Introduction neural networks in R

Hicham Zmarrou, PhD

2018-12-06

Aims of this lesson

Give you a general introduction to neural networks, How they works and how to implement them in R.

What are Neural networks

https://www.youtube.com/watch?v=bxe2T-V8XRs

https://www.youtube.com/watch?v=UJwK6jAStmg

https://www.youtube.com/watch?v=5u0jaA3qAGk&t=45s

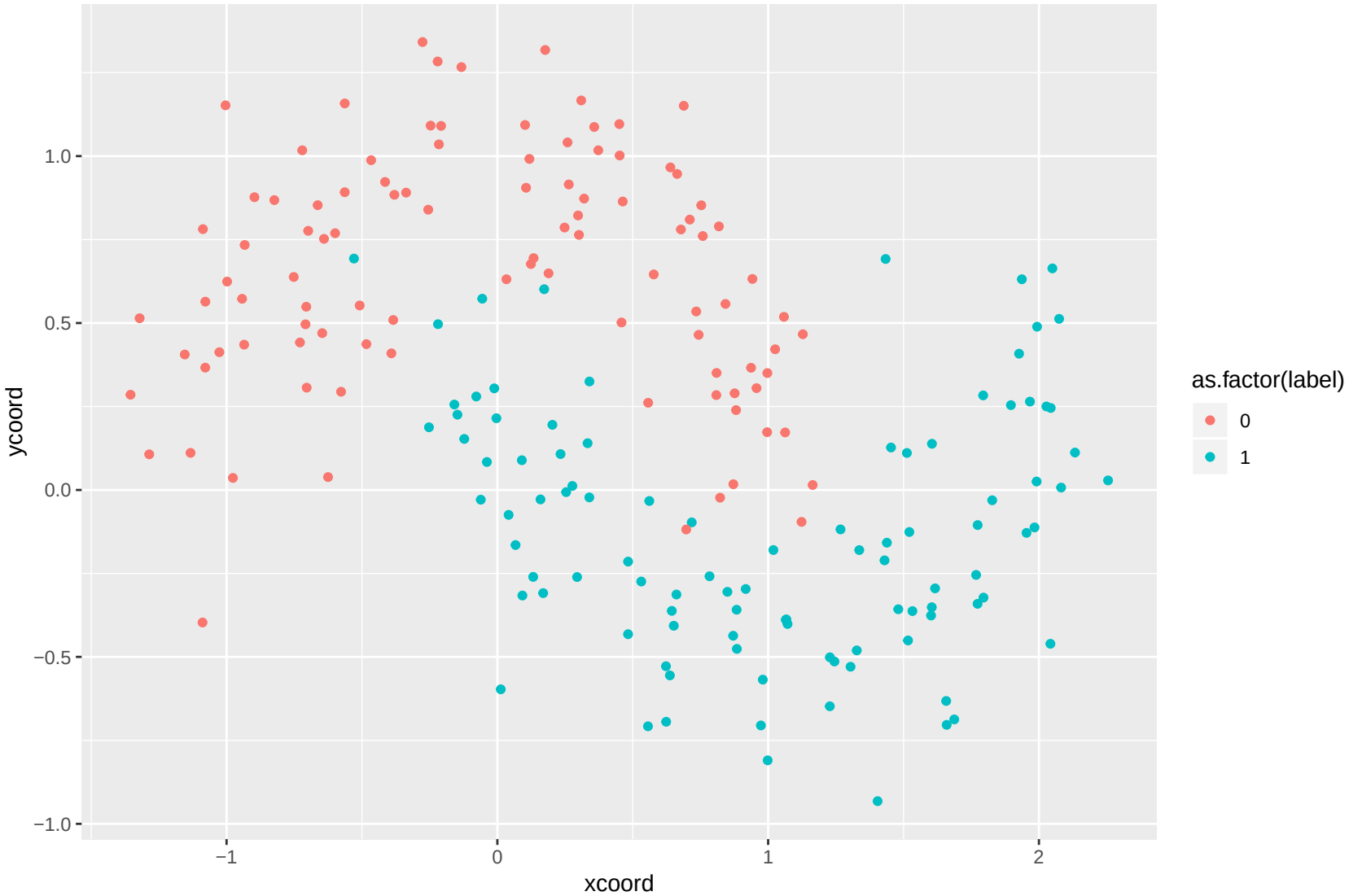
https://www.youtube.com/watch?v=GlcnxUlrtek

https://www.youtube.com/watch?v=pHMzNW8Agq4

https://www.youtube.com/watch?v=9KM9Td6RVgQ

https://www.youtube.com/watch?v=S4ZUwgesjS8

Neural network from scratch



Neural network from scratch

- The dataset we generated has two classes, plotted as red and blue points.

Neural network from scratch

- The dataset we generated has two classes, plotted as red and blue points.
- You can think of the blue dots as male patients and the red dots as female patients, with the x- and y- axis being medical measurements.

Neural network from scratch

- The dataset we generated has two classes, plotted as red and blue points.
- You can think of the blue dots as male patients and the red dots as female patients, with the x- and y- axis being medical measurements.
- Our goal is to train a Machine Learning classifier that predicts the correct class (male of female) given the x- and y- coordinates.

Neural network from scratch

- The dataset we generated has two classes, plotted as red and blue points.
- You can think of the blue dots as male patients and the red dots as female patients, with the x- and y- axis being medical measurements.
- Our goal is to train a Machine Learning classifier that predicts the correct class (male of female) given the x- and y- coordinates.
- Data is not linearly separable, we can't draw a straight line that separates the two classes.

