Assignment 3-1

Amirali Khatib

6/3/2022

Question 1

Import libraries

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.3
```

```
library(lattice) # caret requirement
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.1.3
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.7     v purrr   0.3.4
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## Warning: package 'tibble' was built under R version 4.1.3
```

```
## Warning: package 'tidyr' was built under R version 4.1.3
```

```
## Warning: package 'forcats' was built under R version 4.1.3
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

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

```
## Warning: package 'rpart.plot' was built under R version 4.1.3
```

Read data

```
est_data = read.csv(file = 'D:\\Amirali\\University\\Applied Statistical Analysis\\R-tutorial\\Assignments\\Assig
nment 3\\hw03-problem1-estimation-data.csv', header = TRUE)
val_data = read.csv(file = 'D:\\Amirali\\University\\Applied Statistical Analysis\\R-tutorial\\Assignments\\Assig
nment 3\\hw03-problem1-validation-data.csv', header = TRUE)
tst_data = read.csv(file = 'D:\\Amirali\\University\\Applied Statistical Analysis\\R-tutorial\\Assignments\\Assig
nment 3\\hw03-problem1-test-data.csv', header = TRUE)
```

RMSE function

```
rmse = function(predicted, actual){sqrt(mean((actual - predicted)^2))}
```

Part A: Linear model

Linear model list

```
reg_model_list = list(
  mod_1 = lm(formula = y ~ x, data = est_data),
  mod_3 = lm(formula = y ~ poly(x,3), data = est_data),
  mod_5 = lm(formula = y ~ poly(x,5), data = est_data),
  mod_7 = lm(formula = y ~ poly(x,7), data = est_data),
  mod_9 = lm(formula = y ~ poly(x,9), data = est_data))
```

Predicted value

```
est_reg_predicted_list = lapply(reg_model_list, predict, est_data)
val_reg_predicted_list = lapply(reg_model_list, predict, val_data)
```

RMSE value

```
rmse_reg_est_list = sapply(est_reg_predicted_list, rmse, est_data$y)
rmse_reg_val_list = sapply(val_reg_predicted_list, rmse, val_data$y)
```

