Plotting Trees and Response Curves for Regression and Classification Problems

Tshiamo Phaahla

18/11/2021

Please Note The Following.

The content contained in this document is from the first lab session in a series of lectures prepared for a two-week introductory course in Machine Learning at the University of Cape Town, South Africa. The course is aimed at students with some background in statistical modelling, computing, and linear algebra. It is recommended that you watch the first two Key-Point Lectures in this series before tackling the lab session.

Link to the the relevant lab https://youtu.be/2ki7Yi2Hx3U

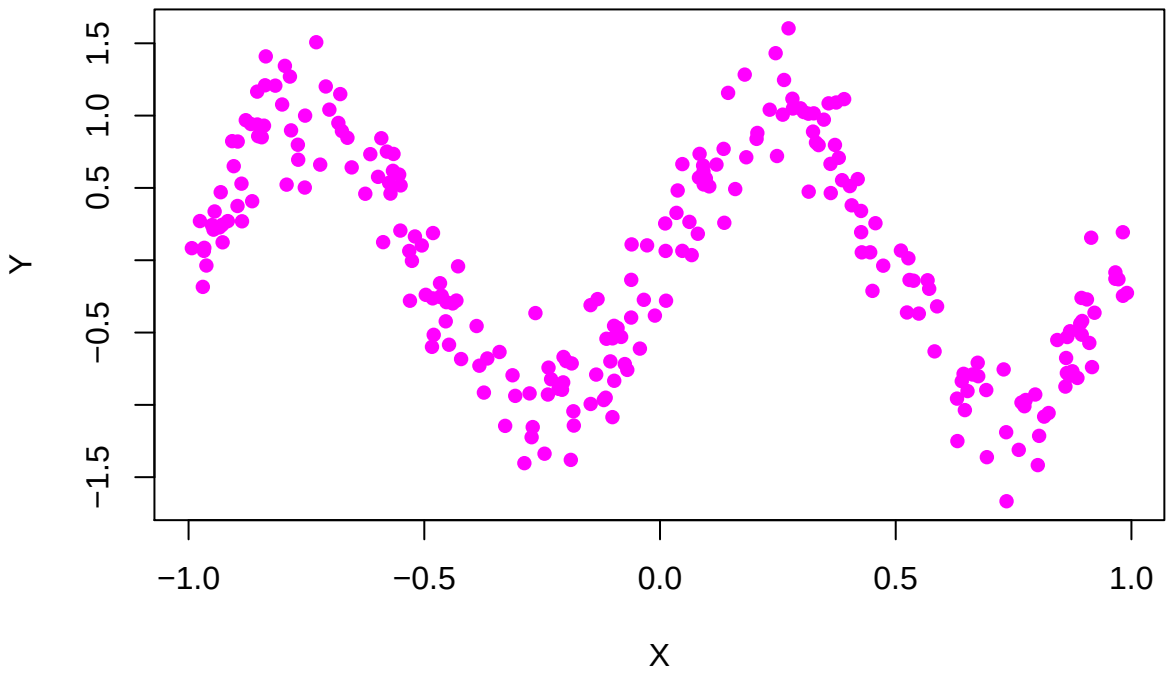
==========

Regression Trees

Goal: Simulate a non-linear (reg.) pattern, apply the partitionin algorithm to the pattern for various stopping criteria, and draw response curves to demonstrate model complexity.

Task 1: Simulate and plot simple sinusoidal pattern with noise.

```
set.seed(2021)
N = 250
X = runif(N, -1, 1)
e = rnorm(N, 0, 0.25)
Y = sin(2*pi*X) + e
plot(Y~X, pch=16, col="magenta")
```