Neural network from scratch

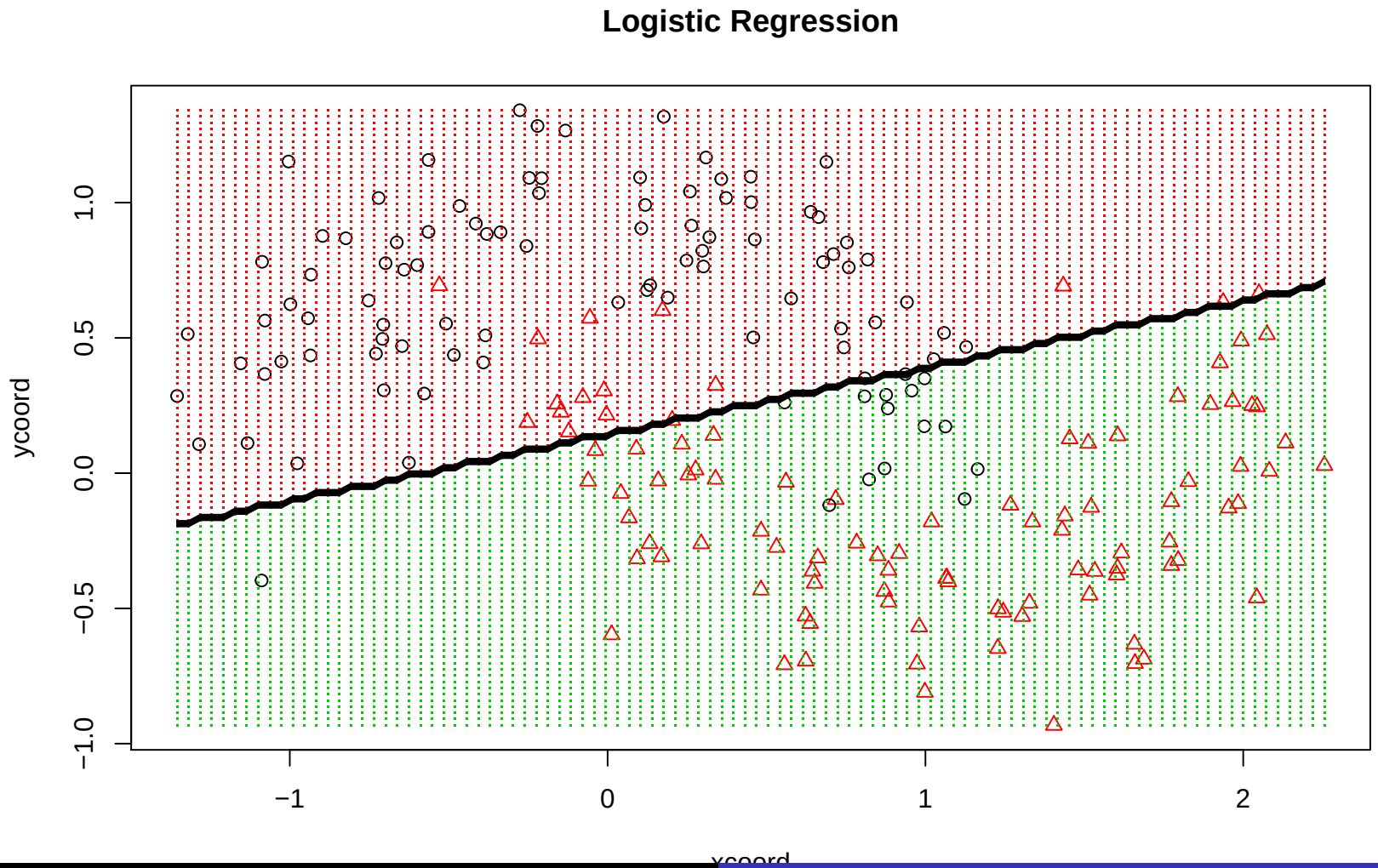
- The dataset we generated has two classes, plotted as red and blue points.
- You can think of the blue dots as male patients and the red dots as female patients, with the x- and y- axis being medical measurements.
- Our goal is to train a Machine Learning classifier that predicts the correct class (male of female) given the x- and y- coordinates.
- Data is not linearly separable, we can't draw a straight line that separates the two classes.
- This means that linear classifiers, such as Logistic Regression, won't be able to fit the data unless you hand-engineer non-linear features (such as polynomials) that work well for the given dataset.

Neural network from scratch

- The dataset we generated has two classes, plotted as red and blue points.
- You can think of the blue dots as male patients and the red dots as female patients, with the x- and y- axis being medical measurements.
- Our goal is to train a Machine Learning classifier that predicts the correct class (male of female) given the x- and y- coordinates.
- Data is not linearly separable, we can't draw a straight line that separates the two classes.
- This means that linear classifiers, such as Logistic Regression, won't be able to fit the data unless you hand-engineer non-linear features (such as polynomials) that work well for the given dataset.
- That's one of the major advantages of Neural Networks. You don't need to worry about feature engineering. The hidden layer of a neural network will learn features for you.

Fitting the logistic regression

```
myplot(model, moons, class = "label",
       main = "Logistic Regression")
```



Training a neural network

- Build a 3-layer neural network with one input layer,

Training a neural network

- Build a 3-layer neural network with one input layer,
  - one input layer, one hidden layer, and one output layer.

Training a neural network

- Build a 3-layer neural network with one input layer,
  - one input layer, one hidden layer, and one output layer.
  - the number of nodes in the input layer is determined by the dimensionality of our data, 2.

Training a neural network

- Build a 3-layer neural network with one input layer,
  - one input layer, one hidden layer, and one output layer.
  - the number of nodes in the input layer is determined by the dimensionality of our data, 2.
  - the number of nodes in the output layer is determined by the number of classes we have, also 2.

Training a neural network

- Build a 3-layer neural network with one input layer,
  - one input layer, one hidden layer, and one output layer.
  - the number of nodes in the input layer is determined by the dimensionality of our data, 2.
  - the number of nodes in the output layer is determined by the number of classes we have, also 2.
  - the input to the network will be xcoord- and ycoord and its output will be two probabilities, one for class red (“female”) and one for class blue (“male”). It looks something like this:

