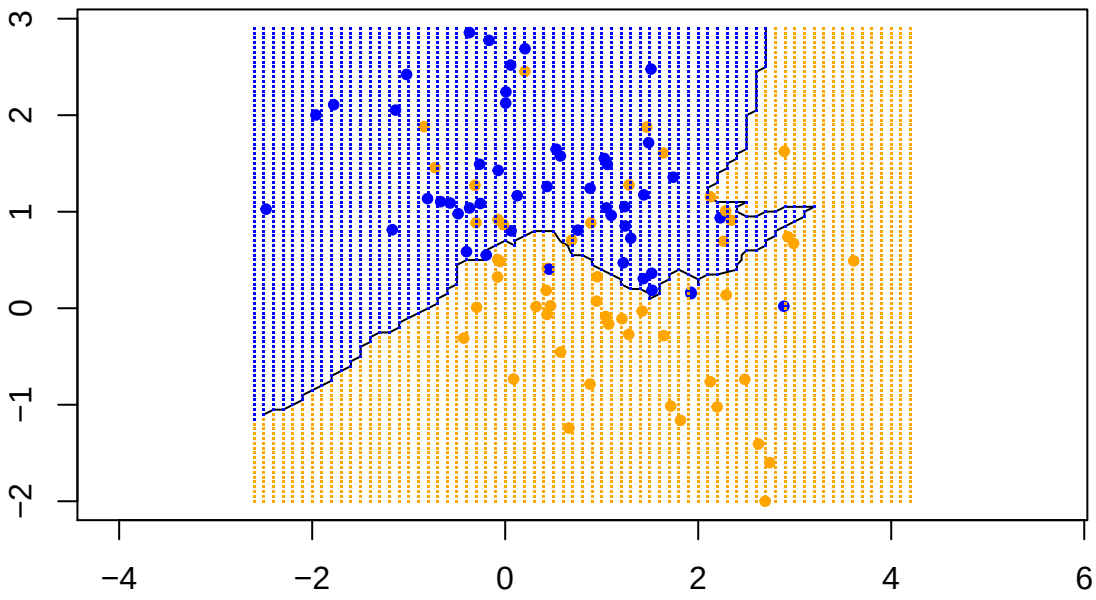

3
2
1
0
−1
−2
−4
−2
0
2
4
6

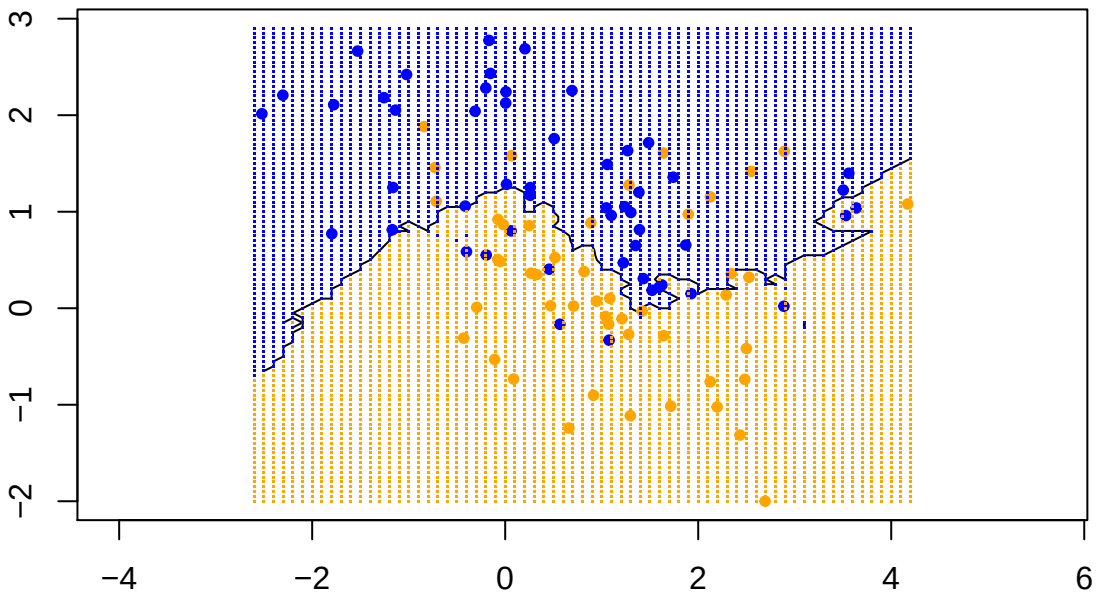



```
sampling.knn(20)
```

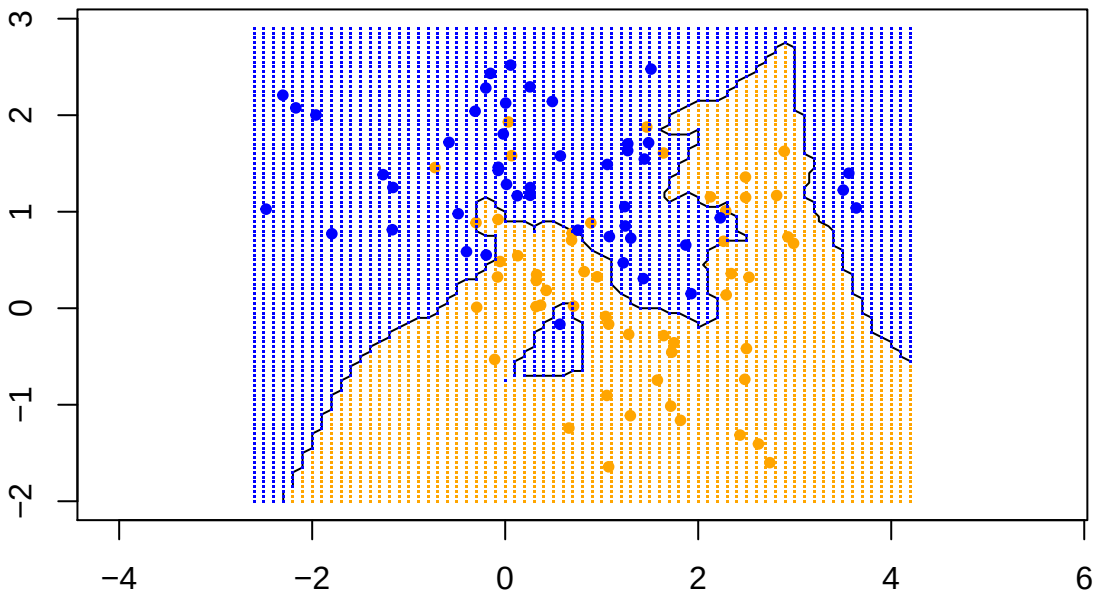