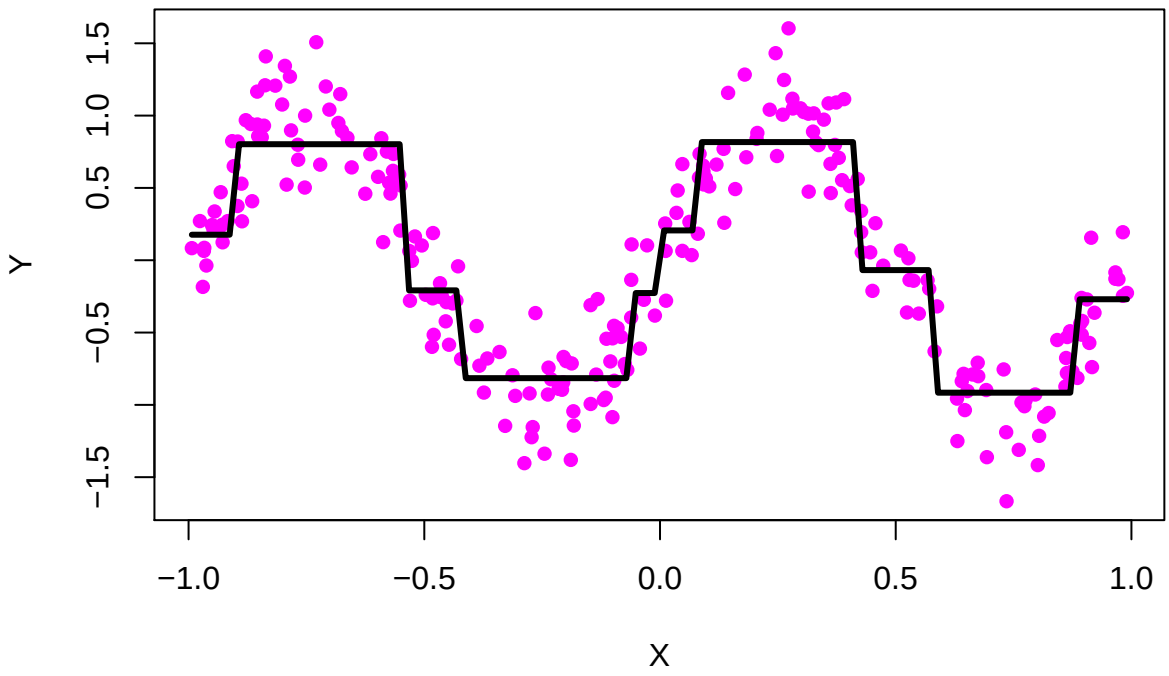


Task 2: Call the `tree` library and fit a regression tree to the data.

```
library(tree)

res = tree(Y~X)
```

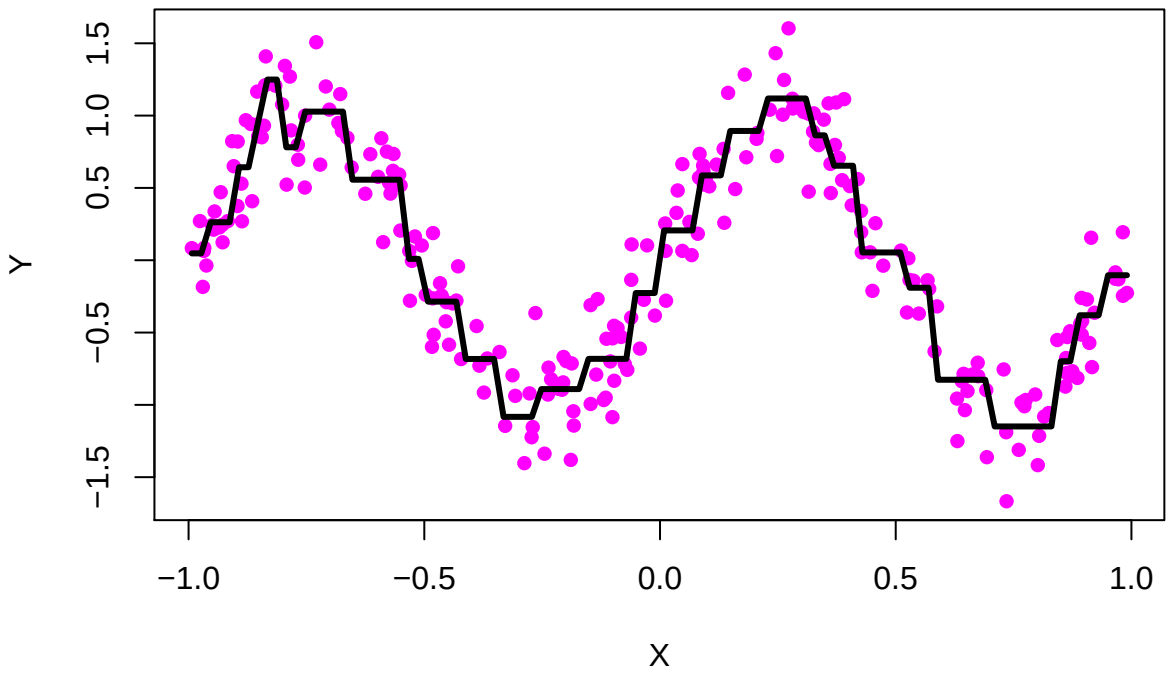
Task 3: Draw a response curve to demostrate the predicted pattern under a regression tree model.

```
M = 100
X_dummy = seq(min(X), max(X), length=M)
Lat = data.frame(X = X_dummy)
pred_Y = predict(res, Lat)
plot(Y~X, pch=16, col='magenta')
lines(pred_Y~ Lat$X, lwd=3)
```