Training a neural network

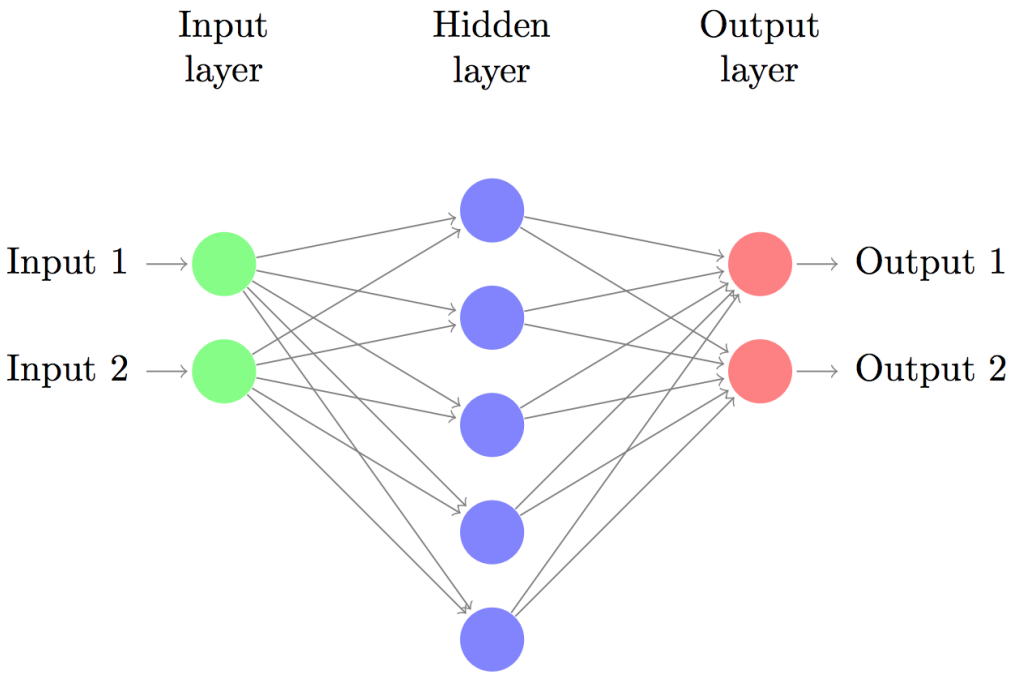


Figure 1: NN illustration

How does the algorithm work?

Go to this nice blog

Implementation in R

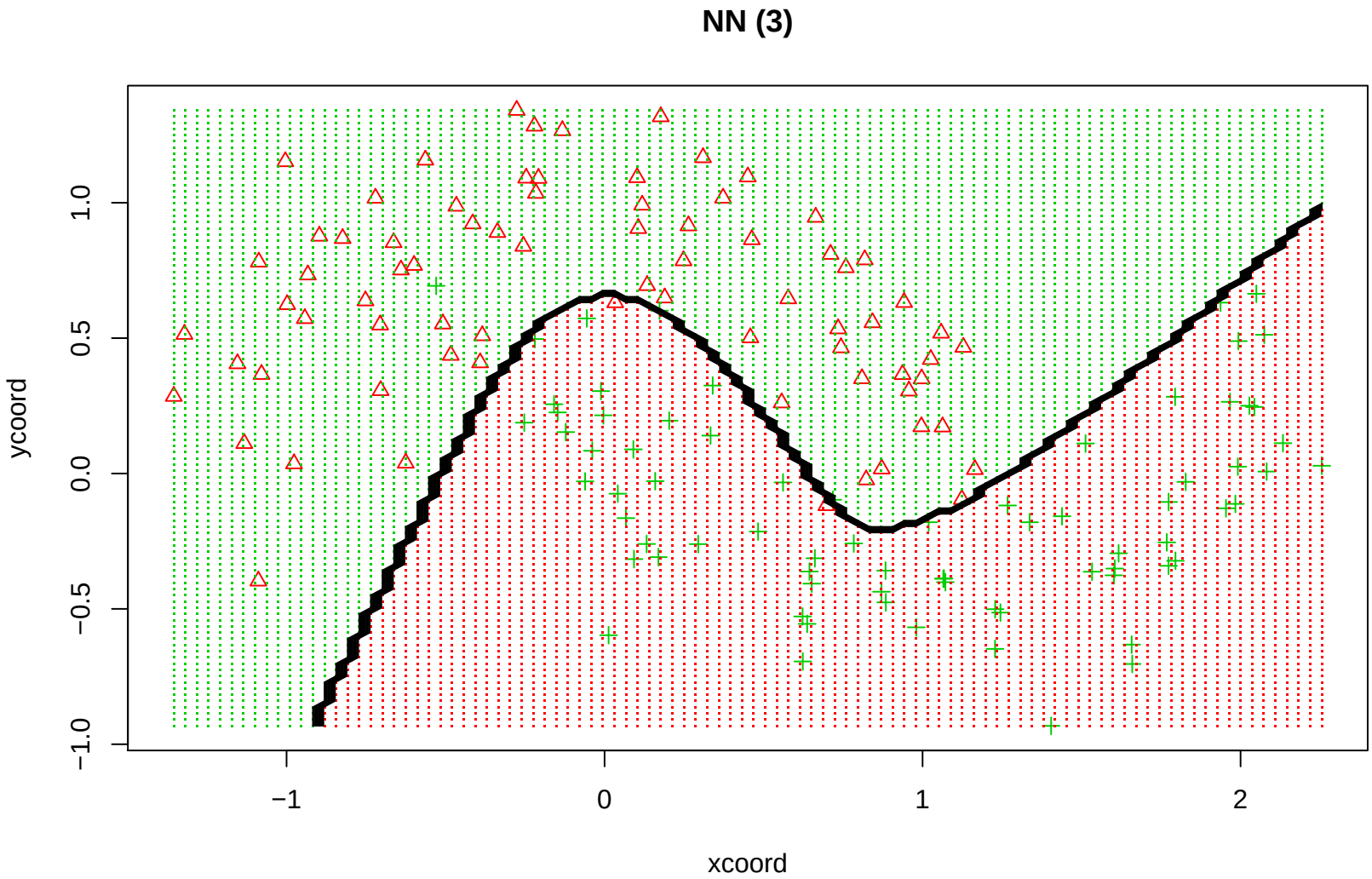
```
## A network with a hidden layer of size 3
x <-  moons[1:150, c("xcoord", "ycoord", "label")]
x$label <- as.factor(x$label)
levels(x$label) <- c("m","f")
head(x)
```

```
      xcoord     ycoord label
1  0.7434612  0.4646563     m
2  1.6575566 -0.6320316     f
3 -0.1587887  0.2558446     f
4 -1.0887520 -0.3969432     m
5  1.7680520 -0.2544321     f
6  1.9541645 -0.1285058     f
```

```
library(nnet)
model <- nnet(label ~ ., data=x, size =3, maxit = 1000, tra
```

A network with a hidden layer of size 3

```
myplot(model, x, class = "label", main = "NN (3)")
```



Circle dataset

```
set.seed(1000)
library(mlbench)
x <- mlbench.circle(100)
x <- cbind(as.data.frame(x$x), factor(x$classes))
colnames(x) <- c("x", "y", "class")

head(x,3)
```

```
           x           y class
1 -0.3442426  0.19873584     1
2  0.5176930 -0.09663637     1
3 -0.7721272 -0.72634916     2
```