Plot line chart for RMSE value

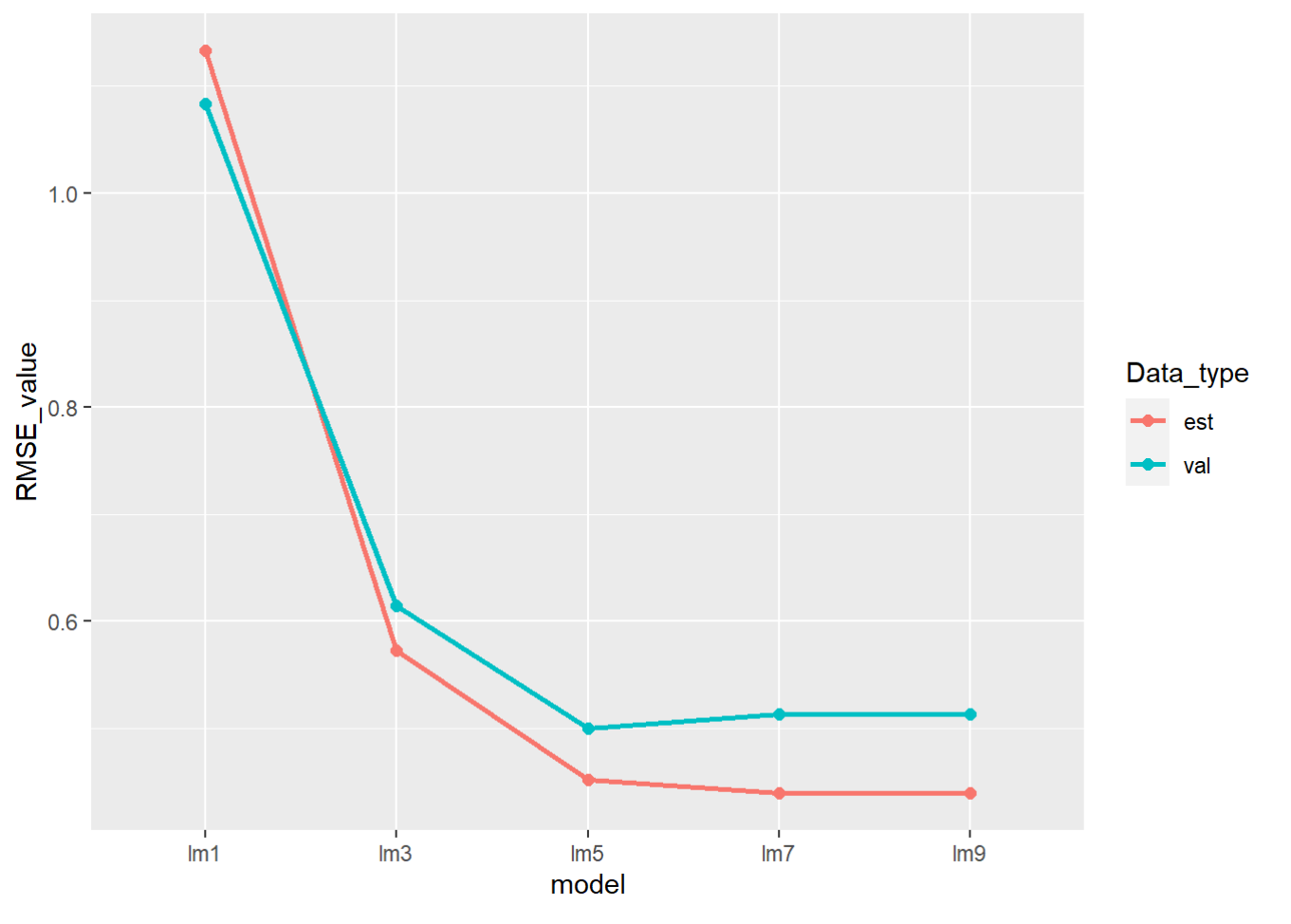
```
### data pre_processing for plot a multiple line chart
model_type = c('lm1','lm3','lm5','lm7','lm9')
model_type = c(model_type, model_type)
data_type = c(rep('est',5), rep('val',5))
rmse_reg_list = c(rmse_reg_est_list, rmse_reg_val_list)

RMSE_reg = data.frame(model = model_type,
                  Data_type = data_type,
                  RMSE_value = rmse_reg_list)
RMSE_reg
```

```
##    model Data_type RMSE_value
## 1    lm1       est  1.1330669
## 2    lm3       est  0.5729571
## 3    lm5       est  0.4518948
## 4    lm7       est  0.4394605
## 5    lm9       est  0.4393661
## 6    lm1       val  1.0831383
## 7    lm3       val  0.6147479
## 8    lm5       val  0.4997136
## 9    lm7       val  0.5133033
## 10   lm9       val  0.5135615
```

```
# plot

ggplot(data = RMSE_reg, aes(x = model, y = RMSE_value ,group = Data_type, colour = Data_type))+
  geom_line(size = 1) + geom_point(size = 2)
```



