Elif Küçük
0040851
COMP 421
HW 05 Report.

COMP/INDR 421/521 HW 05: Decision Tree Regression– Report

This file briefly explains each function and variable defined in the project. The defined function names and variables are written in bold. The code also has the comment lines to make understating easier.

Question 1

In order to separate data into two sets of vector with random order I used sample function of R and chosen 100 indices for training data. To separate test from the training data I used setdiff function of R and took the indices for test data.

In the HW we store the training and test data in **data_train** and **data_test** respectively.

Question 2

In order to learn a decision tree I'm using two key functions **g(**indices_data**)** and **E_m(**indices_data**)** which respectively define estimated value in a node, and which evaluates the goodness of split as root mean square.

In order to learn a decision tree we first define the following data structures:
need_splits : boolean array that stores whether node indicated by index needs split.
node_splits: array that stores the split points of each node. Node is indicated by the index of the array.
is_terminal : boolean array that stores whether node indicated by index is terminal.
node_indices: list that stores the array of indices that are reached by the node. Node is indicated by the index of the list.
node_terms: array that stores the estimated value in a node.

To initialize the data structure with the root node we perform the following code.

```
need_splits<-c(TRUE)
is_terminal<-c(FALSE)
node_indices<-list(1:length(data_train[,1]))
node_terms<-c()
```

In order to learn a decsision tree in a while loop we perform the following things.

- Take the node indices that needs splits as split_nodes
- If split_nodes is of length 0 then we stop since it indicates all nodes are splitted and only leaf nodes are remained.
- Else in the nodes which are in need of splits we loop and perform following actions.
  - We take the **data_indices** of the node meaning data that are below the node.
  - We evaluate the value of node using **g** function.
  - We assert that this node does not splitting again.

Elif Küçük
0040851
COMP 421
HW 05 Report.

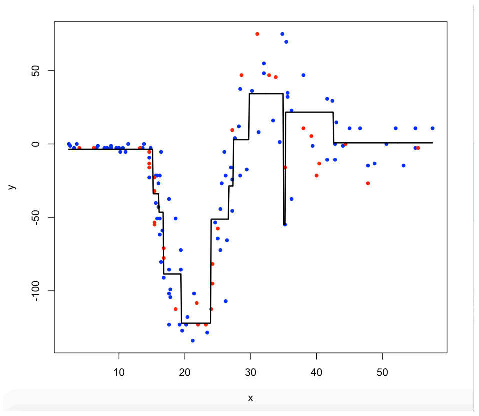
- Then we check if the node is terminal by checking if the remaining number of unique y is below P or remaining x points are all same. If so we set the terminal node as true and return. Else we do the following.
- We assert that node is not terminal and we find the split points.
- We find left and right indices and score the splits using E_m.
- We take minimum score and decide on the best_split and insert the next split values on the data structure.

Question 3

In order to find the decision tree on training data with P 10 and draw the plot I inserted the algorithm defined above in a function called decision_tree(P,data__) which returns predicted y for the data__.

Running the code below draws the plot.

```
data_interval <- seq(from = min(data_train[,1]), to = max(data_train[,1]), by = 0.1)
p_head<-decision_tree(10,data_interval)
plot(data_train[,1], data_train[,2], type = "p", pch = 20, col = "BLUE",
    ylab = "y", xlab = "x")
points(data_test[,1], data_test[,2], type = "p", pch = 20, col = "RED" )
lines(data_interval, p_head, type = "l", lwd = 2, col = "black")
```

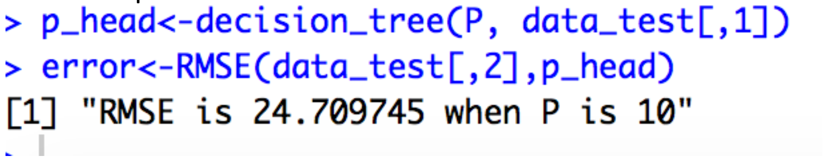


Question 4:

I defined I RMSE(y_truth,p_h) function which calculates mean squared error.

Running the code below we get the following error.
p_head<-decision_tree(P, data_test[,1])
error<-RMSE(data_test[,2],p_head)

Elif Küçük
0040851
COMP 421
HW 05 Report.

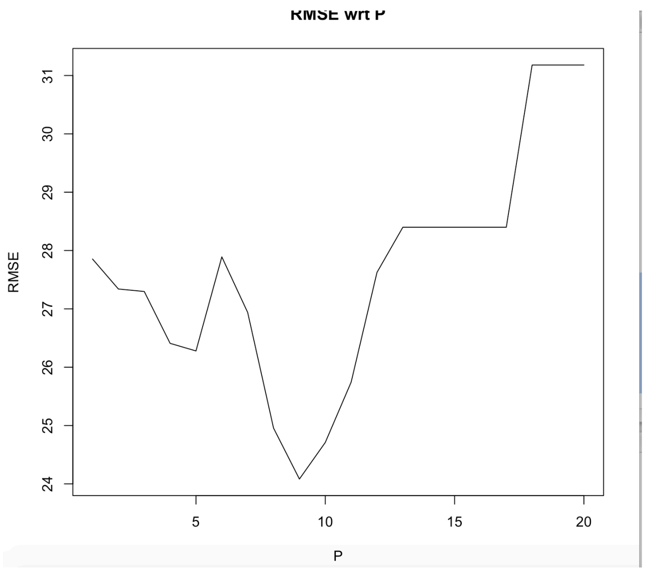


```
> p_head<-decision_tree(P, data_test[,1])
> error<-RMSE(data_test[,2],p_head)
[1] "RMSE is 24.709745 when P is 10"
```

Question 5:

Using the decision_tree(P,data__) test we run the function for P values in range 1:20.

```
Ps<-seq(from=1 , to=20, by=1)
error<-c()
for(p in Ps){
  P<-p
  p_head<-decision_tree(P, data_test[,1])
  error<-c(error, RMSE(data_test[,2],p_head))
}
plot(Ps,error , type="l", xlab="P", ylab="RMSE", main="RMSE wrt P")
```


RMSE wrt P
RMSE
P