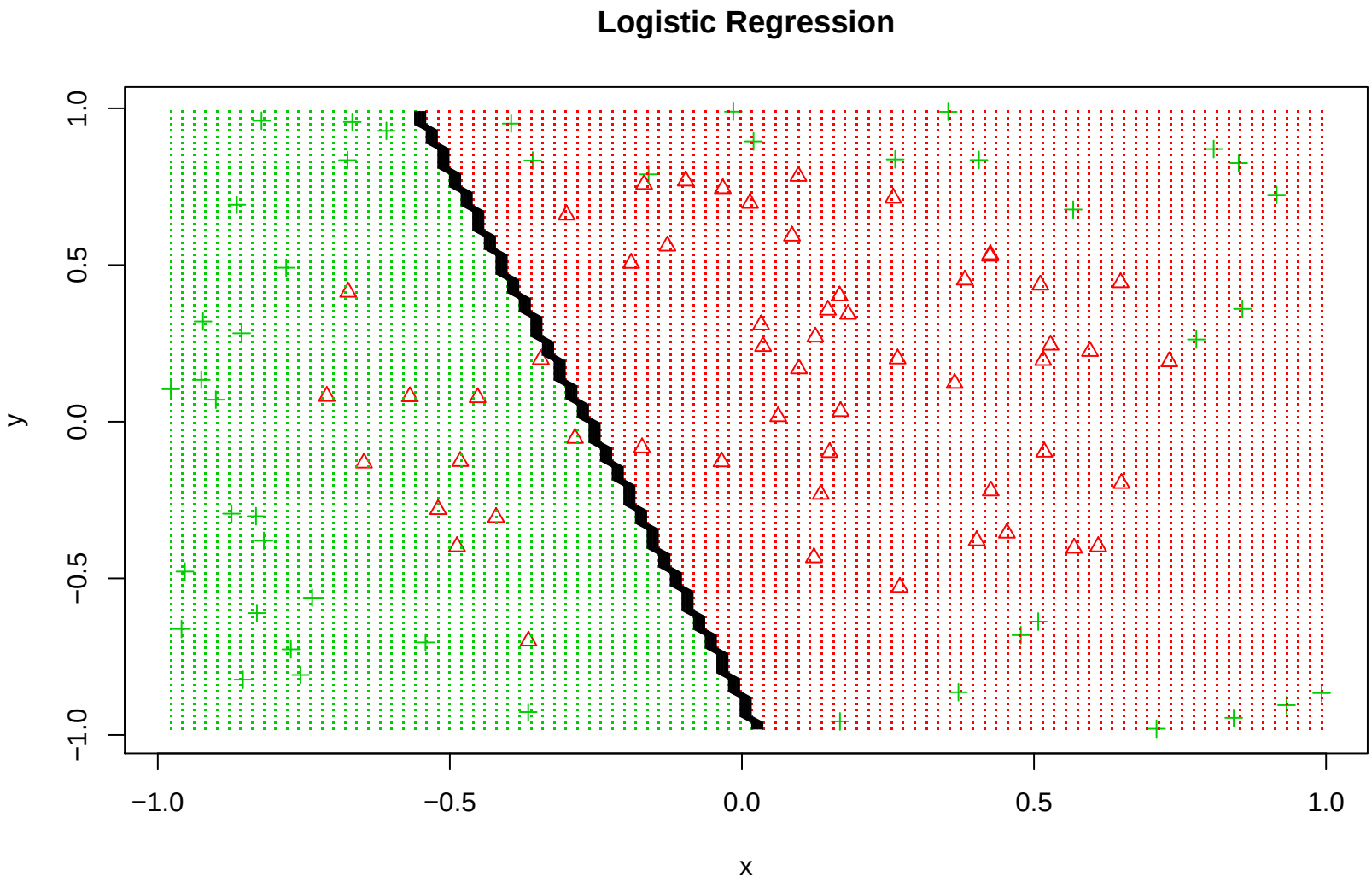
Circle dataset (Logistic Regression)

Logistic Regression Only considers for 2 classes

```
model <- glm(class ~., data = x, family=binomial(link='logi
class(model) <- c("lr", class(model))
predict.lr <- function(object, newdata, ...)
  predict.glm(object, newdata, type = "response") > .5
```

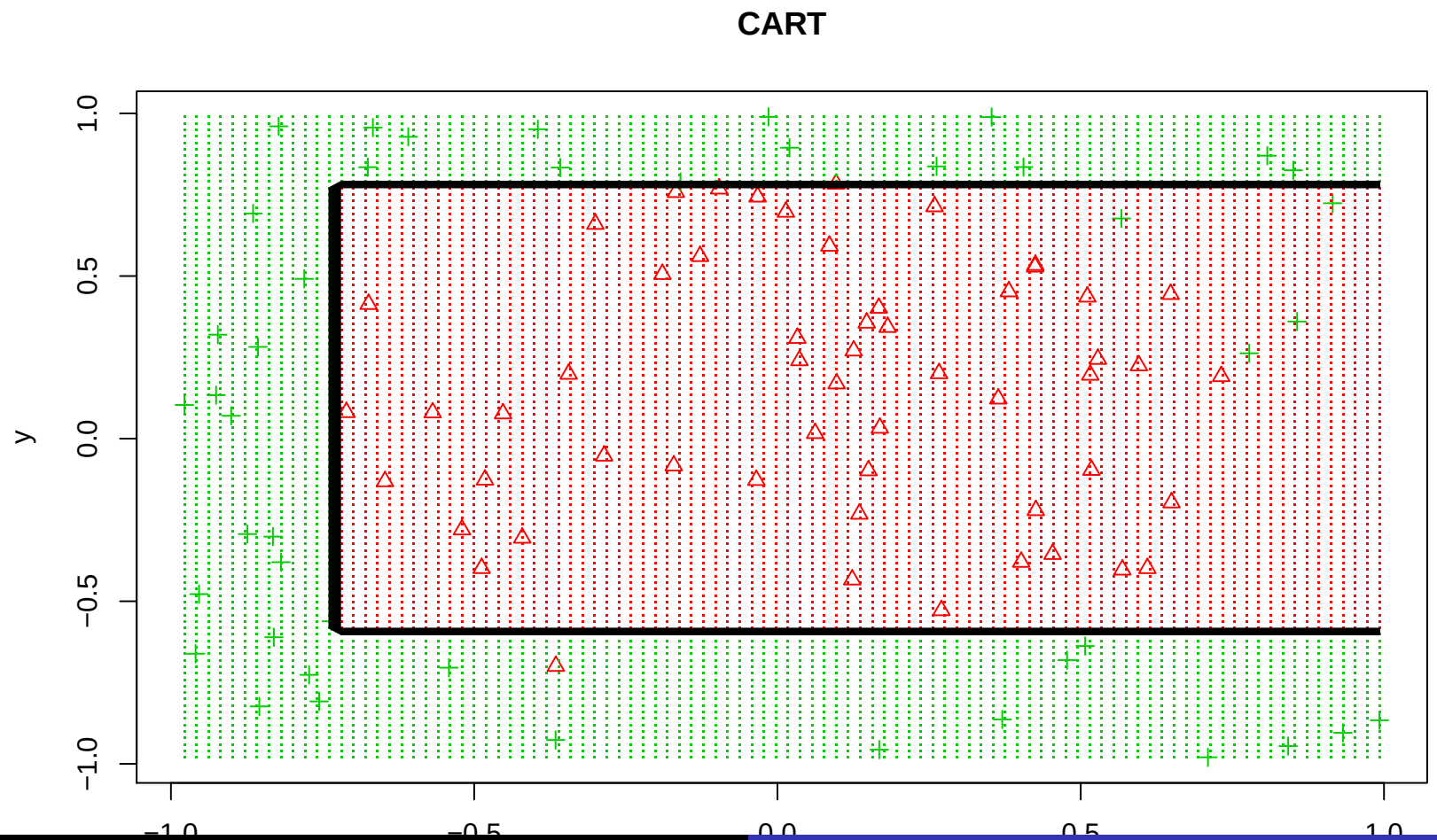
Circle dataset (Logistic Regression)

```
myplot(model, x, class = "class", main = "Logistic Regressi
```



