Machine Learning

HW1

Xin Cheng
runnytone@uchicago.edu

01/20/2018

```
#List the packages we need, install if missing, then load all of them
PackageList =c('MASS','data.table','tree','kknn','rpart','rpart.plot')
NewPackages=PackageList[!(PackageList %in%
                            installed.packages()[,"Package"])]
if(length(NewPackages)) install.packages(NewPackages)

lapply(PackageList,require,character.only=TRUE)#array function

download.file("https://raw.githubusercontent.com/ChicagoBoothML/HelpR/master/docv.R", "docv.R")
source("docv.R") #this has docvknn used below

set.seed(2018) #Always set the seed for reproducibility
```

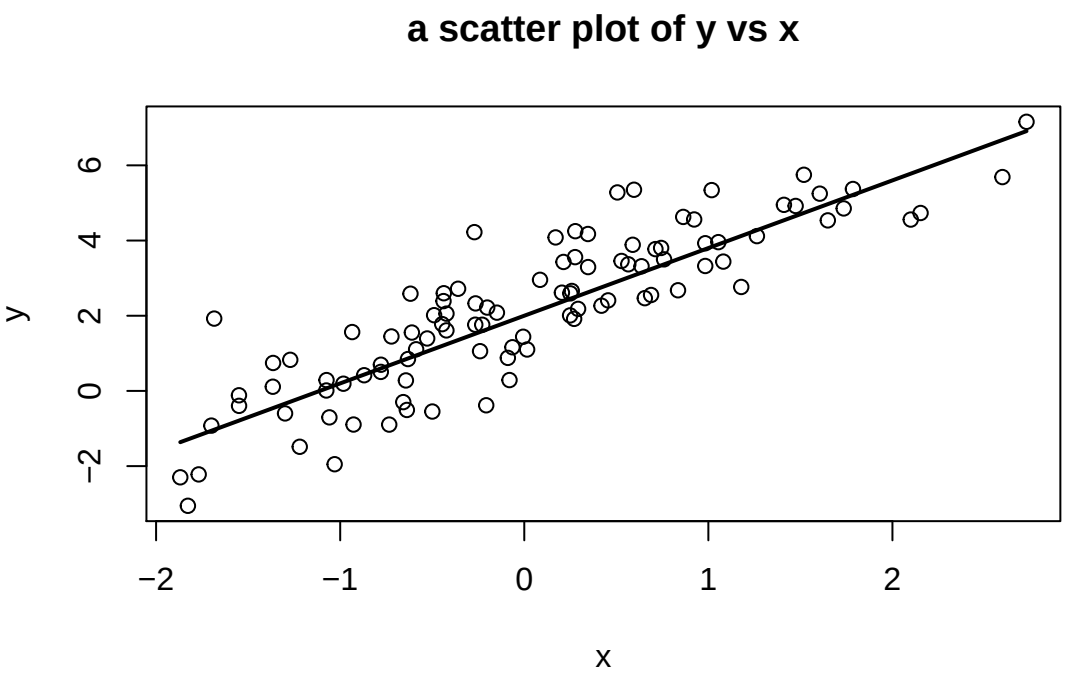
Q1

Q1.1

```
set.seed(2018)
x <- rnorm(100, mean=0, sd=1)
varepsilon <- rnorm(100, mean=0, sd=1)
y <- 1.8*x + 2 + varepsilon
train <- data.frame(y,x)
train <- train[order(train$x),]

x <- rnorm(10000, mean=0, sd=1)
varepsilon <- rnorm(10000, mean=0, sd=1)
y <- 1.8*x + 2 + varepsilon
test <- data.frame(y, x)
test <- test[order(test$x),]
rm(x,y)
```

Q1.2

```
plot(train$x, train$y, main = "a scatter plot of y vs x", xlab = "x", ylab = "y")+
lines(train$x, 1.8*train$x + 2, col="black", lwd = 2)
```



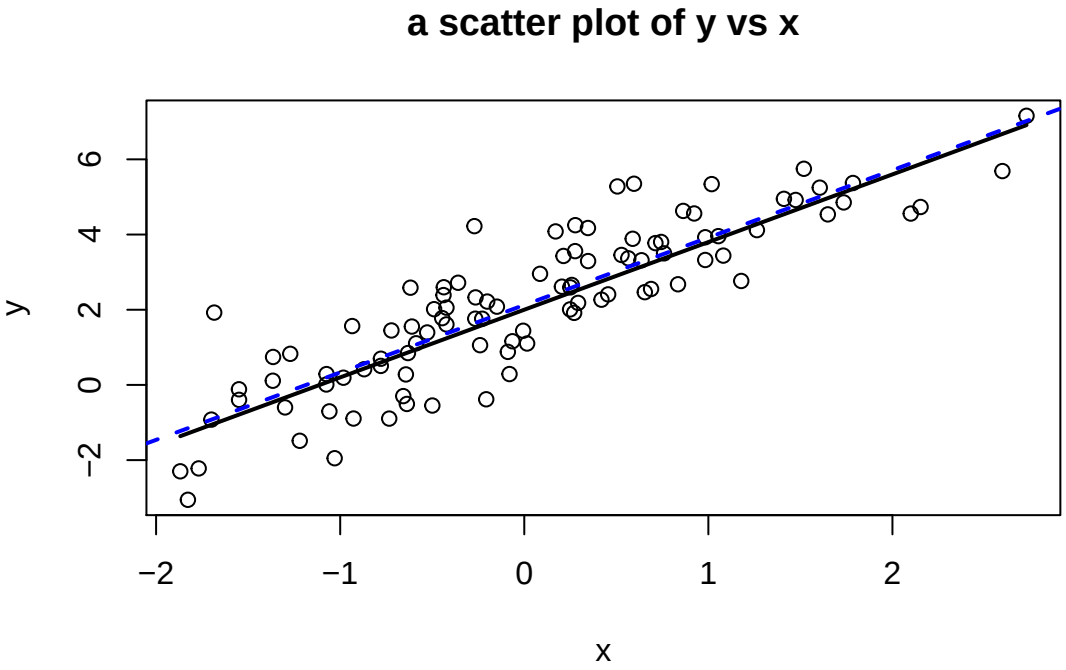

```
## integer(0)
```

Q1.3

```
ls <- lm(train$y~train$x, train)
print(summary(ls))
```

```
##
## Call:
## lm(formula = train$y ~ train$x, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2277 -0.6785  0.0435  0.6321  2.8197
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1254     0.1024   20.76   <2e-16 ***
## train$x       1.7936     0.1015   17.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.023 on 98 degrees of freedom
## Multiple R-squared:  0.7612, Adjusted R-squared:  0.7588
## F-statistic: 312.4 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
plot(train$x, train$y, main = "a scatter plot of y vs x", xlab = "x", ylab = "y")+
lines(train$x, 1.8*train$x + 2, col="black", lwd = 2)+
abline(ls$coef, col="blue", lwd = 2, lty = "dashed")
```



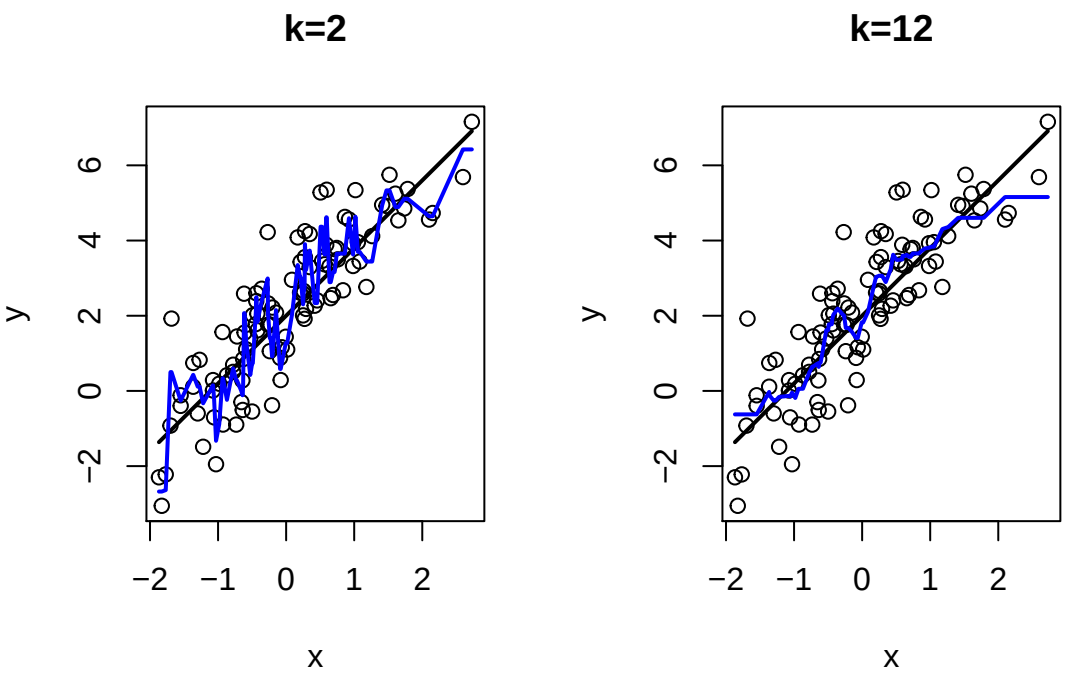

```
## integer(0)
```

Q1.4

```
library(kknn)
kvec <- c(2:15)
kknn.train <- list()
fitted.train <- data.frame(matrix(NA, nrow = 100, ncol = length(kvec)))

for (i in 1:length(kvec)) {
  kknn.train[[i]] <- kknn(y~x, train, train, k = kvec[i], kernel = "rectangular")
  fitted.train[, i] <- kknn.train[[i]]$fitted
  }

par(mfrow=c(1,2))

plot(train$x, train$y, main = "k=2", xlab = "x", ylab = "y")+
lines(train$x, 1.8*train$x + 2, col="black", lwd = 2)+
lines(train$x, fitted.train[, 1], col="blue", lwd = 2)
```

```
## integer(0)
```

```
plot(train$x, train$y, main = "k=12", xlab = "x", ylab = "y")+
lines(train$x, 1.8*train$x + 2, col="black", lwd = 2)+
lines(train$x, fitted.train[, 11], col="blue", lwd = 2)
```