Task 4: Modify the stopping criterion for the partitioning algorithm and redraw the response curve.

```
res1 = tree(Y~X, control=tree.control(N,5,10,0.001))

plot(Y~X, pch=16, col="magenta")
pred_Y1 = predict(res1, Lat)
lines(pred_Y1~Lat$X, lwd=3)
```

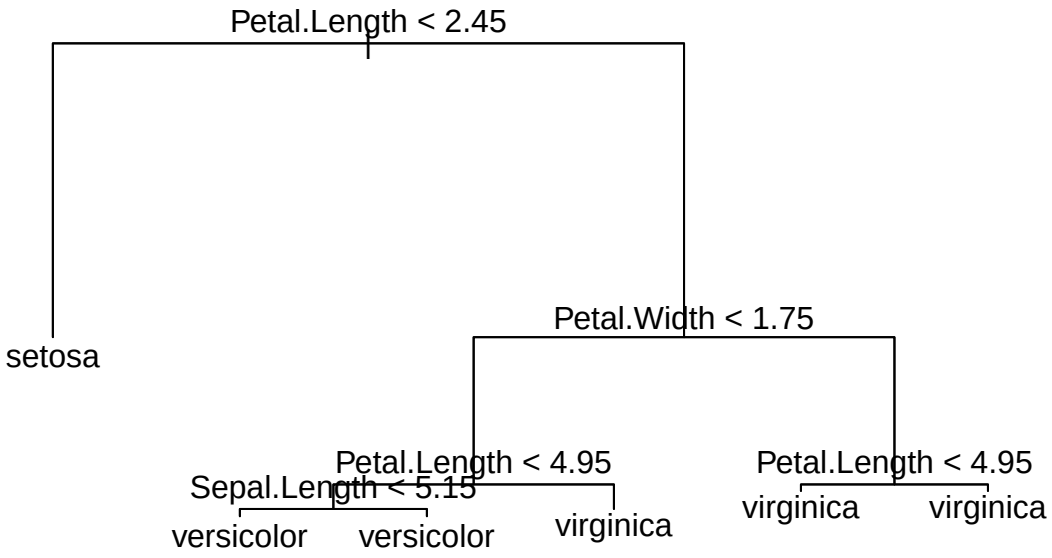


Classification Trees

Goal: Use a tree-based model to analyse the iris data in R. Plot the fitted trees and interpret. (Fit and) Draw a response curve for the Petal-inputs and superimpose the observations on the response curve.