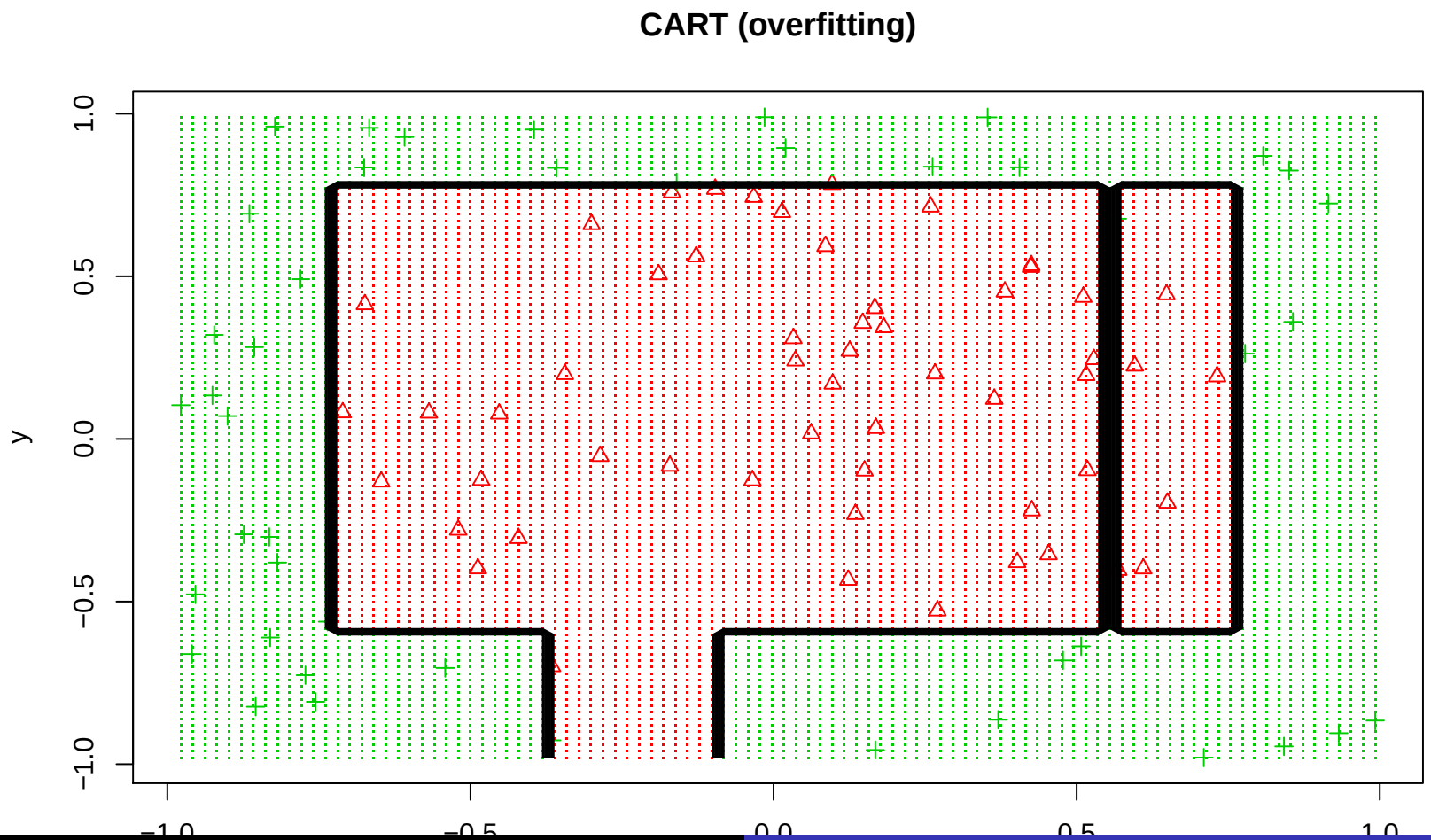
Circle dataset (Decision trees)

```
library("rpart")
model <- rpart(class ~ ., data=x)
myplot(model, x, class = "class", main = "CART")
```