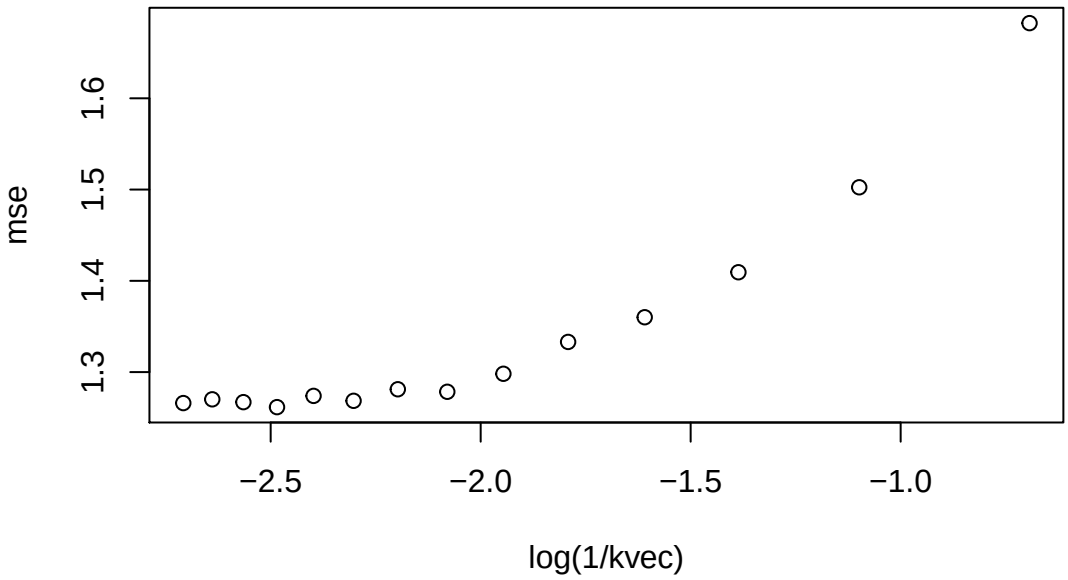

```
## integer(0)
```

Q1.5

```
kvec <- c(2:15)
kknn.test <- list()
fitted.test <- data.frame(matrix(NA, nrow = 10000, ncol = length(kvec)))
mse <- as.numeric()

for(i in 1:length(kvec)) {
  kknn.test[[i]] <- kknn(y~x, train, test, k = kvec[i], kernel = "rectangular")
  fitted.test[, i] <- kknn.test[[i]]$fitted
  mse[i] = mean((test$y-fitted.test[,i])^2)
}
cat("the best k is: ", kvec[which.min(mse)])
```

```
## the best k is:  12
```

```
mse.ls <- mean((test$y-ls$coefficients[1]-ls$coefficients[2]*test$x)^2)

plot(log(1/kvec), mse)+
abline(h = mse.ls, lwd=2, col = "lightgray", lty = 3)
```




```
## integer(0)
```

```
cat("the smallest MSE of knn is", min(mse), ",while the MES of linear gression is", mse.ls, ". \n Linea
```

```
## the smallest MSE of knn is 1.261612 ,while the MES of linear gression is 1.0248 .
##  Linear gression model fits better in this case.
```

Note

kknn(y~x, train, train, k = k, kernel = "rectangular")$fitted gives you prediction of y in train dataset

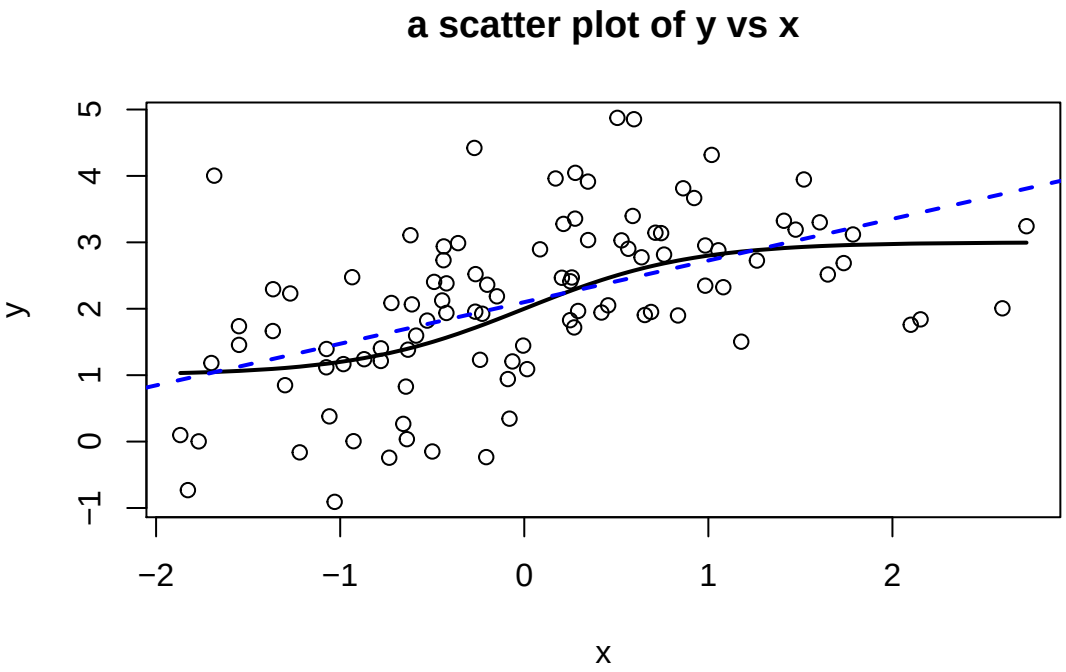
kknn(y~x, train, test, k = k, kernel = "rectangular")$fitted gives you prediction of y in test dataset

The solution should consider naming the columns of the two variables the same and be agnostic to who you are calling in the formula attribute. Otherwise the program will consider that the only information that is reliable is the one in the formula and will only use it and predict a database with its own data.

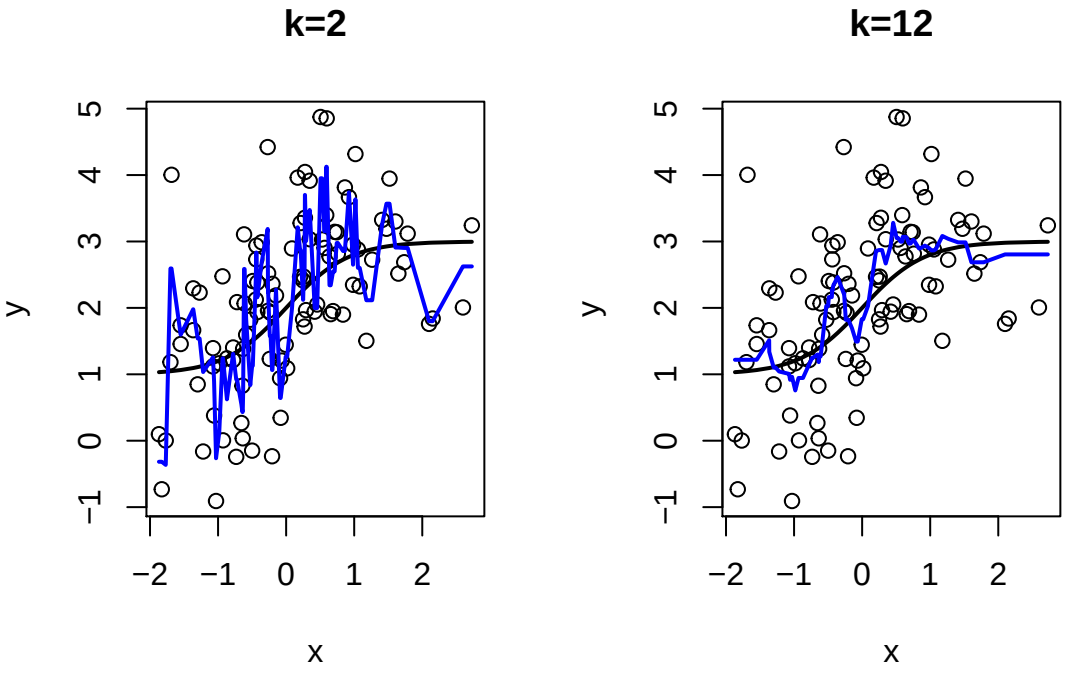
Q1.6

```
##
## Call:
## lm(formula = train$y ~ train$x, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.36223 -0.60754  0.07714  0.60121  2.95953
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.0999     0.1070   19.63  < 2e-16 ***
## train$x       0.6266     0.1060    5.91 4.98e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.069 on 98 degrees of freedom
## Multiple R-squared:  0.2628, Adjusted R-squared:  0.2552
```

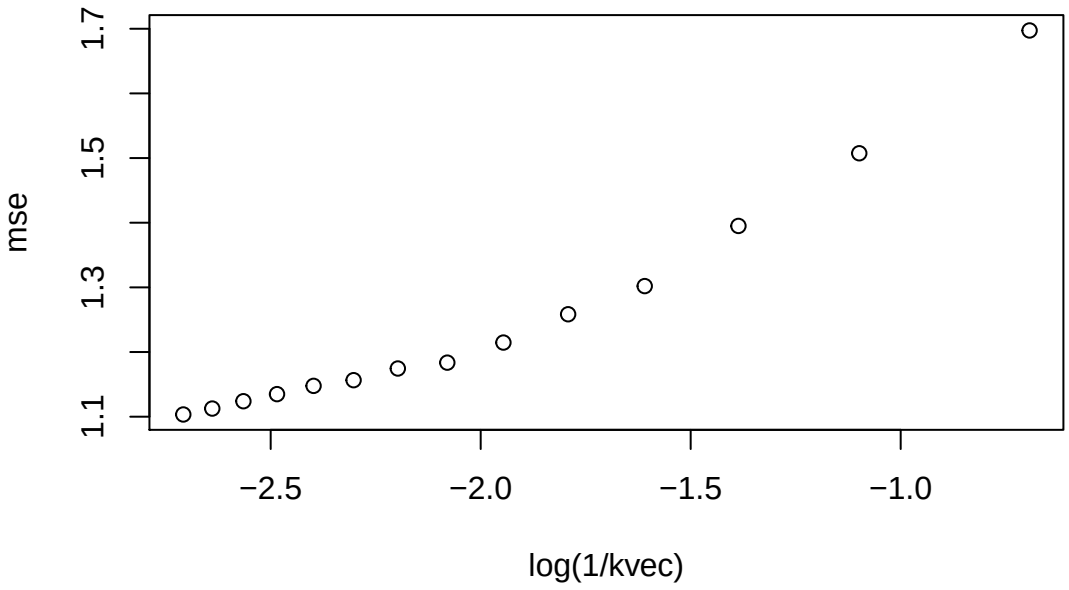
```
## F-statistic: 34.93 on 1 and 98 DF,  p-value: 4.979e-08
```




```
## integer(0)
## integer(0)
```




```
## integer(0)
## the best k is:  15
```