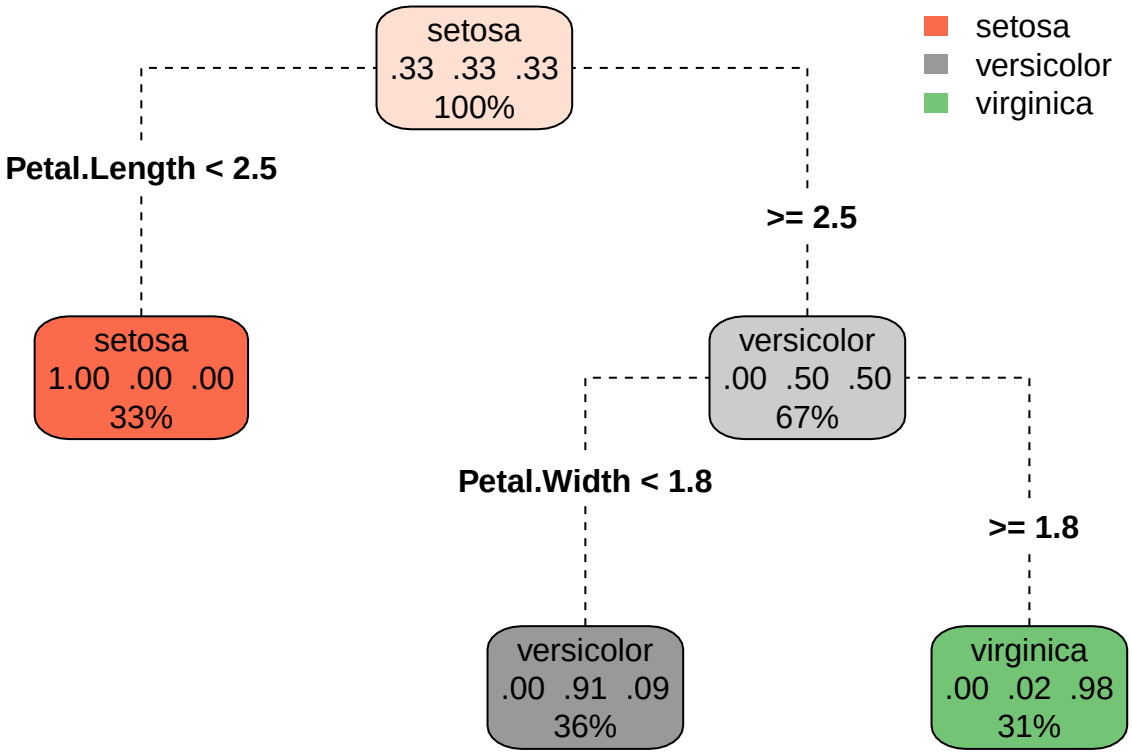
Task 1: Load the iris data and fit a tree-based model to the data. Plot and interpret the tree.

```
data(iris)
attach(iris)
res2 = tree(Species~.,data=iris, control = tree.control(length(Species), 5, 10, mindev=0.01))

plot(res2)
text(res2)
```



Task 2: Re-run the analysis using the rpart/rpart.plot library. Play around with the plot parameters to draw a more readable tree.

```
library(rpart)
library(rpart.plot)

res4 = rpart(Species~., data=iris, control= rpart.control(cp=0.01))
rpart.plot(res4,  type=4, fallen.leaves = F, branch = 1, branch.lty=2)
```



Task 3: Fit a tree using only the petal-inputs and plot a response curve for the fitted tree.

Here what we will be doing is creating a lattice. We will also be creating vectors of x1 coords and x2 ccords which correpsond to point where all these lines intersect by simply repeating all of the elements in the dummy sequence that we created in the apprporiate fashion

```
res4 = rpart(Species~ Petal.Length + Petal.Width, data=iris,
             control=rpart.control(cp=0.01))
M=200
X1_dummy = seq(min(Petal.Width), max(Petal.Width), length=M)
X2_dummy = seq(min(Petal.Length), max(Petal.Length), length=M)


#Taking coordinates of predictors and predicting the response for those coordinates

x1 = rep(X1_dummy, M)
x2 = rep(X2_dummy, each=M)

#plot(x2~x1)

Lat2 = data.frame(Petal.Width=x1, Petal.Length=x2)

pred = predict(res4, newdata = Lat2, type='prob')

#Find whichever has the max probability from our predictions and classify
```

```
clss = apply(pred, 1, which.max)

cols = c('blue', 'gray', 'purple')

plot(x2~x1, pch=16,  col=cols[clss])

text(iris$Petal.Length~iris$Petal.Width, labels = as.numeric(iris$Species))
```