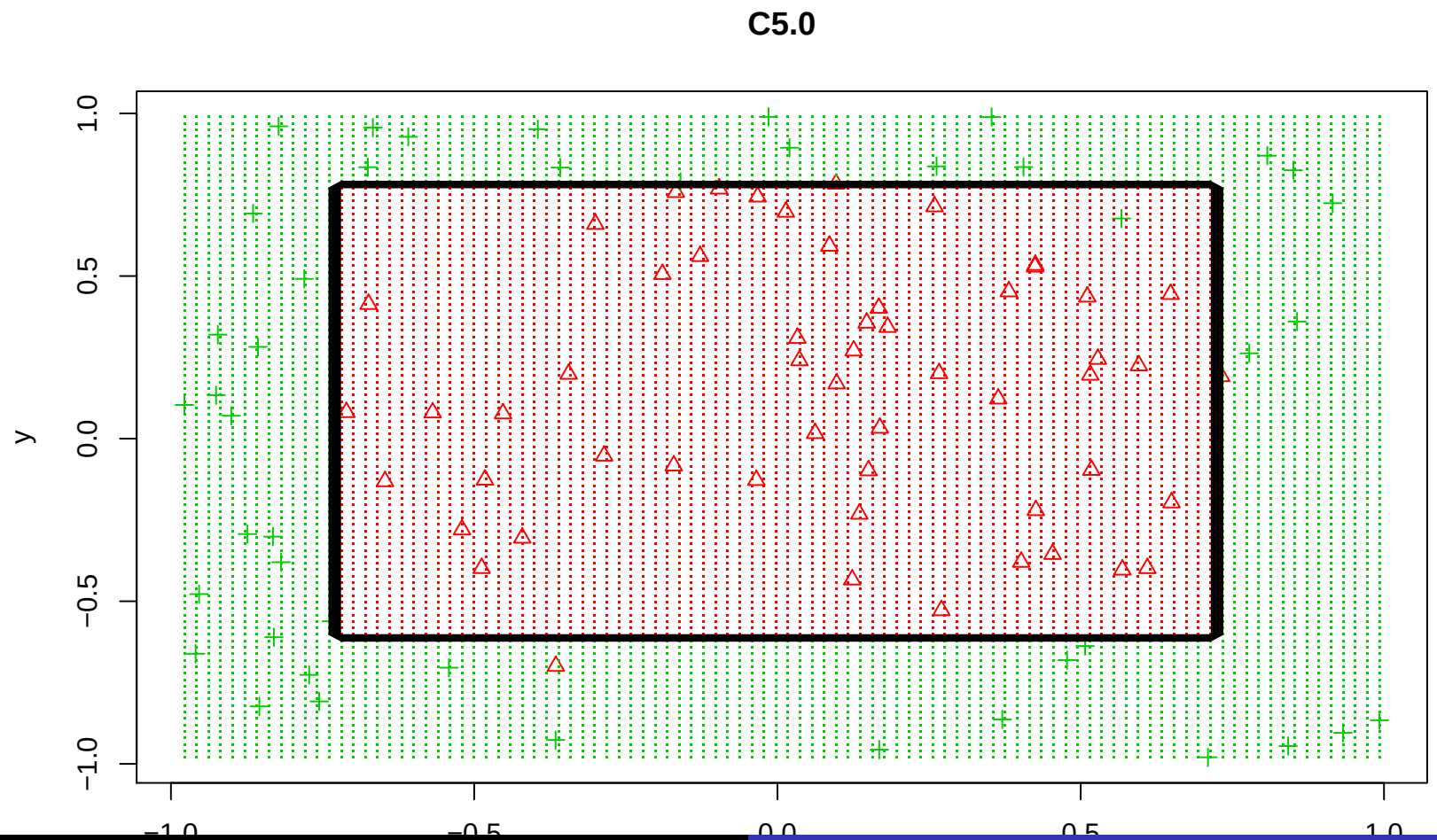
Circle dataset (Decision trees overfitting)

```
model <- rpart(class ~ ., data=x,
  control = rpart.control(cp = 0.001, minsplit = 1))
myplot(model, x, class = "class", main = "CART (overfitting
```



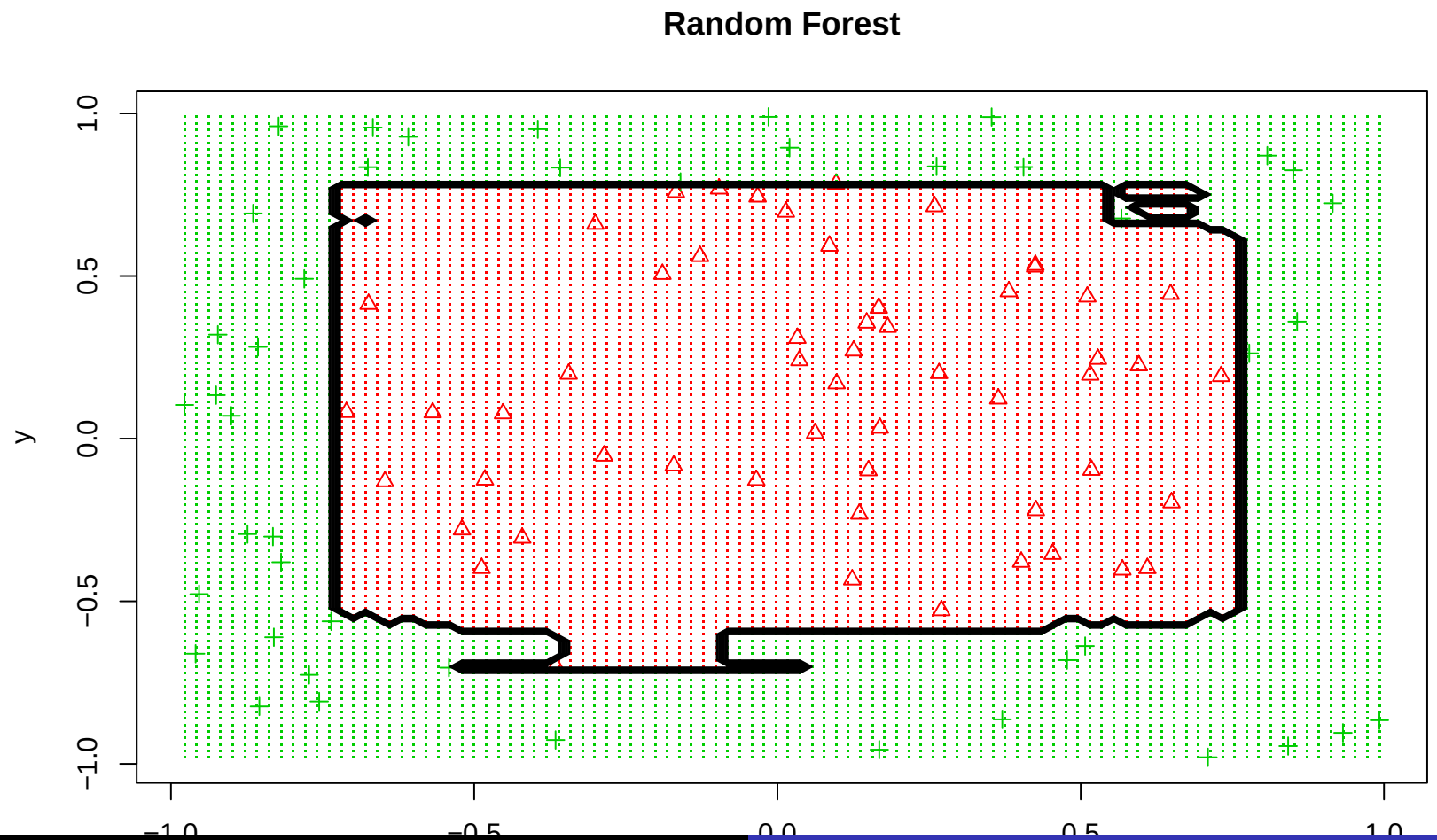
Circle dataset (Decision trees C5.0)

```
library(C50)
model <- C5.0(class ~ ., data=x)
myplot(model, x, class = "class", main = "C5.0")
```