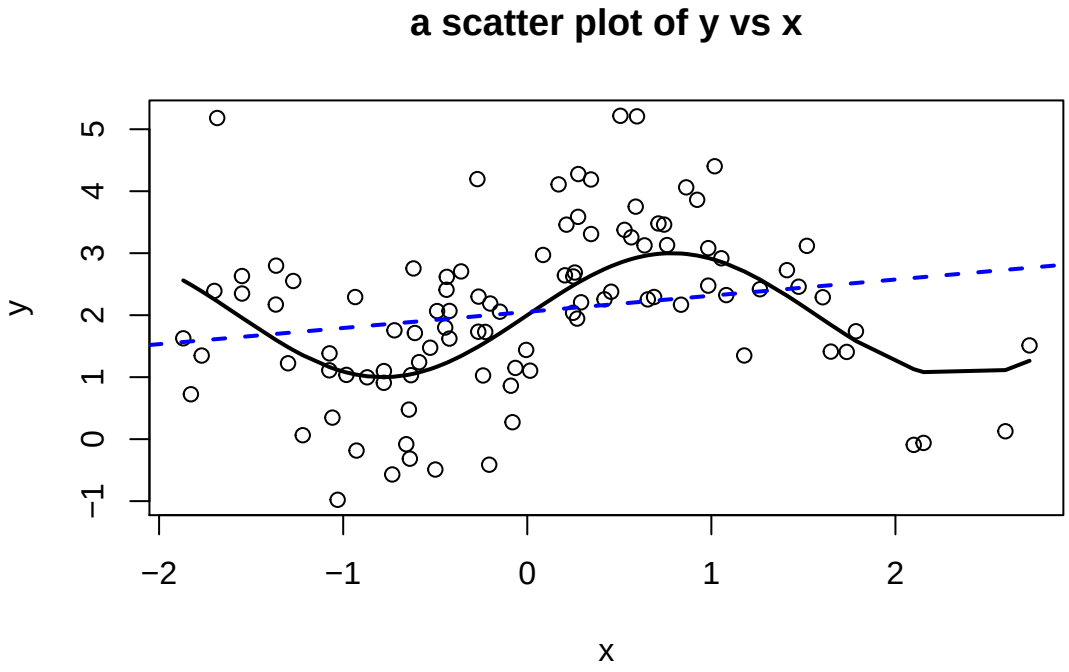



```
## integer(0)
```

```
## the smallest MSE of knn is 1.10342 ,while the MES of linear gression is 1.058055 .
##  linear gression model fits better in this case.
```

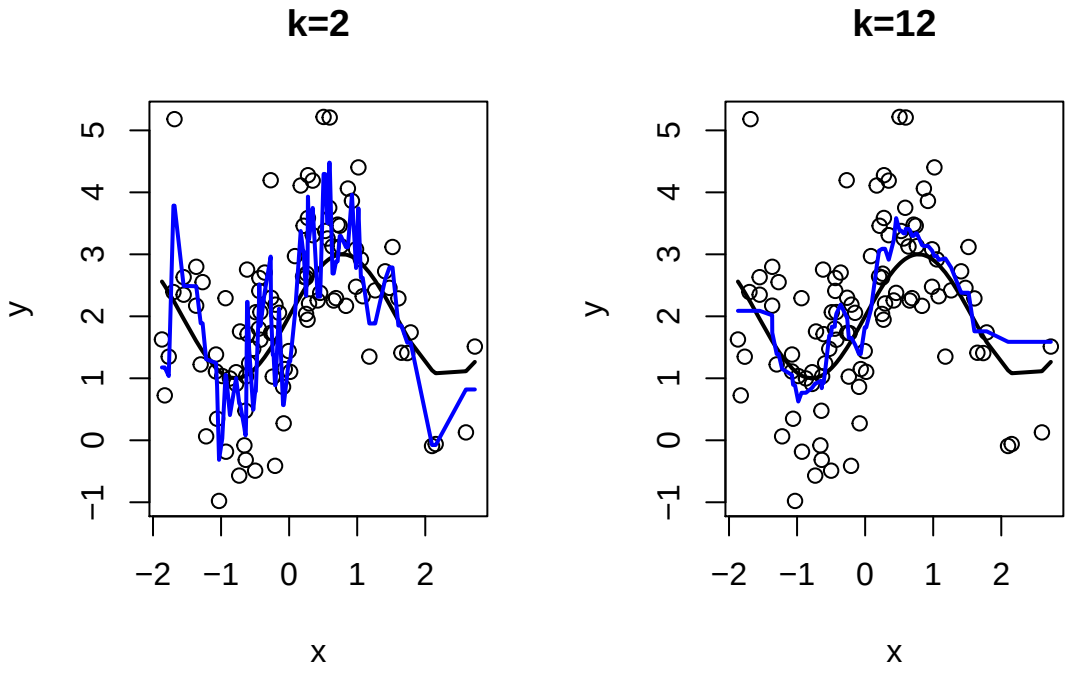
Q1.7

```
##
## Call:
## lm(formula = train$y ~ train$x, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7639 -0.8357  0.0382  0.7927  3.5662
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.0539     0.1311  15.670   <2e-16 ***
## train$x       0.2606     0.1299   2.006   0.0477 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.31 on 98 degrees of freedom
## Multiple R-squared:  0.03943,    Adjusted R-squared:  0.02963
## F-statistic: 4.023 on 1 and 98 DF,  p-value: 0.04765
```



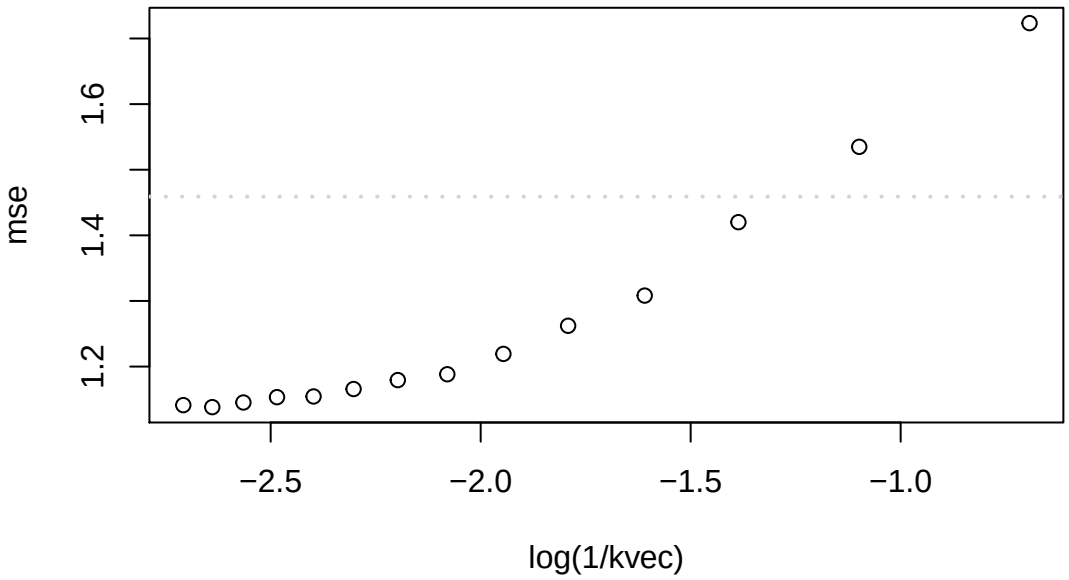

```
## integer(0)
```

```
## integer(0)
```




```
## integer(0)
```

```
## the best k is:  14
```

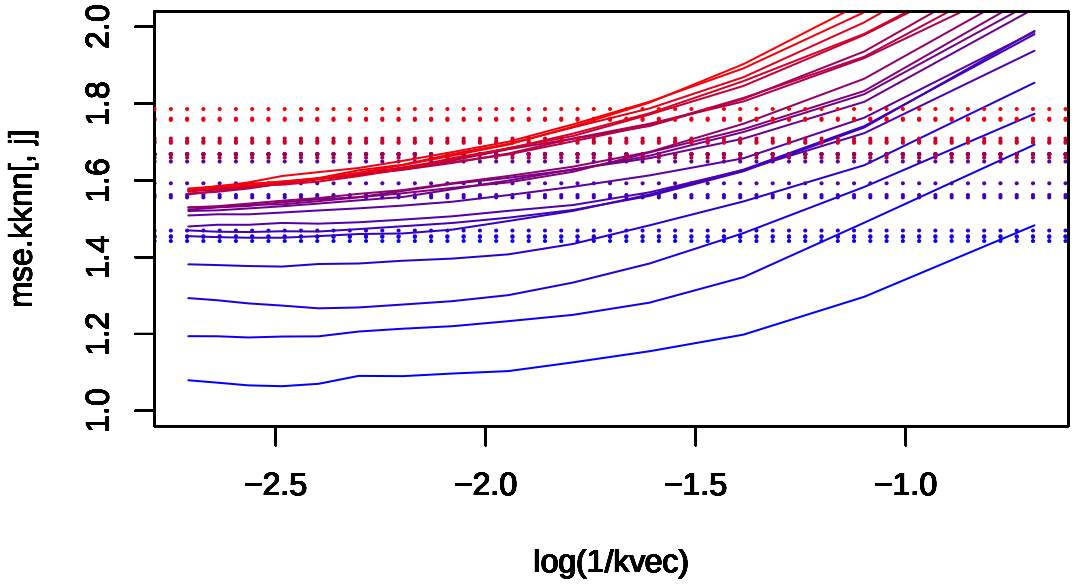



```
## integer(0)
```

```
## the smallest MSE of knn is 1.138171 ,while the MES of linear gression is 1.459013 .
##  knn model fits better in this case.
```