Considering chart above, `lm1 = y ~ x` , `lm3 = y ~ poly(x,3)` suffered from **under fitting** problem.suffered from **under fitting** problem.

the other models' RMSE value are close to each other. but `lm5 = y ~ poly(x,5)` fitted in a better way than the others.

We can say `lm7 = y ~ poly(x,7)` , `lm9 = y ~ poly(x,9)` suffered from **over fitting** problem.

Part B: KNN model

KNN model list

```
knn_model_list = list(
  model_k1 = knnreg(formula = y ~ x, data = est_data, k = 1),
  model_k11 = knnreg(formula = y ~ x, data = est_data, k = 11),
  model_k21 = knnreg(formula = y ~ x, data = est_data, k = 21),
  model_k31 = knnreg(formula = y ~ x, data = est_data, k = 31),
  model_k41 = knnreg(formula = y ~ x, data = est_data, k = 41))
```

Predicted value

```
est_knn_predicted_list = lapply(knn_model_list, predict, est_data)
val_knn_predicted_list = lapply(knn_model_list, predict, val_data)
```

RMSE value

```
rmse_knn_est_list = sapply(est_knn_predicted_list, rmse, est_data$y)
rmse_knn_val_list = sapply(val_knn_predicted_list, rmse, val_data$y)
```