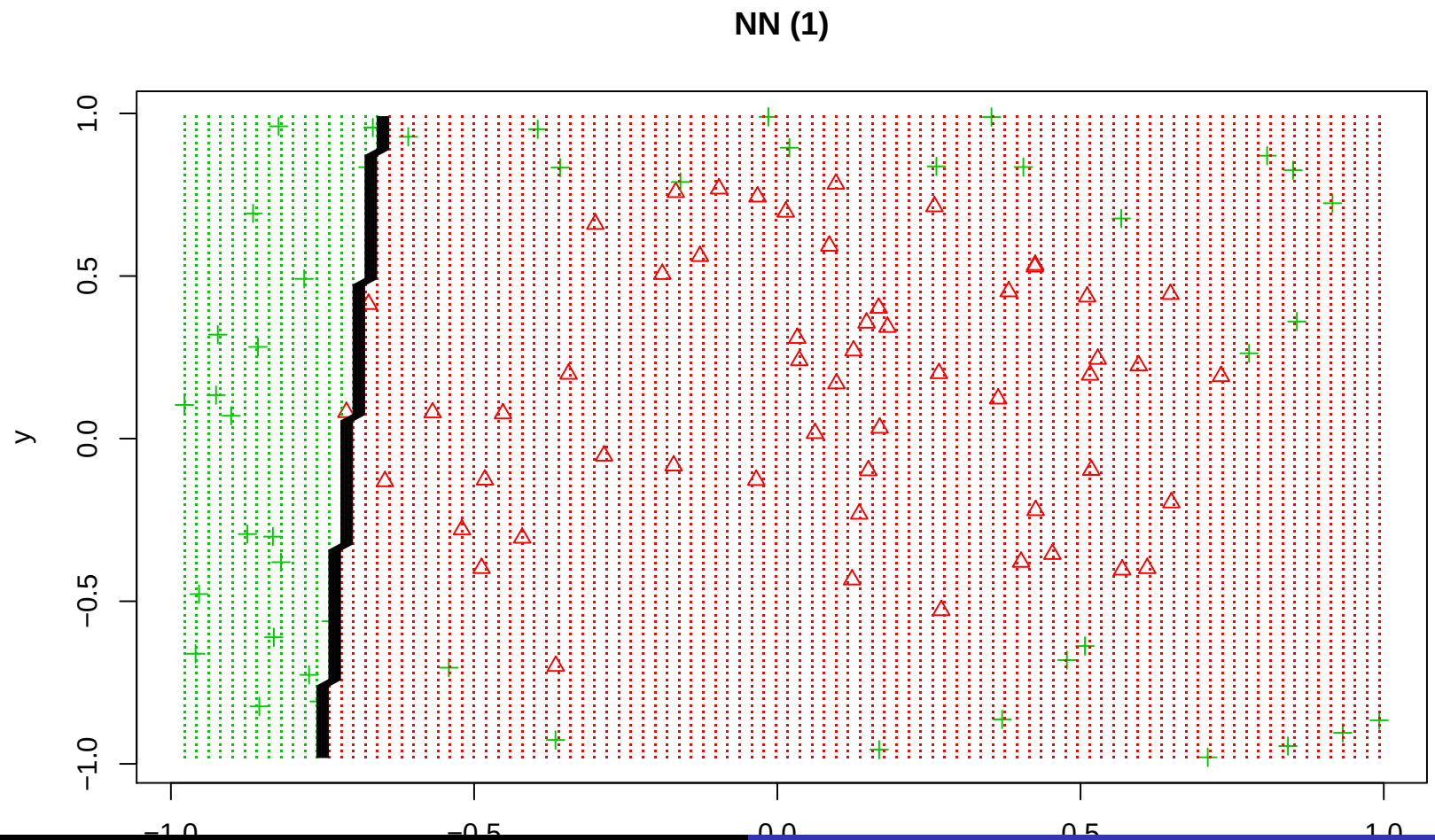
Circle dataset (Random Forest)

```
library(randomForest)
model <- randomForest(class ~ ., data=x)
myplot(model, x, class = "class", main = "Random Forest")
```



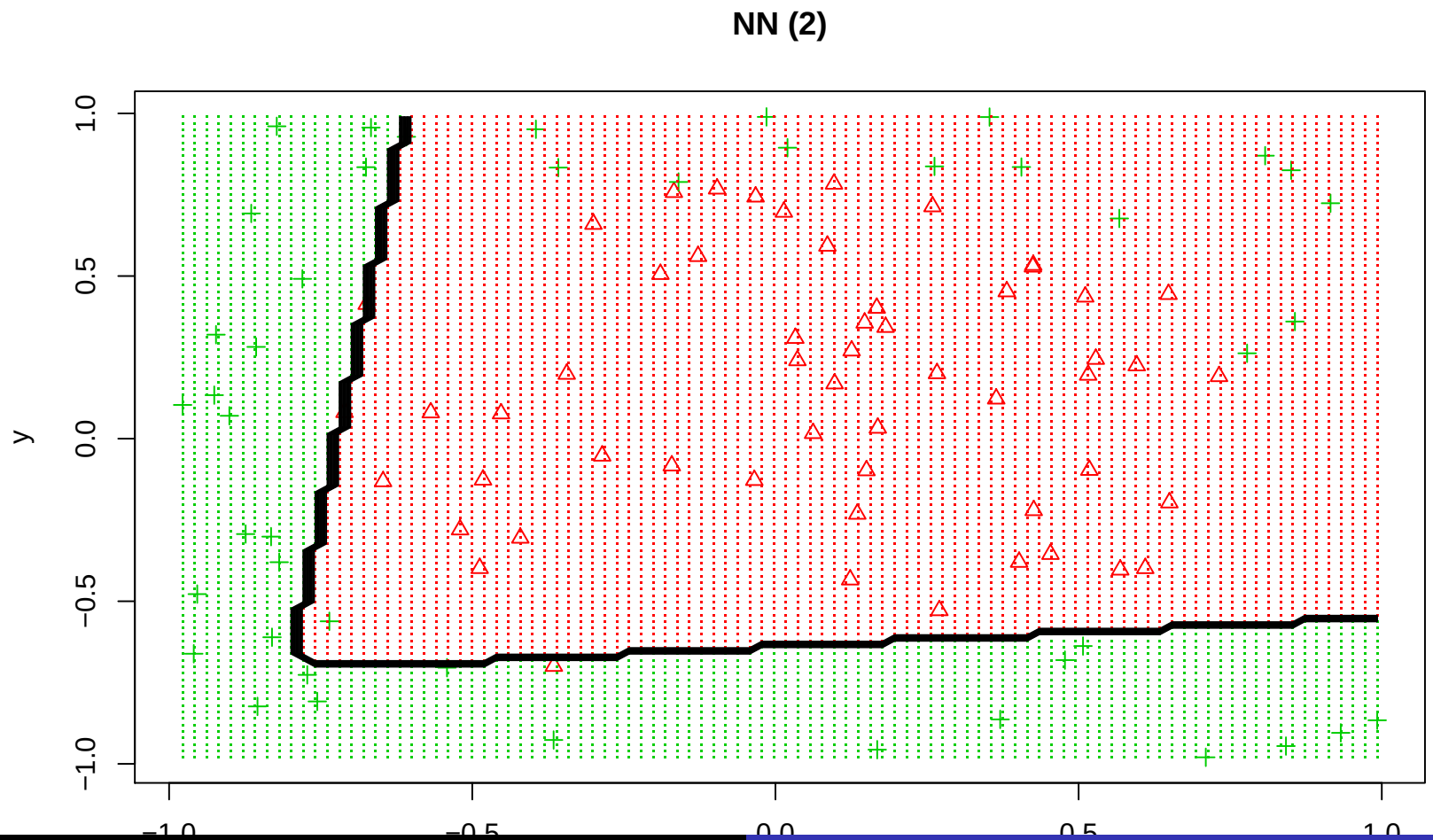
Circle dataset (Neural Network)

```
library(nnet)
model <- nnet(class ~ ., data=x, size = 1, maxit = 1000, tr
myplot(model, x, class = "class", main = "NN (1)")
```