Q1.8

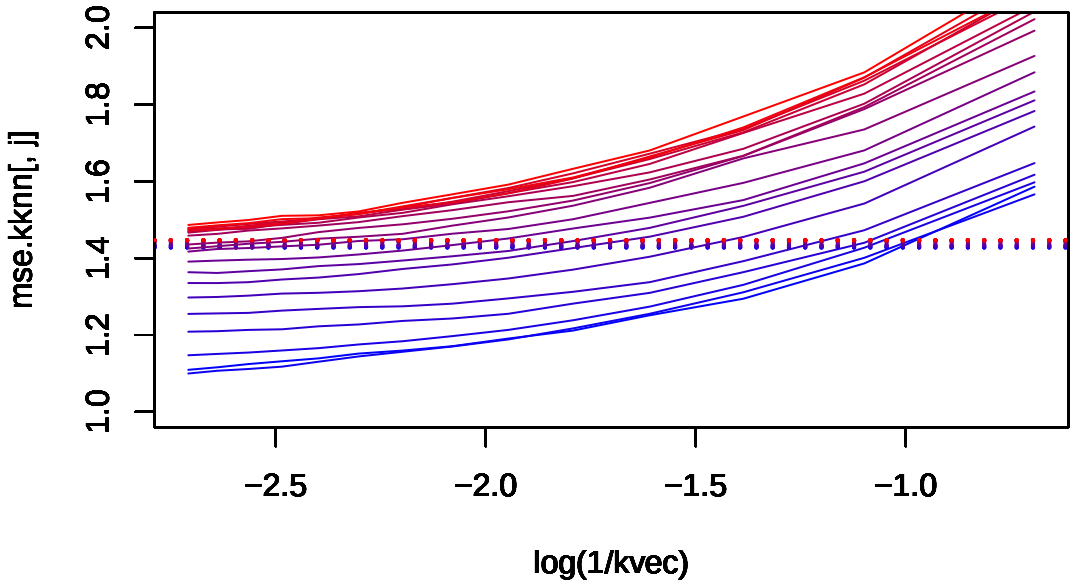
```
## Warning in par(new = TRUE): calling par(new=TRUE) with no plot
```




```
## as number of variables increase, mse for both models increase.
##  on average, knn models with large k is better than linear model
```

Q1.9

```
## Warning in par(new = TRUE): calling par(new=TRUE) with no plot
```




```
## for large train dataset linear model fits better when No. of variables increase;
##  or best k for knn model increase
```

Machine Learning

HW1

Xin Cheng
runnytone@uchicago.edu

01/20/2018

```
knitr::opts_chunk$set(echo = TRUE)
```

```
#List the packages we need, install if missing, then load all of them
PackageList =c('MASS','data.table','tree','kknn','rpart','rpart.plot')
NewPackages=PackageList[!(PackageList %in%
                            installed.packages()[,"Package"])]
if(length(NewPackages)) install.packages(NewPackages)

lapply(PackageList,require,character.only=TRUE)#array function
```

```
## Loading required package: MASS
## Warning: package 'MASS' was built under R version 3.4.3
## Loading required package: data.table
## Loading required package: tree
## Loading required package: kknn
## Loading required package: rpart
## Loading required package: rpart.plot
## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] TRUE
##
## [[5]]
## [1] TRUE
##
## [[6]]
## [1] TRUE
```

```
download.file("https://raw.githubusercontent.com/ChicagoBoothML/HelpR/master/docv.R", "docv.R")
source("docv.R") #this has docvknn used below

set.seed(2018) #Always set the seed for reproducibility
```

Q2

Q2.1

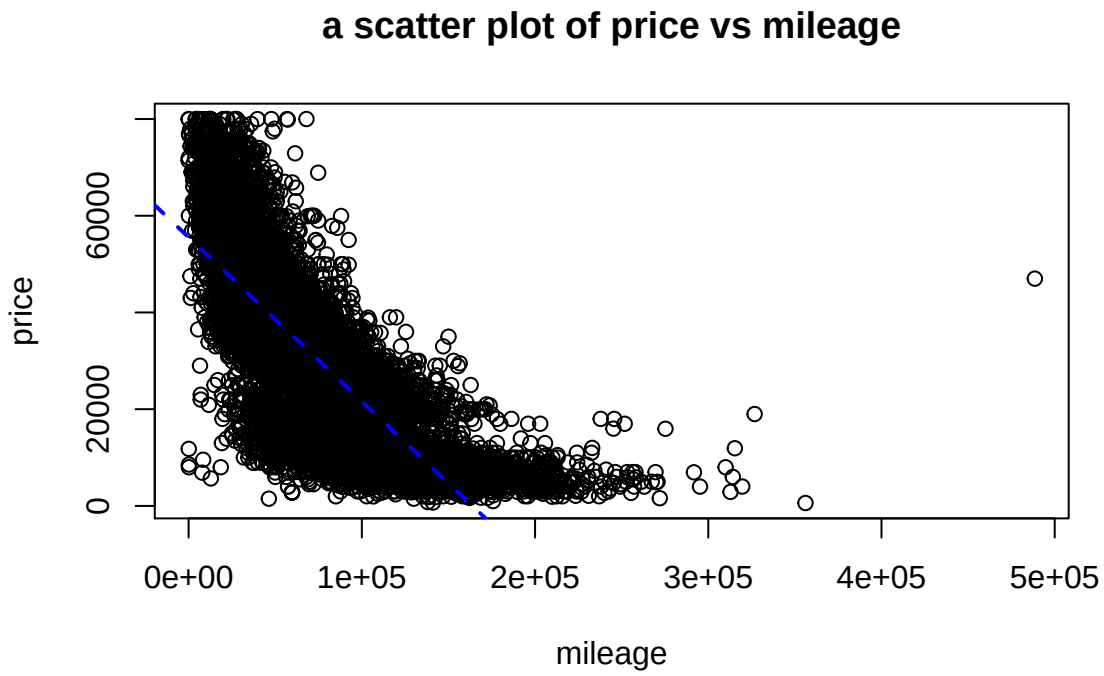
```
raw.data <- read.csv(url("https://github.com/ChicagoBoothML/MLClassData/raw/master/UsedCars/UsedCars.csv
#View(raw.data)
```

Q2.2

```
nrow.data <- nrow(raw.data)
train_indices = sample(nrow.data, size = nrow.data * 0.75, replace = FALSE)
data.train <- raw.data[train_indices,]
data.test <- raw.data[-train_indices,]
```

Q2.3

```
ls <- lm(price ~ mileage, data.train)

plot(data.train$mileage, data.train$price, main = "a scatter plot of price vs mileage", xlab = "mileage
abline(ls$coef, col="blue", lwd = 2, lty = "dashed")
```




```
## integer(0)
```

Q2.4

Q2.5.a : knn

Q2.5.b : regression tree

```
set.seed(2018) #always set the seed!
data.train <- raw.data[,c(1,4)]

tree <- rpart(price~.,      #Formula
              data = data.train ,#Data
              control = rpart.control(minsplit=5,#the minimum number of observations that must exist in
              cp=0.0001, #complexity, the lower, the larger the tree is
              xval=10   #number of cross validations
              ))

nbig <- length(unique(tree$where))
cat('size of big tree: ',nbig,'\n')
```

```
## size of big tree:  85
```

```
(cptable <- printcp(tree))
```

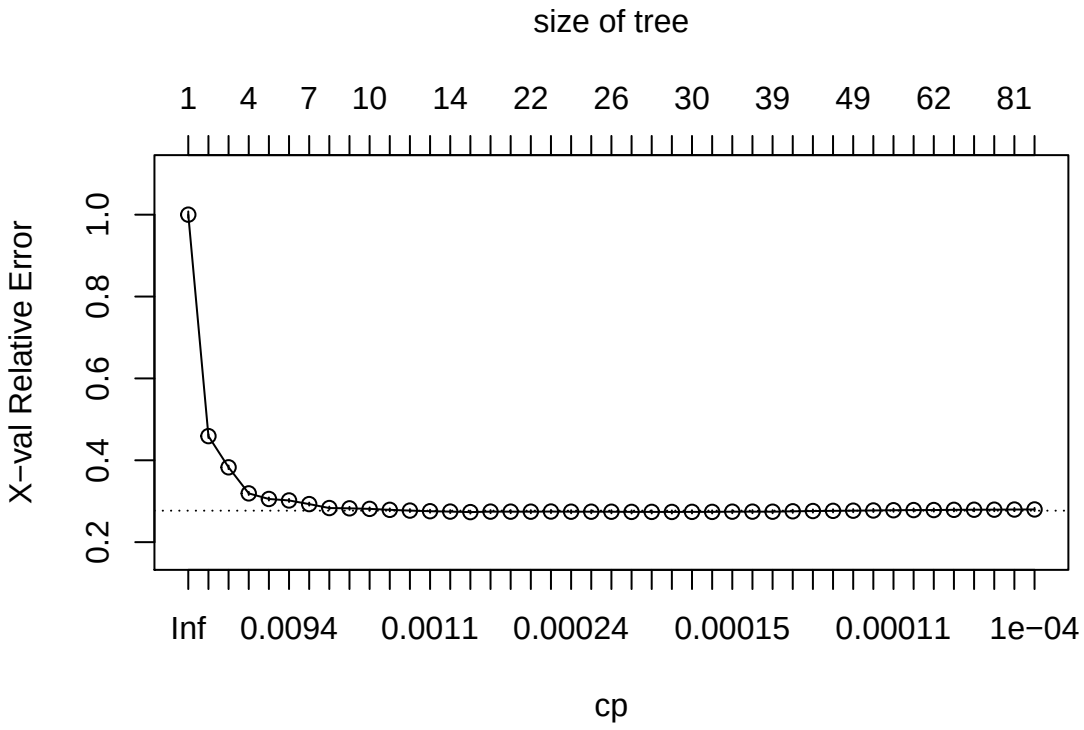
```
##
## Regression tree:
## rpart(formula = price ~ ., data = data.train, control = rpart.control(minsplit = 5,
##     cp = 1e-04, xval = 10))
##
## Variables actually used in tree construction:
## [1] mileage
##
## Root node error: 6.712e+12/20063 = 334547097
##
## n= 20063
##
##            CP nsplit rel error  xerror      xstd
## 1  0.54186822      0   1.00000 1.00008 0.0078953
## 2  0.07677819      1   0.45813 0.45902 0.0045779
## 3  0.06568471      2   0.38135 0.38281 0.0043563
## 4  0.00983547      3   0.31567 0.31913 0.0037764
## 5  0.00947080      4   0.30583 0.30547 0.0036970
## 6  0.00941710      5   0.29636 0.30187 0.0036853
## 7  0.00735430      6   0.28695 0.29292 0.0036450
## 8  0.00209090      7   0.27959 0.28331 0.0035773
## 9  0.00188021      8   0.27750 0.28255 0.0035877
## 10 0.00173232      9   0.27562 0.28130 0.0035885
## 11 0.00140958     10   0.27389 0.27892 0.0035688
## 12 0.00115084     11   0.27248 0.27713 0.0035714
## 13 0.00096808     12   0.27133 0.27557 0.0035752
## 14 0.00087708     13   0.27036 0.27463 0.0035682
## 15 0.00032198     14   0.26948 0.27337 0.0035633
## 16 0.00030034     17   0.26852 0.27458 0.0036942
## 17 0.00028392     20   0.26762 0.27451 0.0036925
## 18 0.00026599     21   0.26733 0.27445 0.0036951
```

```
## 19 0.00024480     22   0.26707 0.27474 0.0037169
## 20 0.00022635     23   0.26682 0.27438 0.0037110
## 21 0.00022346     24   0.26659 0.27435 0.0037101
## 22 0.00021454     25   0.26637 0.27428 0.0037103
## 23 0.00020968     26   0.26616 0.27391 0.0037059
## 24 0.00018895     27   0.26595 0.27394 0.0037055
## 25 0.00016781     28   0.26576 0.27379 0.0037054
## 26 0.00016751     29   0.26559 0.27399 0.0037108
## 27 0.00015261     31   0.26525 0.27394 0.0037093
## 28 0.00014746     32   0.26510 0.27446 0.0037236
## 29 0.00014251     33   0.26495 0.27453 0.0037239
## 30 0.00013903     38   0.26424 0.27440 0.0037229
## 31 0.00012635     43   0.26355 0.27537 0.0037358
## 32 0.00012392     45   0.26329 0.27602 0.0037466
## 33 0.00011880     47   0.26305 0.27648 0.0037533
## 34 0.00011856     48   0.26293 0.27687 0.0037637
## 35 0.00011630     51   0.26257 0.27736 0.0037696
## 36 0.00011270     53   0.26234 0.27795 0.0037769
## 37 0.00011075     54   0.26223 0.27832 0.0037803
## 38 0.00010885     61   0.26143 0.27844 0.0037803
## 39 0.00010441     65   0.26099 0.27894 0.0037833
## 40 0.00010423     78   0.25961 0.27921 0.0037868
## 41 0.00010401     79   0.25950 0.27936 0.0037861
## 42 0.00010249     80   0.25940 0.27965 0.0037904
## 43 0.00010000     84   0.25899 0.27987 0.0037923
```