Plot line chart for RMSE value

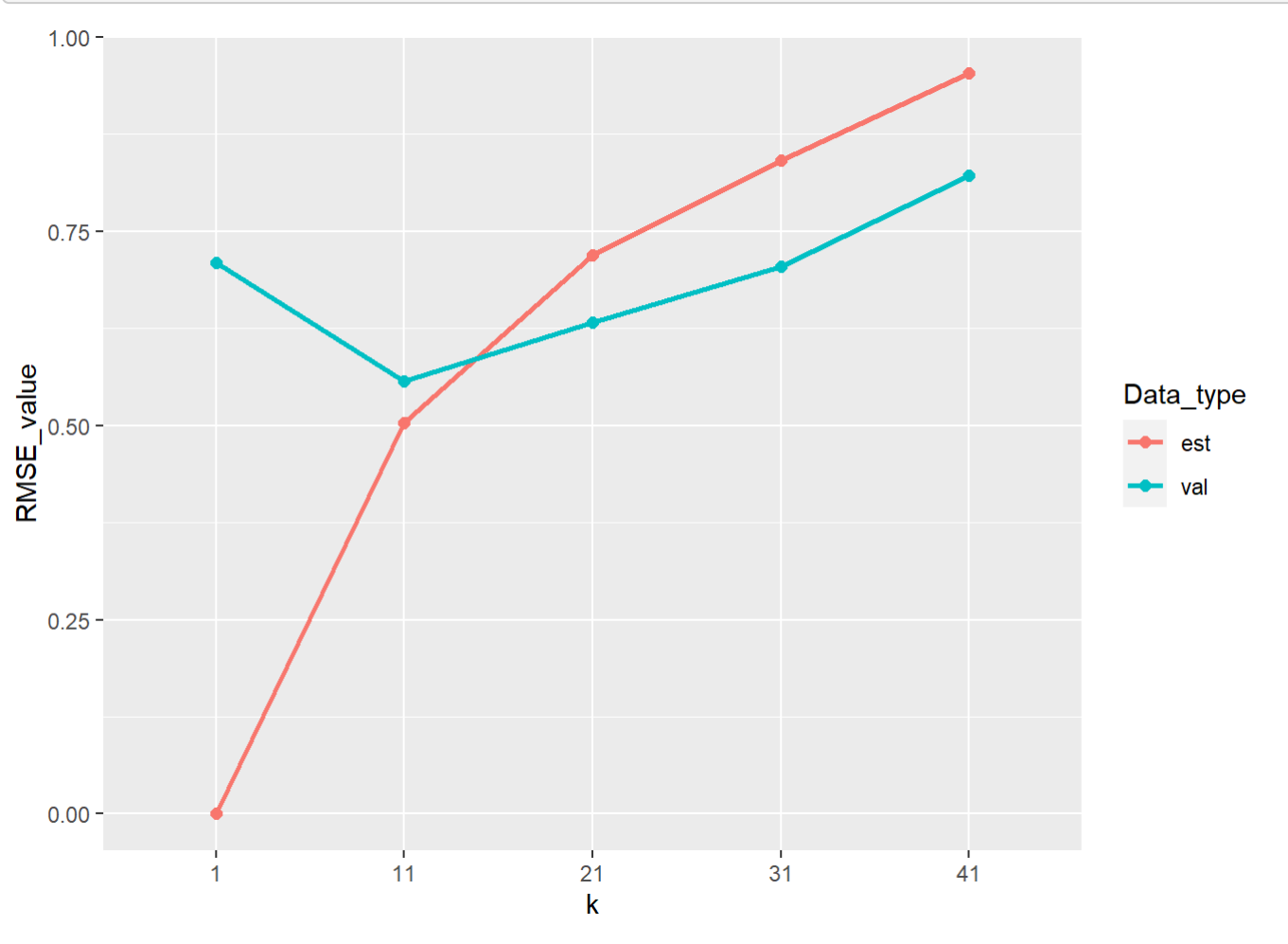
```
### data pre_processing for plot a multiple line chart
k_value = c('1','11','21','31','41')
k_value = c(k_value,k_value)
data_type = c(rep('est',5), rep('val',5))
rmse_knn_list = c(rmse_knn_est_list, rmse_knn_val_list)

RMSE_knn = data.frame(k = k_value,
                      Data_type = data_type,
                      RMSE_value = rmse_knn_list)

RMSE_knn
```

```
##     k Data_type RMSE_value
## 1   1       est  0.0000000
## 2  11       est  0.5035220
## 3  21       est  0.7188703
## 4  31       est  0.8412497
## 5  41       est  0.9534224
## 6   1       val  0.7098151
## 7  11       val  0.5569235
## 8  21       val  0.6322079
## 9  31       val  0.7051768
## 10 41       val  0.8222024
```

```
# plot
ggplot(data = RMSE_knn, aes(x = k, y = RMSE_value ,group = Data_type, colour = Data_type))+
  geom_line(size = 1)+geom_point(size = 2)
```