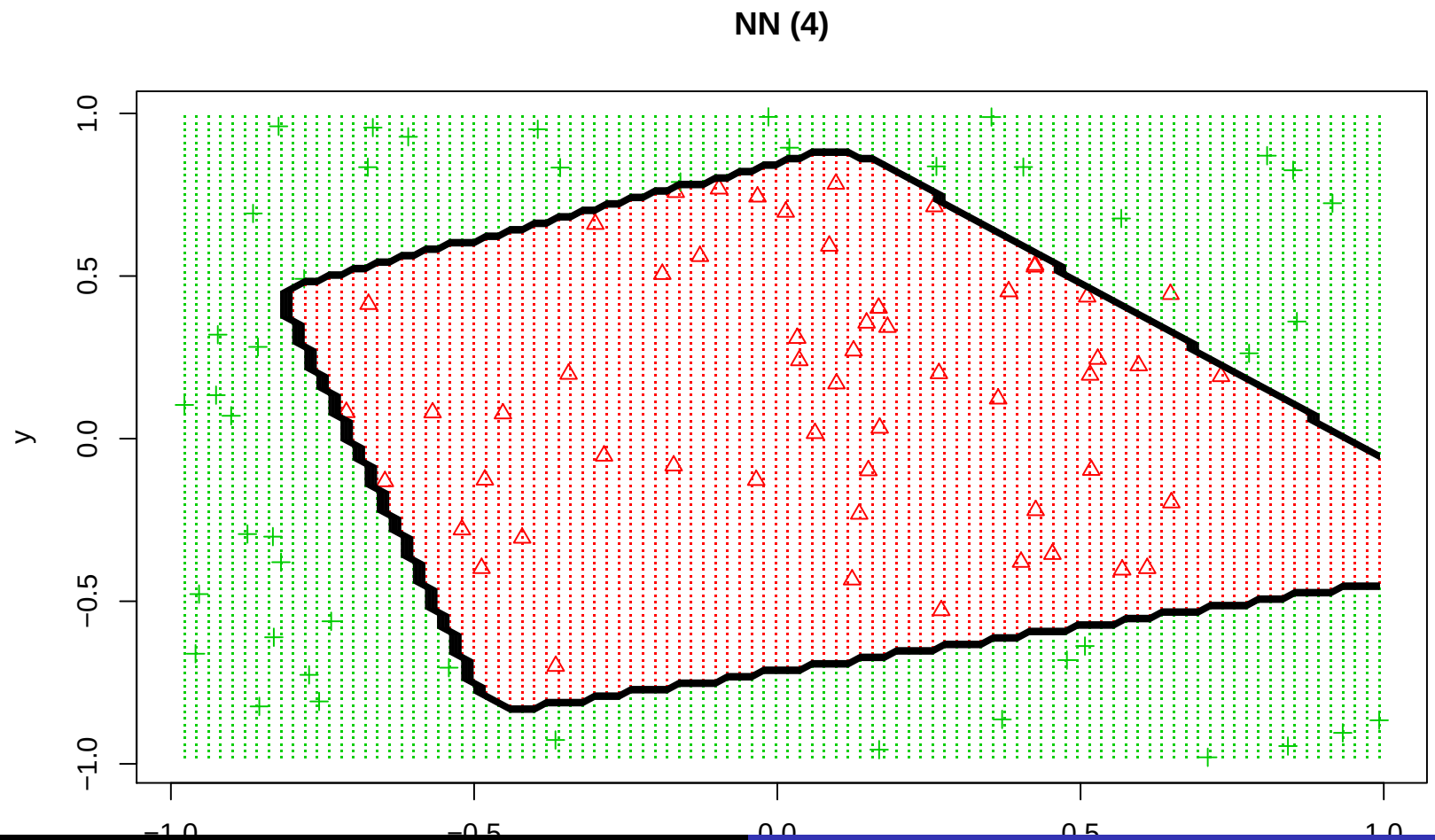
Circle dataset (Neural Network)

```
library(nnet)
model <- nnet(class ~ ., data=x, size = 2, maxit = 1000, tr
myplot(model, x, class = "class", main = "NN (2)")
```



Circle dataset (Neural Network)

```
library(nnet)
model <- nnet(class ~ ., data=x, size = 4, maxit = 1000, tr
myplot(model, x, class = "class", main = "NN (4)")
```



Circle dataset (Neural Network)

```
library(nnet)
model <- nnet(class ~ ., data=x, size = 10, maxit = 1000, t
myplot(model, x, class = "class", main = "NN (10)")
```