3
2
1
0
−1
−2
−4
−2
0
2
4
6

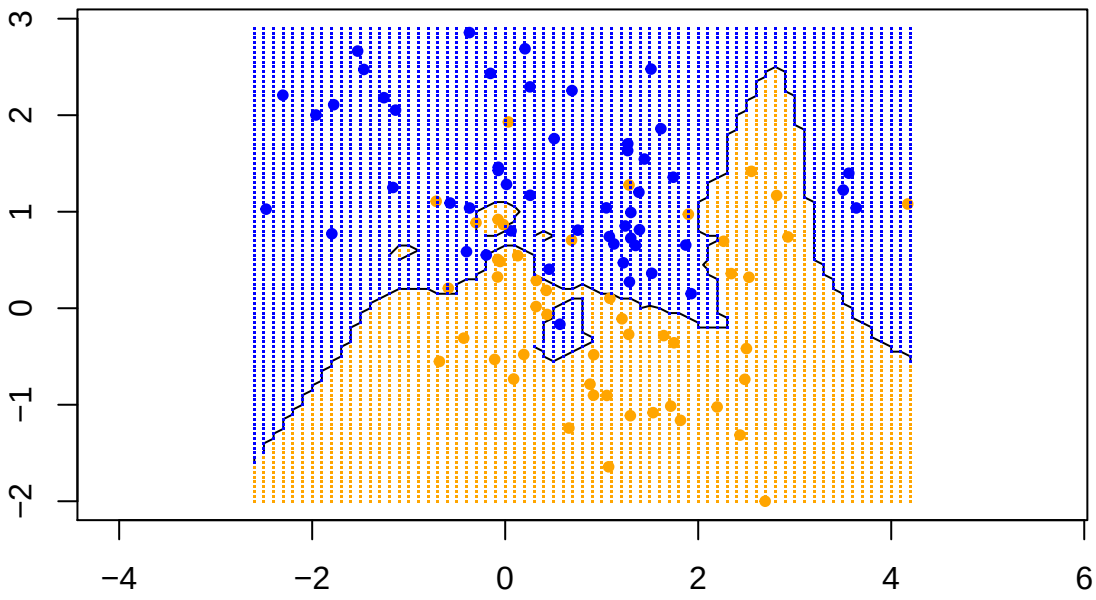

3
2
1
0
−1
−2
−4
−2
0
2
4
6

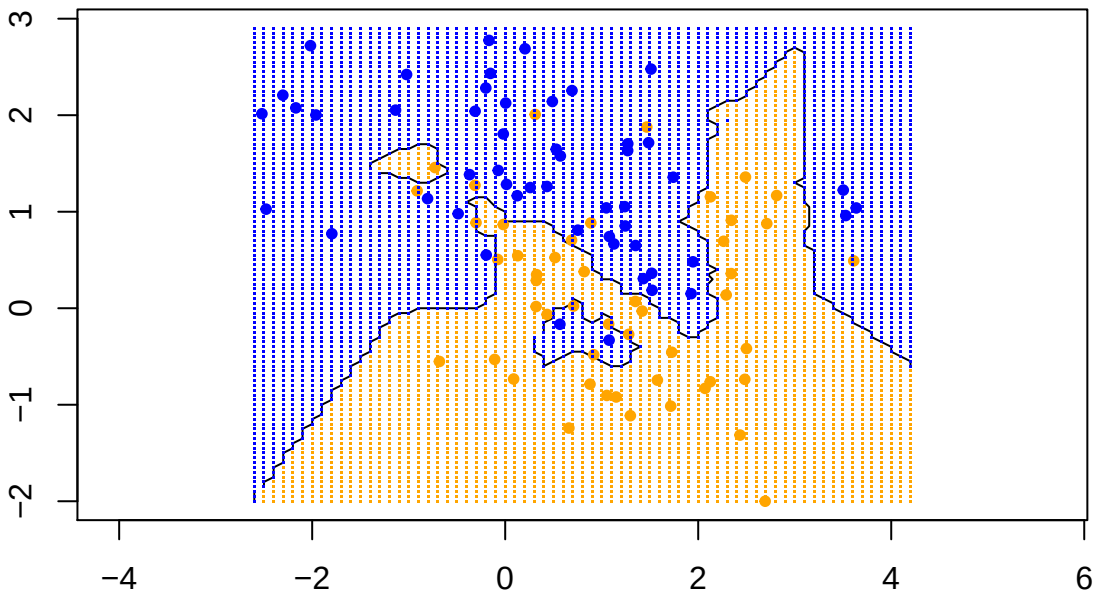

3
2
1
0
−1
−2
−4
−2
0
2
4
6

