```
# Appendix 2

# Libraries
library(tree)
library(boot)


# Setup
dataframe = read.csv2("State.csv")
data.sorted = dataframe[order(dataframe$MET),] # sorted data
n = dim(data.sorted)[1]                        # number of observations
tree.control = tree.control(n, minsize=8)      # control

# Functions

# Returns decision tree
get_tree = function(){
  return(tree(EX~MET, data=data.sorted, control=tree.control))
}

# Returns pruned decision tree
get_pruned_tree = function(tree, leaves){
  return(prune.tree(tree, best=leaves))
}

# Calculates the residuals
calculate_residuals = function(predictions, observations){
  return(observations-predictions)
}

# Returns residuals of given model
get_residuals = function(mle){
  fitted = predict(mle, data.sorted)
  return(calculate_residuals(fitted, data.sorted$EX))
}

# Returns a tree fitted on given data
fit_tree = function(in.data){
  return(tree(EX~MET, data=in.data, control=tree.control))
}

# Returns predictions for non-parametic bootstrapping
f1 = function(in.data, ind){
  data1 = in.data[ind,]
  tree1 = fit_tree(data1)
  pruned.tree1 = get_pruned_tree(tree1, 3)
  fitted1 = predict(pruned.tree1, newdata=data.sorted)
  return(fitted1)
}

# Returns predictions for parametic bootstrapping
f2 = function(in.data){
  tree2 = fit_tree(in.data)
  pruned.tree2 = get_pruned_tree(tree2, 3)
  fitted2 = predict(pruned.tree2, newdata=data.sorted)
  return(fitted2)
}

# Returns predictions for parametic bootstrapping, with a distribution added to the predicionts
f3 = function(in.data){
  tree3 = fit_tree(in.data)
  pruned.tree3 = get_pruned_tree(tree3, 3)
  fitted3 = predict(pruned.tree3, newdata=data.sorted)
  predicted3 = rnorm(n, fitted3, sd(get_residuals(mle)))
  return(predicted3)
}

# Returns data with values "moved" around under the assumption that Y~N(μi, sigma^2)
rng = function(in.data, mle){
  data = data.frame(EX=in.data$EX, MET=in.data$MET)
  data$EX = rnorm(n, predict(mle, newdata=data), sd(get_residuals(mle)))
  return(data)
}

# Plots the confidence band of given envelope and predictions
plot_confidence_band = function(e, fitted, main){
  plot(data.sorted$MET, data.sorted$EX, col="blue", ylim=c(150, 475), main=main, xlab="MET", ylab="EX")
  points(data.sorted$MET, fitted, type="l")
  points(data.sorted$MET, e$point[2,], type="l", col="blue")
  points(data.sorted$MET, e$point[1,], type="l", col="blue")
```

```r
}



# Task 1 - Plot EX vs. MET
plot(data.sorted$MET, data.sorted$EX, xlab="MET", ylab="EX", main="EX vs. MET") # Some sort of regressional model, 2nd
polynomial


# Task 2
# Fit the decision tree
fit.tree = get_tree()
set.seed(12354)

# Cross-validation
cv.fit.tree = cv.tree(fit.tree)

# Plot deviations compared to size=# of leaves & dev compared to k = value of cost-complexity pruning parameter of each
tree
plot(cv.fit.tree$size, cv.fit.tree$dev, type="b", col="forestgreen",
     main="Deviance vs. # of leaves", xlab="# of leaves", ylab="Deviance") # size=3 leaves gives minimum deviance
plot(log(cv.fit.tree$k), cv.fit.tree$dev, type="b", col="forestgreen",
     main="Deviance vs Cost-Complexity", xlab="Cost-Complexity", ylab="Deviance") # k=160486 leaves gives minimum
deviance

# Generate final tree, 3 leaves chosen
pruned.tree = get_pruned_tree(fit.tree, 3)
plot(pruned.tree)
text(pruned.tree, pretty=0)

# Make predictions
pruned.fitted = predict(pruned.tree, newdata=data.sorted)

# Plot predictions and actual values
plot(data.sorted$MET, data.sorted$EX, col="blue" , xlab="MET", ylab="EX", main="EX vs. MET")
points(data.sorted$MET, pruned.fitted, col="red")
legend("topright", legend=c("Fitted Data", "Original Data"), lty=1, col=c("red", "blue"))

# Plot the residuals
residuals = calculate_residuals(pruned.fitted, data.sorted$EX)
hist(residuals, main="Residuals", xlab="Residual value", col="forestgreen", xlim=c(-125, 125)) # somewhat evenly spread



# Task 3 - Non-parametic: Confidence band
set.seed(12345)
boot1 = boot(data.sorted, f1, R=1000)
e1 = envelope(boot1)
plot_confidence_band(e1, pruned.fitted, "Confidence band for Non-Parametic Bootstrap")



# Task 4 - Parametic: Confidence band and prediction band
set.seed(12345)
mle = pruned.tree
# Confidence band
boot2 = boot(data.sorted, statistic=f2, R=1000, mle=mle, ran.gen=rng, sim="parametric")
e2 = envelope(boot2)
plot_confidence_band(e2, pruned.fitted, "Confidence band for Parametic Bootstrap")

# Prediction band
set.seed(12345)
boot3 = boot(data.sorted, statistic=f3, R=1000, mle=mle, ran.gen=rng, sim="parametric")
e3 = envelope(boot3)
plot_confidence_band(e3, pruned.fitted, "Prediction band for Parametic Bootstrap")
```