Considering chart above, `knn(k=1)` suffered from **over fitting** problem.

`knn(k=21)` , `knn(k=31)` , `knn(k = 41)` suffered from **under fitting** problem.

Part C: Regression tree model

Regression tree model list

```
regtr_model_list = list(
  model_cp1 = rpart(y ~ x, data = est_data, cp = 1, minsplit = 2),
  model_cp.1 = rpart(y ~ x, data = est_data, cp = 0.1, minsplit = 2),
  model_cp.01 = rpart(y ~ x, data = est_data, cp = 0.01, minsplit = 2),
  model_cp.001 = rpart(y ~ x, data = est_data, cp = 0.001, minsplit = 2),
  model_cp0 = rpart(y ~ x, data = est_data, cp = 0, minsplit = 2))
```

Predicted value

```
est_regtr_predicted_list = lapply(regtr_model_list, predict, est_data)
val_regtr_predicted_list = lapply(regtr_model_list, predict, val_data)
```

RMSE value

```
rmse_regtr_est_list = sapply(est_regtr_predicted_list, rmse, est_data$y)
rmse_regtr_val_list = sapply(val_regtr_predicted_list, rmse, val_data$y)
```

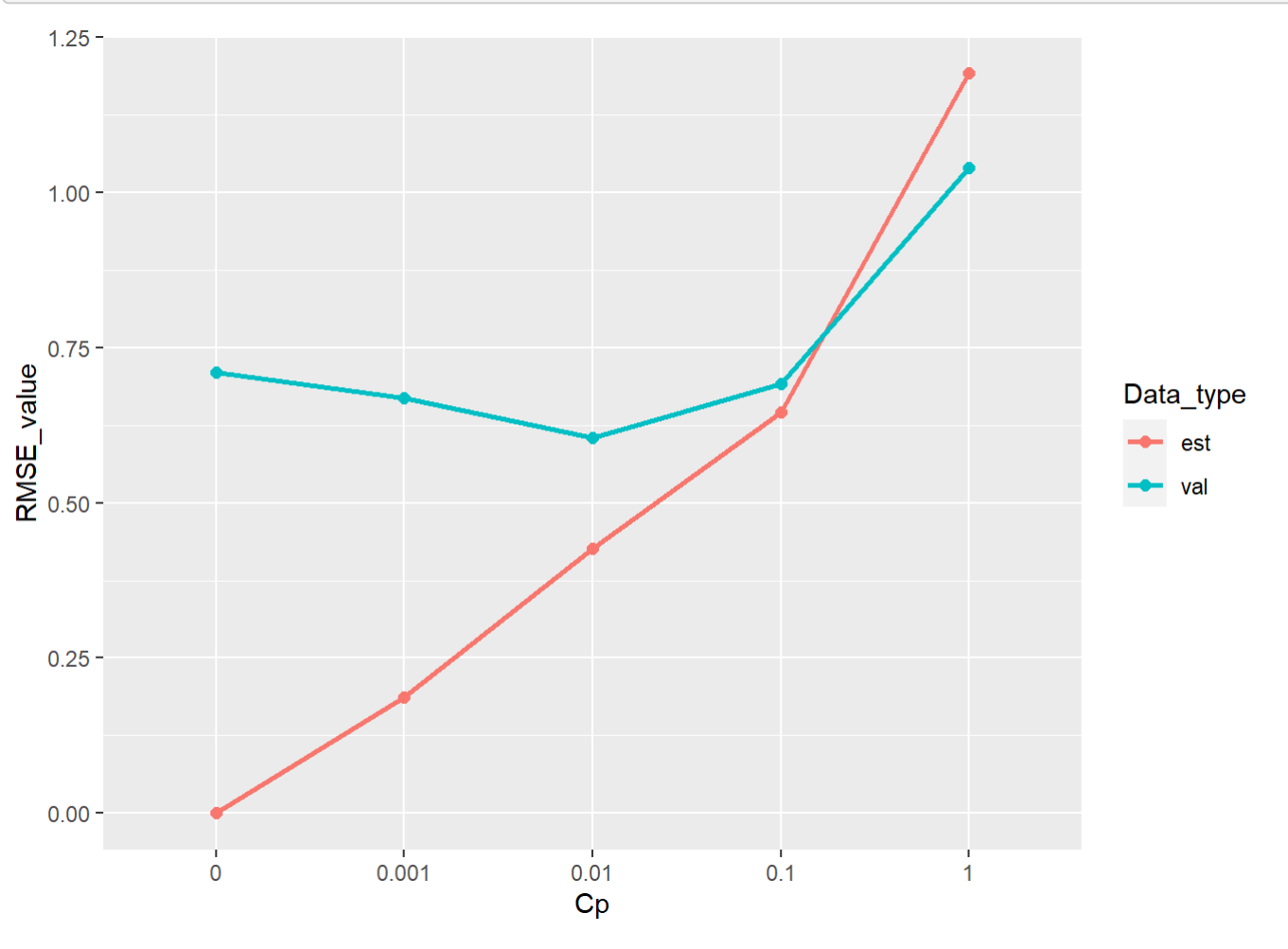
Plot line chart for RMSE value

```
### data pre_processing for plot a multiple line chart
cp_value = c('1','0.1','0.01','0.001','0')
cp_value = c(cp_value, cp_value)
data_type = c(rep('est',5), rep('val',5))
rmse_regtr_list = c(rmse_regtr_est_list, rmse_regtr_val_list)
RMSE_regtr = data.frame(Cp = cp_value,
                        Data_type = data_type,
                        RMSE_value = rmse_regtr_list)

RMSE_regtr
```

```
##       Cp Data_type RMSE_value
## 1      1       est  1.1915899
## 2    0.1       est  0.6457393
## 3   0.01       est  0.4254363
## 4  0.001       est  0.1865541
## 5      0       est  0.0000000
## 6      1       val  1.0395898
## 7    0.1       val  0.6909840
## 8   0.01       val  0.6048785
## 9  0.001       val  0.6688524
## 10     0       val  0.7098151
```

```
### line plot
ggplot(data = RMSE_regtr, aes(x = Cp, y = RMSE_value, group = Data_type, colour = Data_type))+
  geom_line(size = 1)+geom_point(size = 2)
```