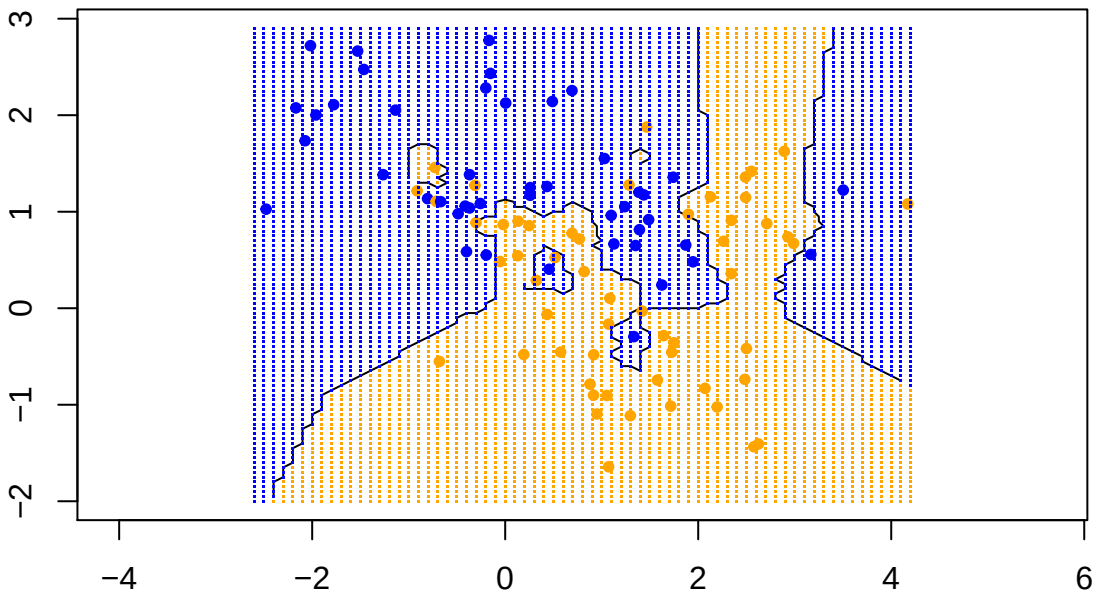



```
sampling.knn(k=2)
```


3
2
1
0
−1
−2
−4
−2
0
2
4
6


3
2
1
0
−1
−2
−4
−2
0
2
4
6


3
2
1
0
−1
−2
−4
−2
0
2
4
6


3
2
1
0
−1
−2
−4
−2
0
2
4
6