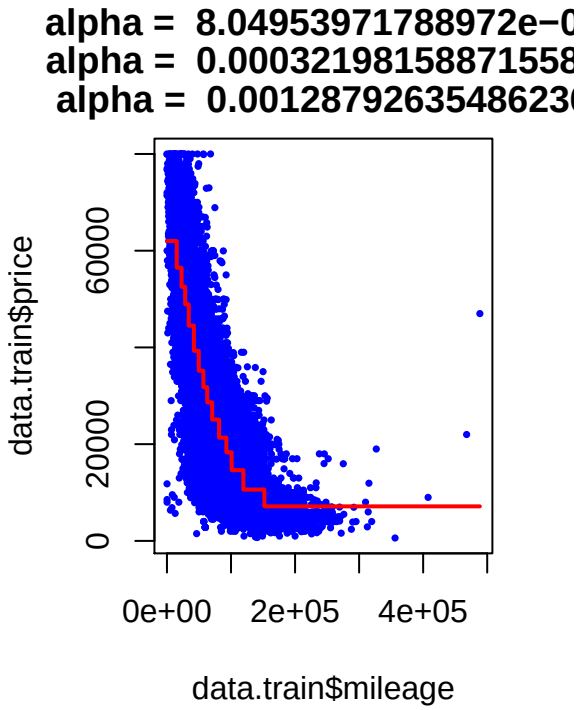
```
##              CP nsplit rel error    xerror        xstd
## 1  0.5418682228      0 1.0000000 1.0000819 0.007895339
## 2  0.0767781855      1 0.4581318 0.4590190 0.004577853
## 3  0.0656847090      2 0.3813536 0.3828089 0.004356304
## 4  0.0098354692      3 0.3156689 0.3191254 0.003776441
## 5  0.0094707964      4 0.3058334 0.3054688 0.003697001
## 6  0.0094170974      5 0.2963626 0.3018720 0.003685347
## 7  0.0073543046      6 0.2869455 0.2929190 0.003644999
## 8  0.0020909046      7 0.2795912 0.2833136 0.003577266
## 9  0.0018802103      8 0.2775003 0.2825547 0.003587663
## 10 0.0017323164      9 0.2756201 0.2812966 0.003588548
## 11 0.0014095826     10 0.2738878 0.2789154 0.003568772
## 12 0.0011508368     11 0.2724782 0.2771289 0.003571410
## 13 0.0009680830     12 0.2713274 0.2755652 0.003575245
## 14 0.0008770769     13 0.2703593 0.2746297 0.003568197
## 15 0.0003219816     14 0.2694822 0.2733736 0.003563267
## 16 0.0003003441     17 0.2685163 0.2745822 0.003694200
## 17 0.0002839200     20 0.2676152 0.2745092 0.003692538
## 18 0.0002659865     21 0.2673313 0.2744485 0.003695144
## 19 0.0002447951     22 0.2670653 0.2747394 0.003716905
## 20 0.0002263458     23 0.2668205 0.2743849 0.003711014
## 21 0.0002234602     24 0.2665942 0.2743478 0.003710136
## 22 0.0002145388     25 0.2663707 0.2742837 0.003710267
## 23 0.0002096818     26 0.2661562 0.2739135 0.003705925
## 24 0.0001889473     27 0.2659465 0.2739431 0.003705473
## 25 0.0001678106     28 0.2657576 0.2737929 0.003705380
## 26 0.0001675141     29 0.2655897 0.2739866 0.003710778
## 27 0.0001526129     31 0.2652547 0.2739406 0.003709286
```

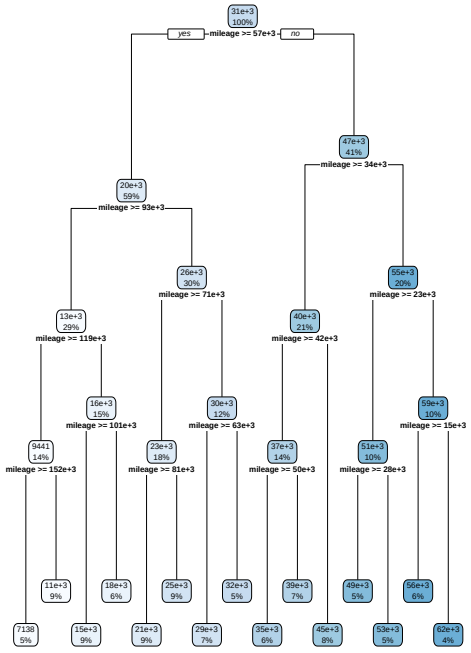
```
## 28 0.0001474566     32 0.2651021 0.2744604 0.003723598
## 29 0.0001425104     33 0.2649546 0.2745307 0.003723932
## 30 0.0001390333     38 0.2642421 0.2744019 0.003722905
## 31 0.0001263504     43 0.2635469 0.2753655 0.003735803
## 32 0.0001239232     45 0.2632942 0.2760202 0.003746605
## 33 0.0001188043     47 0.2630464 0.2764819 0.003753288
## 34 0.0001185621     48 0.2629276 0.2768690 0.003763734
## 35 0.0001162952     51 0.2625719 0.2773583 0.003769578
## 36 0.0001126995     53 0.2623393 0.2779526 0.003776853
## 37 0.0001107476     54 0.2622266 0.2783219 0.003780304
## 38 0.0001088455     61 0.2614300 0.2784408 0.003780344
## 39 0.0001044124     65 0.2609946 0.2789380 0.003783267
## 40 0.0001042259     78 0.2596063 0.2792089 0.003786761
## 41 0.0001040144     79 0.2595021 0.2793586 0.003786077
## 42 0.0001024874     80 0.2593981 0.2796464 0.003790424
## 43 0.0001000000     84 0.2589881 0.2798715 0.003792258
```

```
bestcp <- cptable[ which.min(cptable[,"xerror"]), "CP" ] # this is the optimal cp parameter

plotcp(tree) # plot results
```




```
# show fit from some trees
oo = order(data.train$mileage)
cpvec = c(bestcp / 4, bestcp,bestcp*4)

par(mfrow=c(1,2))
plot(data.train$mileage, data.train$price, pch=16, col='blue', cex=.5)
ptree = prune(tree, cp=bestcp)
pfit = predict(ptree)
lines(data.train$mileage[oo],pfit[oo],col='red',lwd=2)
title(paste('alpha = ',cpvec))
rpart.plot(ptree)
```






```
cat('RMSE is ',sqrt(mean((data.train$price -predict(tree, data.train))^2)),'\n')
```

```
## RMSE is  9308.261
```

Q2.6.A

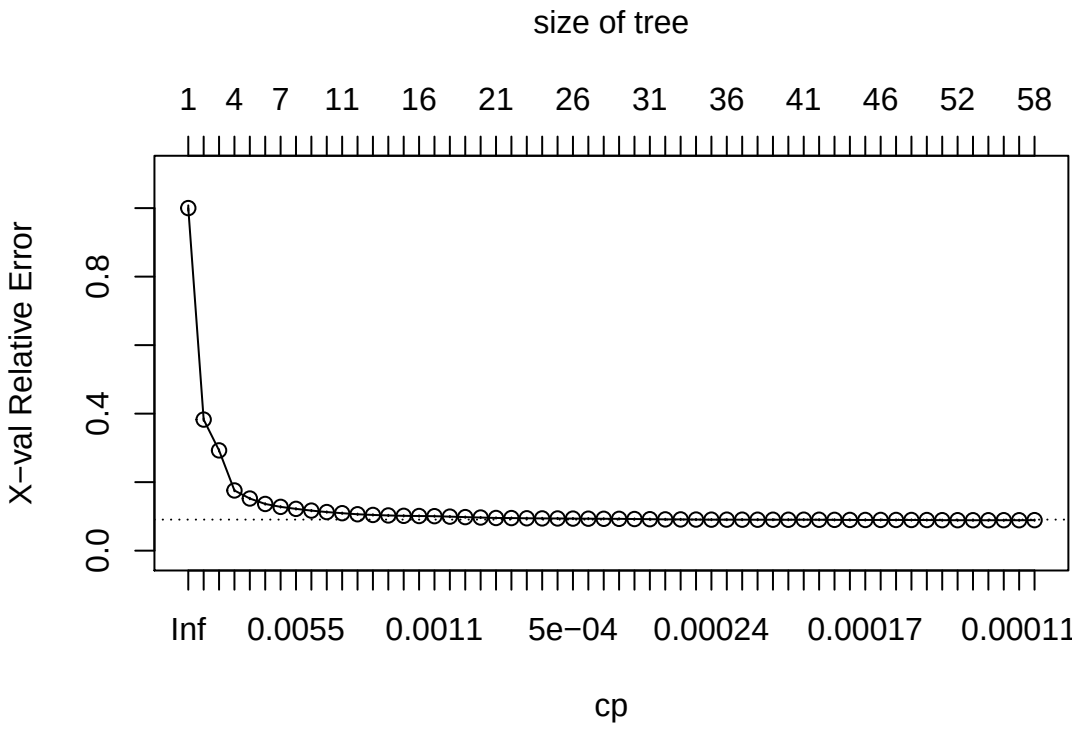
Q2.6.B

```
## size of big tree:  58
```