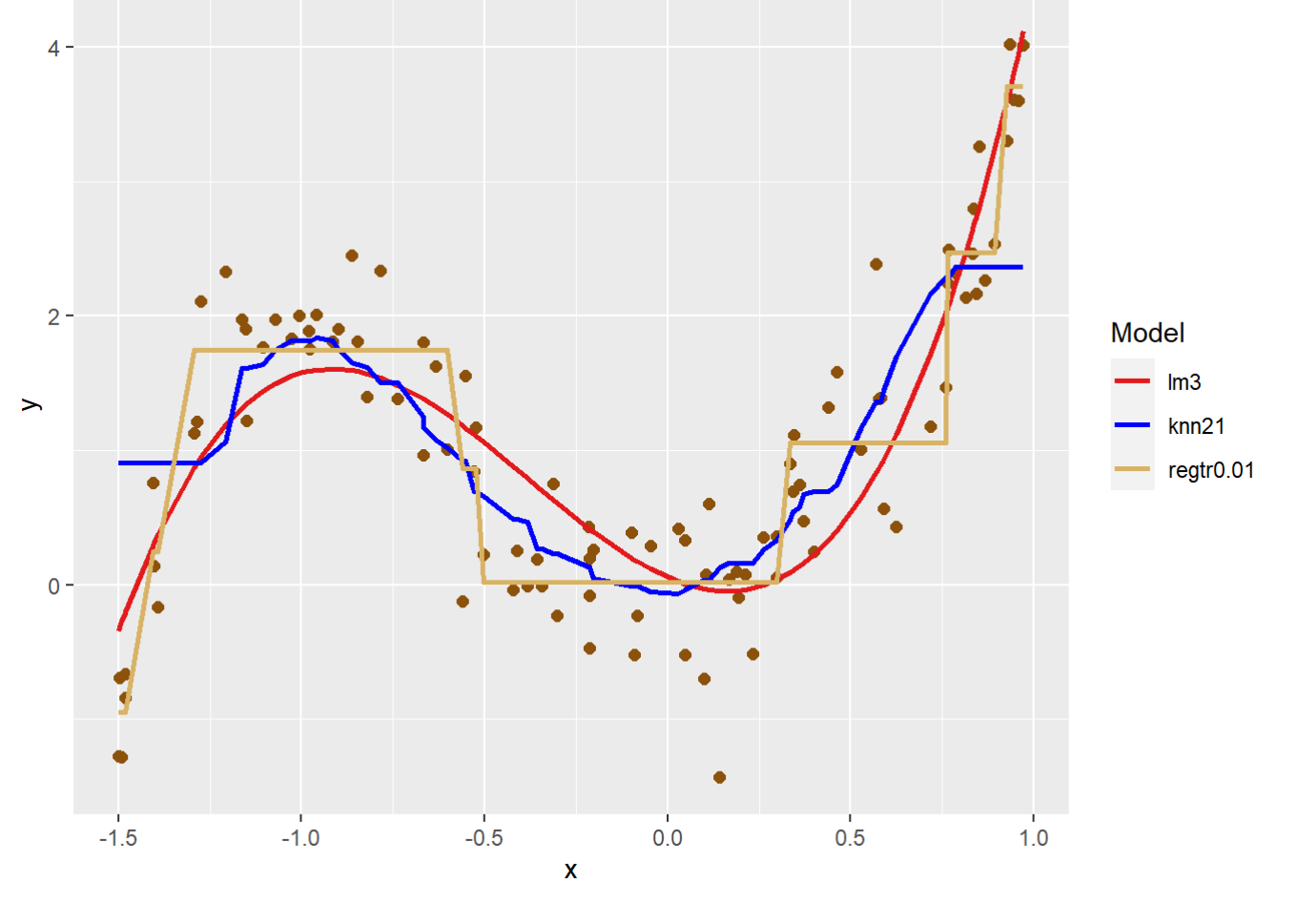
Considering chart above, `regtr(cp = 0)` , `regtr(cp = 0.001)` , suffered from **over fitting** problem.

`regtr(cp = 0.1)` , `regtr(cp = 1)` suffered from **under fitting** problem.

Part D: Scatter plot and 3 best fitted lines

```
df_plot = data.frame(x = est_data$x,
                     y = est_data$y,
                     lm5 = predict(reg_model_list$mod_3,est_data),
                     knn21 = predict(knn_model_list$model_k21,est_data),
                     regtr0.01 = predict(regtr_model_list$model_cp.01, est_data))


ggplot(data = df_plot, aes(x = x))+
  ## actual
  geom_point(aes(y = y), size = 2, color ="#8C510A")+
  ## linear model
  geom_line(aes(y = lm5, color = 'lm3'), size = 1)+
  ## knn model
  geom_line(aes(y = knn21, color = 'knn21'), size = 1)+
  ## regression tree model
  geom_line(aes(y = regtr0.01, color = 'regtr0.01'), size = 1)+
  ## add legend
  labs(x = 'x', y = 'y', color = 'Model')+
  scale_color_manual(values = c("lm3" = "#E31A1C", "knn21" = "blue",  "regtr0.01" = "#D8B365"))
```



Part E: Conclusion for final model

```
final_rmse_list = list(
  rmse_lm5 = rmse(predict(reg_model_list$mod_5, val_data), val_data$y),
  rmse_knn21 = rmse(predict(knn_model_list$model_k21, val_data), val_data$y),
  rmse_regtr = rmse(predict(regtr_model_list$model_cp.01,val_data), val_data$y))

final_rmse_list
```

```
## $rmse_lm5
## [1] 0.4997136
##
## $rmse_knn21
## [1] 0.6322079
##
## $rmse_regtr
## [1] 0.6048785
```

```
trn_data = rbind(est_data, val_data)
final_model = lm(formula = y ~ I(x ^ 5), data = trn_data)
rmse(predicted = predict(final_model,tst_data), tst_data$y)
```

```
## [1] 1.072325
```