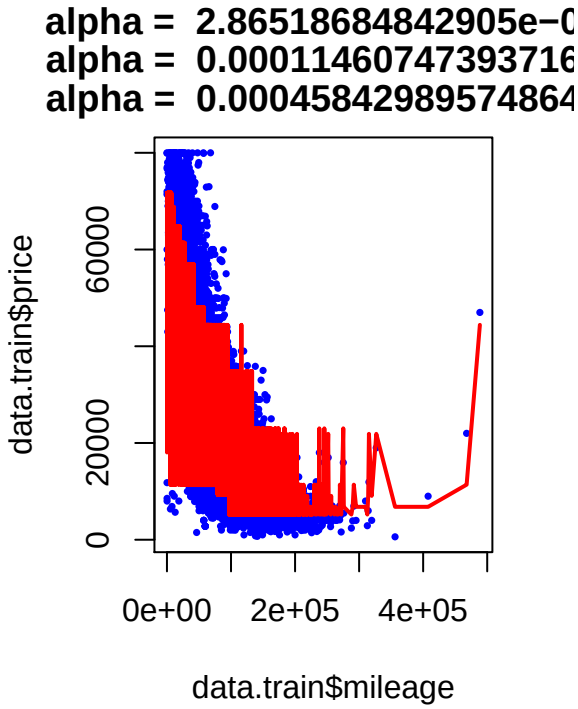
```
##
## Regression tree:
## rpart(formula = price ~ ., data = data.train, control = rpart.control(minsplit = 5,
##     cp = 1e-04, xval = 10))
##
## Variables actually used in tree construction:
## [1] mileage year
##
## Root node error: 6.712e+12/20063 = 334547097
##
## n= 20063
##
##            CP nsplit rel error   xerror      xstd
## 1  0.61745306      0  1.000000 1.000082 0.0078953
## 2  0.10473065      1  0.382547 0.382637 0.0037275
## 3  0.10327469      2  0.277816 0.292688 0.0031616
## 4  0.02098479      3  0.174542 0.175714 0.0024384
## 5  0.01779416      4  0.153557 0.152203 0.0022745
## 6  0.00850321      5  0.135763 0.136341 0.0021682
```

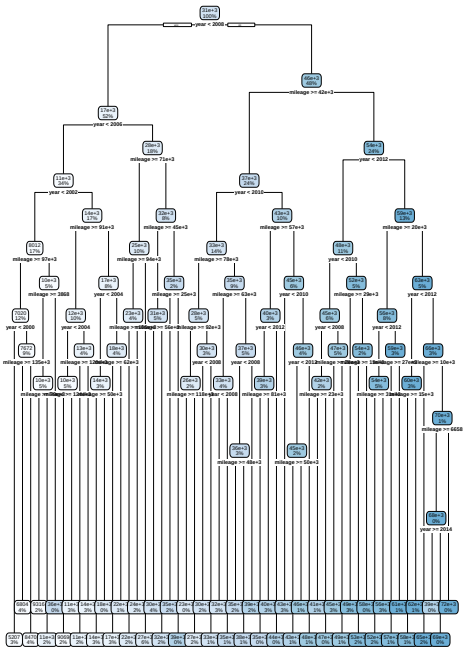
```
## 7  0.00591208      6  0.127259 0.127836 0.0021492
## 8  0.00508649      7  0.121347 0.121975 0.0020833
## 9  0.00466490      8  0.116261 0.116942 0.0020327
## 10 0.00337253      9  0.111596 0.112678 0.0020101
## 11 0.00315866     10  0.108223 0.109334 0.0019880
## 12 0.00176967     11  0.105065 0.106226 0.0019652
## 13 0.00147409     12  0.103295 0.104269 0.0019534
## 14 0.00117973     13  0.101821 0.102774 0.0019527
## 15 0.00115968     14  0.100641 0.101663 0.0019475
## 16 0.00115098     15  0.099482 0.101108 0.0019461
## 17 0.00113163     16  0.098331 0.100687 0.0019418
## 18 0.00106427     17  0.097199 0.099358 0.0019226
## 19 0.00092440     18  0.096135 0.097896 0.0019086
## 20 0.00078188     19  0.095210 0.096698 0.0018988
## 21 0.00074205     20  0.094428 0.095374 0.0018901
## 22 0.00063428     21  0.093686 0.095135 0.0018926
## 23 0.00059386     22  0.093052 0.094490 0.0019001
## 24 0.00055950     23  0.092458 0.094128 0.0018974
## 25 0.00051452     24  0.091899 0.093920 0.0018955
## 26 0.00049105     25  0.091384 0.093584 0.0018908
## 27 0.00046518     26  0.090893 0.093306 0.0018857
## 28 0.00045354     27  0.090428 0.093013 0.0018829
## 29 0.00041035     28  0.089974 0.092867 0.0018855
## 30 0.00039879     29  0.089564 0.092316 0.0018983
## 31 0.00033051     30  0.089165 0.091827 0.0018969
## 32 0.00031679     31  0.088835 0.091329 0.0018970
## 33 0.00030231     32  0.088518 0.091217 0.0018962
## 34 0.00025795     33  0.088216 0.090799 0.0018928
## 35 0.00023010     34  0.087958 0.090519 0.0018909
## 36 0.00021767     35  0.087728 0.090416 0.0018918
## 37 0.00021643     36  0.087510 0.090460 0.0018986
## 38 0.00021388     37  0.087294 0.090325 0.0018993
## 39 0.00020715     38  0.087080 0.090291 0.0019004
## 40 0.00020573     39  0.086873 0.090239 0.0019002
## 41 0.00019661     40  0.086667 0.090380 0.0019019
## 42 0.00019063     41  0.086470 0.090213 0.0018996
## 43 0.00017710     42  0.086280 0.089940 0.0018971
## 44 0.00017699     43  0.086102 0.089703 0.0018886
## 45 0.00017112     44  0.085925 0.089590 0.0018836
## 46 0.00017088     45  0.085754 0.089665 0.0018862
## 47 0.00016747     47  0.085413 0.089597 0.0018861
## 48 0.00016328     48  0.085245 0.089491 0.0018865
## 49 0.00014389     49  0.085082 0.089358 0.0018918
## 50 0.00013194     50  0.084938 0.088968 0.0018820
## 51 0.00013086     51  0.084806 0.088863 0.0018825
## 52 0.00013056     52  0.084675 0.088835 0.0018824
## 53 0.00011461     53  0.084545 0.088790 0.0018812
## 54 0.00011269     54  0.084430 0.088862 0.0018810
## 55 0.00010257     55  0.084317 0.088822 0.0018856
## 56 0.00010000     57  0.084112 0.089057 0.0018948

##              CP nsplit  rel error     xerror        xstd
## 1  0.6174530567      0 1.00000000 1.00008188 0.007895339
## 2  0.1047306454      1 0.38254694 0.38263677 0.003727517
```

```
## 3  0.1032746948      2 0.27781630 0.29268774 0.003161601
## 4  0.0209847944      3 0.17454160 0.17571415 0.002438389
## 5  0.0177941626      4 0.15355681 0.15220326 0.002274547
## 6  0.0085032114      5 0.13576265 0.13634114 0.002168153
## 7  0.0059120783      6 0.12725943 0.12783639 0.002149175
## 8  0.0050864898      7 0.12134736 0.12197549 0.002083299
## 9  0.0046649040      8 0.11626087 0.11694246 0.002032724
## 10 0.0033725344      9 0.11159596 0.11267819 0.002010097
## 11 0.0031586567     10 0.10822343 0.10933369 0.001988024
## 12 0.0017696708     11 0.10506477 0.10622574 0.001965164
## 13 0.0014740941     12 0.10329510 0.10426923 0.001953379
## 14 0.0011797309     13 0.10182101 0.10277396 0.001952667
## 15 0.0011596773     14 0.10064128 0.10166333 0.001947454
## 16 0.0011509814     15 0.09948160 0.10110845 0.001946134
## 17 0.0011316313     16 0.09833062 0.10068704 0.001941772
## 18 0.0010642661     17 0.09719899 0.09935794 0.001922598
## 19 0.0009244033     18 0.09613472 0.09789578 0.001908579
## 20 0.0007818811     19 0.09521032 0.09669750 0.001898806
## 21 0.0007420492     20 0.09442844 0.09537355 0.001890059
## 22 0.0006342795     21 0.09368639 0.09513486 0.001892586
## 23 0.0005938602     22 0.09305211 0.09449047 0.001900056
## 24 0.0005595041     23 0.09245825 0.09412795 0.001897428
## 25 0.0005145156     24 0.09189874 0.09391971 0.001895539
## 26 0.0004910513     25 0.09138423 0.09358402 0.001890844
## 27 0.0004651768     26 0.09089318 0.09330583 0.001885666
## 28 0.0004535413     27 0.09042800 0.09301276 0.001882936
## 29 0.0004103541     28 0.08997446 0.09286655 0.001885489
## 30 0.0003987939     29 0.08956410 0.09231642 0.001898316
## 31 0.0003305128     30 0.08916531 0.09182707 0.001896877
## 32 0.0003167933     31 0.08883480 0.09132871 0.001896989
## 33 0.0003023093     32 0.08851800 0.09121745 0.001896209
## 34 0.0002579513     33 0.08821569 0.09079937 0.001892827
## 35 0.0002300995     34 0.08795774 0.09051920 0.001890865
## 36 0.0002176682     35 0.08772764 0.09041585 0.001891826
## 37 0.0002164306     36 0.08750997 0.09046049 0.001898581
## 38 0.0002138809     37 0.08729354 0.09032494 0.001899313
## 39 0.0002071544     38 0.08707966 0.09029096 0.001900353
## 40 0.0002057281     39 0.08687251 0.09023936 0.001900221
## 41 0.0001966058     40 0.08666678 0.09037993 0.001901871
## 42 0.0001906289     41 0.08647017 0.09021335 0.001899636
## 43 0.0001770995     42 0.08627955 0.08993990 0.001897075
## 44 0.0001769870     43 0.08610245 0.08970281 0.001888605
## 45 0.0001711229     44 0.08592546 0.08959010 0.001883590
## 46 0.0001708848     45 0.08575434 0.08966493 0.001886182
## 47 0.0001674737     47 0.08541257 0.08959737 0.001886140
## 48 0.0001632774     48 0.08524509 0.08949091 0.001886539
## 49 0.0001438928     49 0.08508182 0.08935785 0.001891784
## 50 0.0001319400     50 0.08493792 0.08896826 0.001881984
## 51 0.0001308631     51 0.08480598 0.08886275 0.001882538
## 52 0.0001305619     52 0.08467512 0.08883523 0.001882394
## 53 0.0001146075     53 0.08454456 0.08879009 0.001881250
## 54 0.0001126930     54 0.08442995 0.08886166 0.001881031
## 55 0.0001025745     55 0.08431726 0.08882167 0.001885605
## 56 0.0001000000     57 0.08411211 0.08905731 0.001894783
```




```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```





```
## RMSE is  5304.664
```

Q 2.7

```r
set.seed(2018) #always set the seed!
data.train <- raw.data

tree <- rpart(price~.,      #Formula
              data = data.train ,#Data
              control = rpart.control(minsplit=5,#the minimum number of observations that must exist in
              cp=0.0001, #complexity, the lower, the larger the tree is
              xval=10   #number of cross validations
              ))

nbig <- length(unique(tree$where))
cat('size of big tree: ',nbig,'\n')
```

```
## size of big tree:  88
```

```r
(cptable <- printcp(tree))
```

```
##
## Regression tree:
## rpart(formula = price ~ ., data = data.train, control = rpart.control(minsplit = 5,
##     cp = 1e-04, xval = 10))
##
## Variables actually used in tree construction:
## [1] color        displacement mileage      region       soundSystem
## [6] trim         wheelType    year
##
## Root node error: 6.712e+12/20063 = 334547097
##
## n= 20063
##
##            CP nsplit rel error   xerror      xstd
## 1  0.61745306      0  1.000000 1.000082 0.0078953
## 2  0.10473065      1  0.382547 0.382637 0.0037275
## 3  0.10327469      2  0.277816 0.292688 0.0031616
## 4  0.02098479      3  0.174542 0.175714 0.0024384
## 5  0.01779416      4  0.153557 0.152203 0.0022745
## 6  0.00850321      5  0.135763 0.136341 0.0021682
## 7  0.00787159      6  0.127259 0.127836 0.0021492
## 8  0.00602254      7  0.119388 0.120110 0.0019840
## 9  0.00591208      8  0.113365 0.114435 0.0018870
## 10 0.00508649      9  0.107453 0.107921 0.0017928
## 11 0.00392445     10  0.102367 0.103186 0.0017654
## 12 0.00352162     11  0.098442 0.099633 0.0017020
## 13 0.00337253     12  0.094921 0.096579 0.0016553
## 14 0.00328152     13  0.091548 0.091737 0.0016068
## 15 0.00312226     14  0.088267 0.089540 0.0015801
## 16 0.00185921     15  0.085144 0.086283 0.0014912
## 17 0.00163518     16  0.083285 0.084543 0.0014758
## 18 0.00147894     17  0.081650 0.083259 0.0014718
## 19 0.00147409     18  0.080171 0.082325 0.0014730
## 20 0.00126758     19  0.078697 0.079956 0.0014351
## 21 0.00121757     20  0.077429 0.079094 0.0014281
## 22 0.00117973     22  0.074994 0.078215 0.0014197
## 23 0.00115098     23  0.073814 0.076342 0.0014021
```

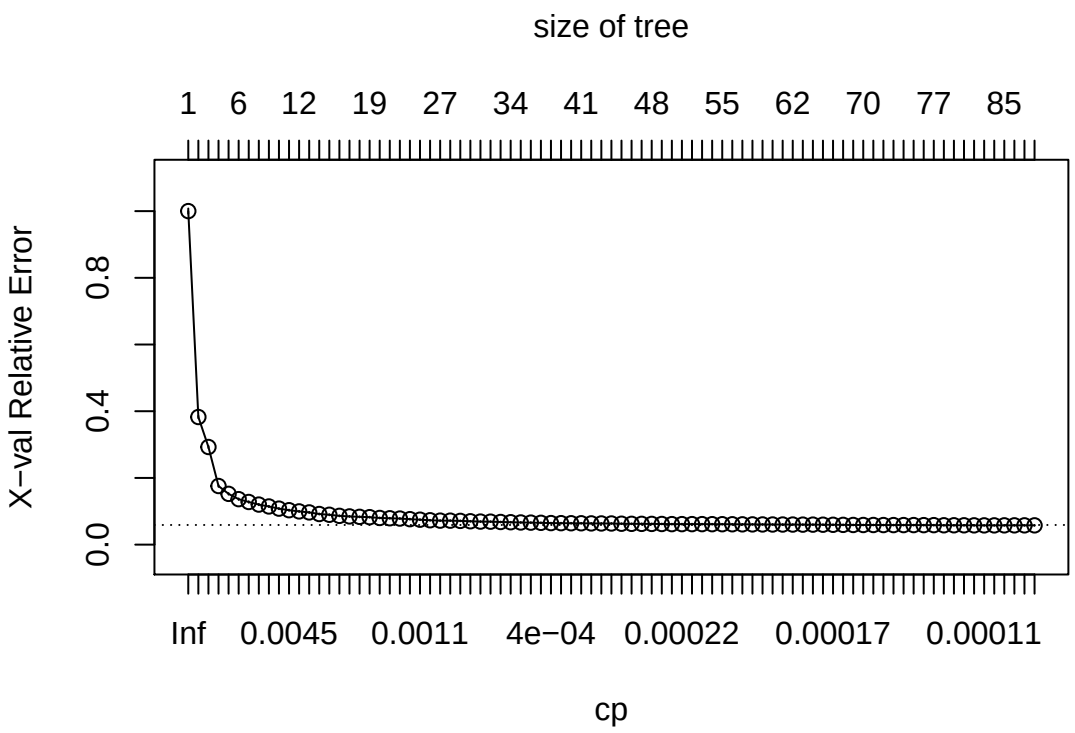
```
## 24 0.00108525     24  0.072664 0.074778 0.0013862
## 25 0.00089550     25  0.071578 0.073059 0.0013450
## 26 0.00089536     26  0.070683 0.071955 0.0013310
## 27 0.00087479     27  0.069787 0.071877 0.0013293
## 28 0.00080420     28  0.068913 0.071158 0.0013200
## 29 0.00076717     29  0.068108 0.070036 0.0012960
## 30 0.00073484     30  0.067341 0.069087 0.0012859
## 31 0.00072036     31  0.066606 0.068822 0.0012834
## 32 0.00061502     32  0.065886 0.067841 0.0012678
## 33 0.00061290     33  0.065271 0.067162 0.0012600
## 34 0.00049105     34  0.064658 0.066446 0.0012569
## 35 0.00045354     35  0.064167 0.065940 0.0012503
## 36 0.00041035     36  0.063714 0.065401 0.0012503
## 37 0.00039879     37  0.063303 0.064546 0.0012710
## 38 0.00039519     38  0.062904 0.064337 0.0012703
## 39 0.00039098     39  0.062509 0.064140 0.0012704
## 40 0.00038133     40  0.062118 0.063994 0.0012705
## 41 0.00036692     41  0.061737 0.063645 0.0012684
## 42 0.00035976     42  0.061370 0.063493 0.0012643
## 43 0.00033843     43  0.061010 0.063367 0.0012649
## 44 0.00029644     44  0.060672 0.062669 0.0012569
## 45 0.00029192     45  0.060375 0.062341 0.0012552
## 46 0.00028579     46  0.060083 0.062262 0.0012551
## 47 0.00026241     47  0.059798 0.062096 0.0012538
## 48 0.00025748     48  0.059535 0.061884 0.0012530
## 49 0.00022157     49  0.059278 0.061471 0.0012482
## 50 0.00021833     50  0.059056 0.061482 0.0012551
## 51 0.00021770     51  0.058838 0.061455 0.0012571
## 52 0.00021767     52  0.058620 0.061435 0.0012571
## 53 0.00021388     53  0.058402 0.061335 0.0012570
## 54 0.00020667     54  0.058189 0.061188 0.0012565
## 55 0.00020197     55  0.057982 0.061051 0.0012577
## 56 0.00020136     56  0.057780 0.060851 0.0012553
## 57 0.00019661     57  0.057579 0.060657 0.0012531
## 58 0.00019338     58  0.057382 0.060500 0.0012553
## 59 0.00019063     59  0.057189 0.060376 0.0012550
## 60 0.00018875     60  0.056998 0.060277 0.0012555
## 61 0.00018424     61  0.056809 0.060168 0.0012540
## 62 0.00017991     62  0.056625 0.059938 0.0012490
## 63 0.00017699     63  0.056445 0.059904 0.0012493
## 64 0.00017573     64  0.056268 0.059726 0.0012475
## 65 0.00017088     65  0.056092 0.059450 0.0012388
## 66 0.00016253     67  0.055751 0.059199 0.0012401
## 67 0.00015777     68  0.055588 0.058933 0.0012379
## 68 0.00015524     69  0.055430 0.058848 0.0012383
## 69 0.00014553     70  0.055275 0.058823 0.0012406
## 70 0.00014389     71  0.055129 0.058642 0.0012420
## 71 0.00014194     72  0.054986 0.058624 0.0012421
## 72 0.00014185     73  0.054844 0.058624 0.0012421
## 73 0.00013471     74  0.054702 0.058580 0.0012423
## 74 0.00013124     75  0.054567 0.058462 0.0012421
## 75 0.00013046     76  0.054436 0.058419 0.0012413
## 76 0.00011771     77  0.054305 0.058060 0.0012367
## 77 0.00011768     78  0.054188 0.057990 0.0012357
```

```
## 78 0.00011660     79  0.054070 0.057964 0.0012356
## 79 0.00010822     80  0.053953 0.057794 0.0012318
## 80 0.00010669     81  0.053845 0.057702 0.0012336
## 81 0.00010436     82  0.053738 0.057735 0.0012351
## 82 0.00010065     84  0.053530 0.057773 0.0012364
## 83 0.00010065     85  0.053429 0.057751 0.0012362
## 84 0.00010055     86  0.053328 0.057751 0.0012362
## 85 0.00010000     87  0.053228 0.057758 0.0012363
```

```
##              CP nsplit  rel error     xerror        xstd
## 1  0.6174530567      0 1.00000000 1.00008188 0.007895339
## 2  0.1047306454      1 0.38254694 0.38263677 0.003727517
## 3  0.1032746948      2 0.27781630 0.29268774 0.003161601
## 4  0.0209847944      3 0.17454160 0.17571415 0.002438389
## 5  0.0177941626      4 0.15355681 0.15220326 0.002274547
## 6  0.0085032114      5 0.13576265 0.13634114 0.002168153
## 7  0.0078715934      6 0.12725943 0.12783639 0.002149175
## 8  0.0060225351      7 0.11938784 0.12011037 0.001984010
## 9  0.0059120783      8 0.11336531 0.11443489 0.001887024
## 10 0.0050864898      9 0.10745323 0.10792126 0.001792768
## 11 0.0039244521     10 0.10236674 0.10318645 0.001765400
## 12 0.0035216184     11 0.09844229 0.09963284 0.001701974
## 13 0.0033725344     12 0.09492067 0.09657936 0.001655281
## 14 0.0032815216     13 0.09154813 0.09173689 0.001606769
## 15 0.0031222636     14 0.08826661 0.08954043 0.001580114
## 16 0.0018592053     15 0.08514435 0.08628347 0.001491162
## 17 0.0016351773     16 0.08328514 0.08454280 0.001475757
## 18 0.0014789407     17 0.08164997 0.08325873 0.001471761
## 19 0.0014740941     18 0.08017102 0.08232541 0.001473037
## 20 0.0012675754     19 0.07869693 0.07995598 0.001435076
## 21 0.0012175709     20 0.07742936 0.07909432 0.001428143
## 22 0.0011797309     22 0.07499421 0.07821460 0.001419704
## 23 0.0011509814     23 0.07381448 0.07634212 0.001402074
## 24 0.0010852536     24 0.07266350 0.07477766 0.001386173
## 25 0.0008955012     25 0.07157825 0.07305903 0.001345041
## 26 0.0008953571     26 0.07068275 0.07195505 0.001331050
## 27 0.0008747947     27 0.06978739 0.07187707 0.001329295
## 28 0.0008042039     28 0.06891259 0.07115814 0.001320015
## 29 0.0007671658     29 0.06810839 0.07003635 0.001296026
## 30 0.0007348386     30 0.06734122 0.06908717 0.001285931
## 31 0.0007203571     31 0.06660639 0.06882169 0.001283438
## 32 0.0006150199     32 0.06588603 0.06784129 0.001267795
## 33 0.0006129028     33 0.06527101 0.06716232 0.001259967
## 34 0.0004910513     34 0.06465811 0.06644550 0.001256937
## 35 0.0004535413     35 0.06416706 0.06594005 0.001250310
## 36 0.0004103541     36 0.06371351 0.06540121 0.001250297
## 37 0.0003987939     37 0.06330316 0.06454632 0.001271005
## 38 0.0003951917     38 0.06290437 0.06433661 0.001270325
## 39 0.0003909842     39 0.06250917 0.06414007 0.001270428
## 40 0.0003813280     40 0.06211819 0.06399390 0.001270488
## 41 0.0003669194     41 0.06173686 0.06364527 0.001268393
## 42 0.0003597575     42 0.06136994 0.06349286 0.001264349
## 43 0.0003384270     43 0.06101019 0.06336711 0.001264924
## 44 0.0002964387     44 0.06067176 0.06266905 0.001256858
```

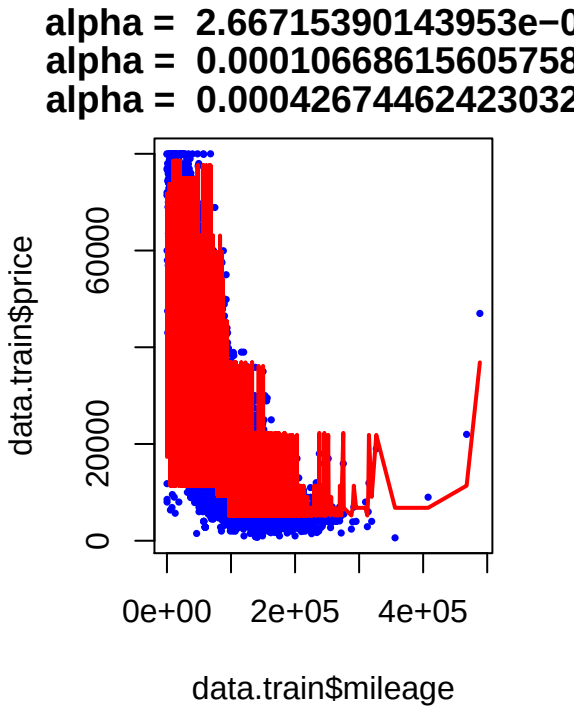
```
## 45 0.0002919218     45 0.06037532 0.06234127 0.001255180
## 46 0.0002857931     46 0.06008340 0.06226195 0.001255057
## 47 0.0002624093     47 0.05979760 0.06209645 0.001253787
## 48 0.0002574764     48 0.05953520 0.06188363 0.001253036
## 49 0.0002215679     49 0.05927772 0.06147069 0.001248218
## 50 0.0002183341     50 0.05905615 0.06148183 0.001255124
## 51 0.0002177049     51 0.05883782 0.06145549 0.001257074
## 52 0.0002176682     52 0.05862011 0.06143474 0.001257083
## 53 0.0002138809     53 0.05840244 0.06133459 0.001256960
## 54 0.0002066716     54 0.05818856 0.06118756 0.001256549
## 55 0.0002019670     55 0.05798189 0.06105122 0.001257708
## 56 0.0002013568     56 0.05777992 0.06085055 0.001255260
## 57 0.0001966058     57 0.05757857 0.06065732 0.001253074
## 58 0.0001933824     58 0.05738196 0.06050032 0.001255275
## 59 0.0001906289     59 0.05718858 0.06037623 0.001254957
## 60 0.0001887522     60 0.05699795 0.06027690 0.001255538
## 61 0.0001842448     61 0.05680920 0.06016805 0.001253957
## 62 0.0001799124     62 0.05662495 0.05993839 0.001248999
## 63 0.0001769870     63 0.05644504 0.05990430 0.001249264
## 64 0.0001757330     64 0.05626805 0.05972644 0.001247539
## 65 0.0001708848     65 0.05609232 0.05944988 0.001238757
## 66 0.0001625325     67 0.05575055 0.05919865 0.001240101
## 67 0.0001577677     68 0.05558802 0.05893319 0.001237936
## 68 0.0001552353     69 0.05543025 0.05884814 0.001238311
## 69 0.0001455282     70 0.05527502 0.05882347 0.001240595
## 70 0.0001438928     71 0.05512949 0.05864195 0.001242046
## 71 0.0001419400     72 0.05498559 0.05862392 0.001242135
## 72 0.0001418470     73 0.05484365 0.05862392 0.001242135
## 73 0.0001347120     74 0.05470181 0.05857987 0.001242343
## 74 0.0001312422     75 0.05456710 0.05846222 0.001242086
## 75 0.0001304606     76 0.05443585 0.05841897 0.001241324
## 76 0.0001177091     77 0.05430539 0.05806026 0.001236730
## 77 0.0001176782     78 0.05418768 0.05799037 0.001235723
## 78 0.0001166027     79 0.05407001 0.05796444 0.001235553
## 79 0.0001082188     80 0.05395340 0.05779435 0.001231826
## 80 0.0001066862     81 0.05384518 0.05770173 0.001233603
## 81 0.0001043635     82 0.05373850 0.05773473 0.001235134
## 82 0.0001006511     84 0.05352977 0.05777333 0.001236378
## 83 0.0001006466     85 0.05342912 0.05775127 0.001236156
## 84 0.0001005469     86 0.05332847 0.05775127 0.001236156
## 85 0.0001000000     87 0.05322793 0.05775817 0.001236311
```

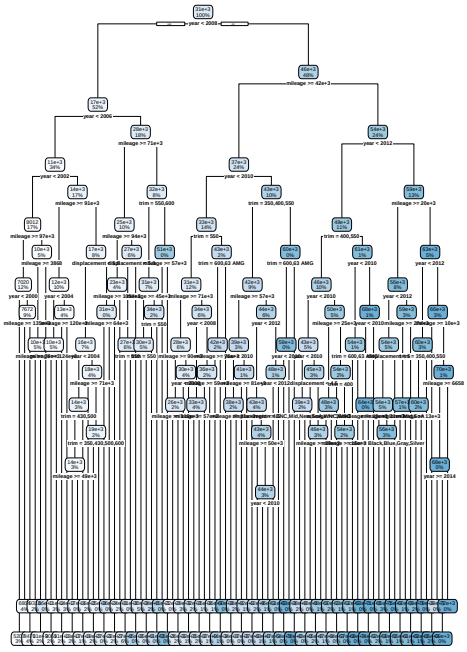
```r
bestcp <- cptable[ which.min(cptable[,"xerror"]), "CP" ] # this is the optimal cp parameter

plotcp(tree) # plot results
```




```
# show fit from some trees
oo = order(data.train$mileage)
cpvec = c(bestcp / 4, bestcp,bestcp*4)

par(mfrow=c(1,2))
plot(data.train$mileage, data.train$price, pch=16, col='blue', cex=.5)
ptree = prune(tree, cp=bestcp)
pfit = predict(ptree)
lines(data.train$mileage[oo],pfit[oo],col='red',lwd=2)
title(paste('alpha = ',cpvec))
rpart.plot(ptree)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```





```
cat('RMSE is ',sqrt(mean((data.train$price -predict(tree, data.train))^2)),'\n')
```

```
## RMSE is  4219.864
```