Machine Learning

HW1

Xin Cheng
runnytone@uchicago.edu

01/20/2018

```r
#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
```

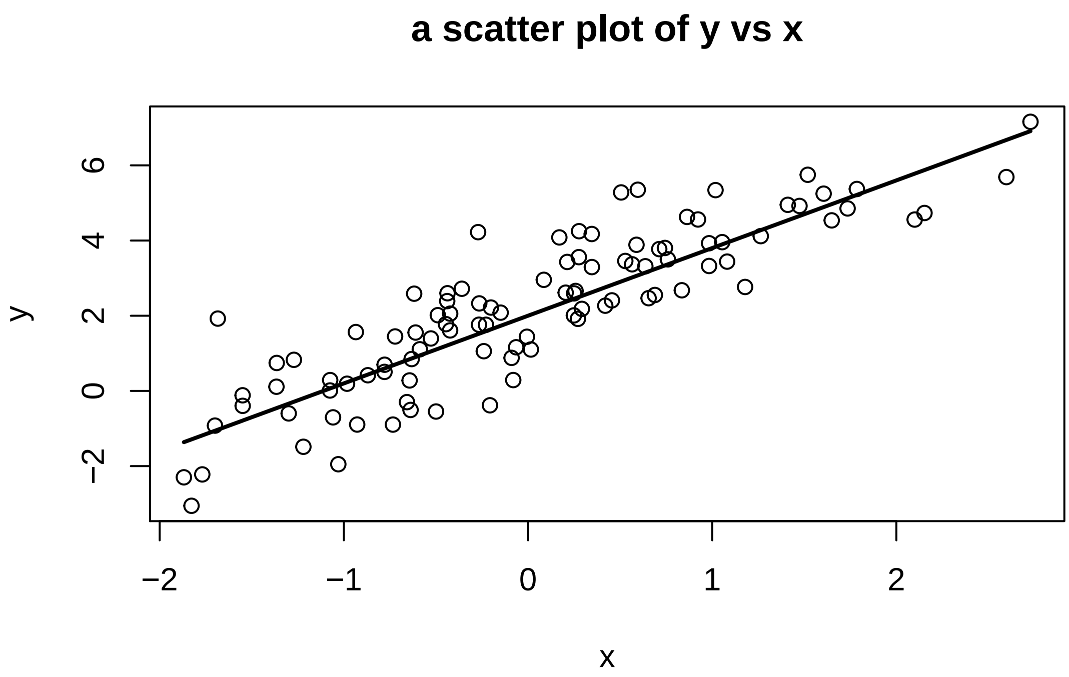
Q1

Q1.1

```r
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

```r
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)
```



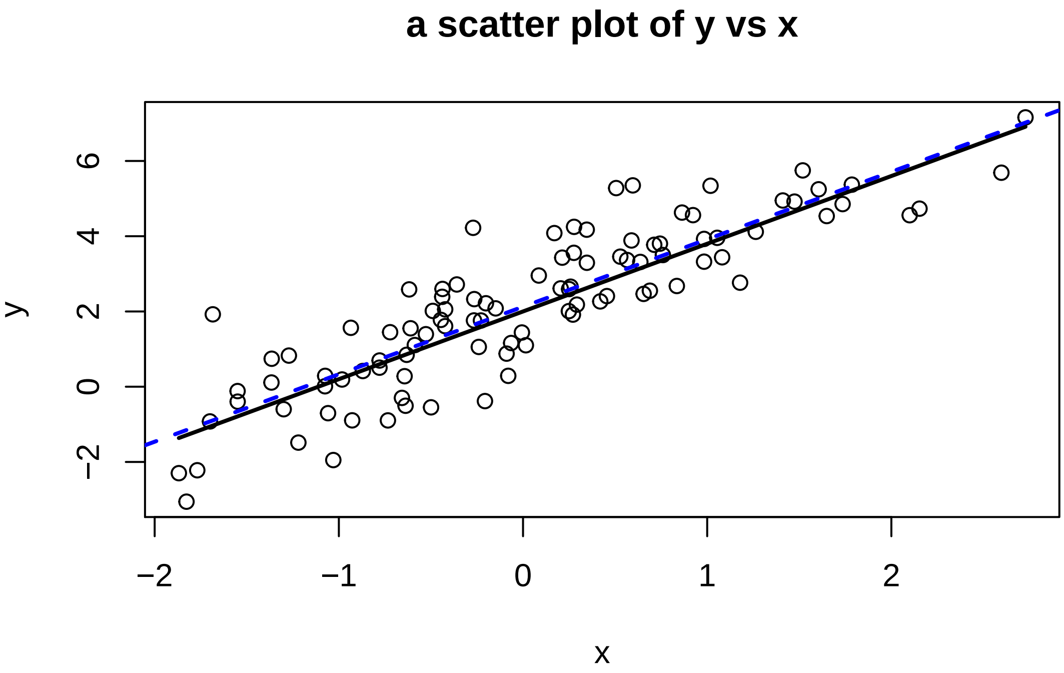
```
## integer(0)
```

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")
```



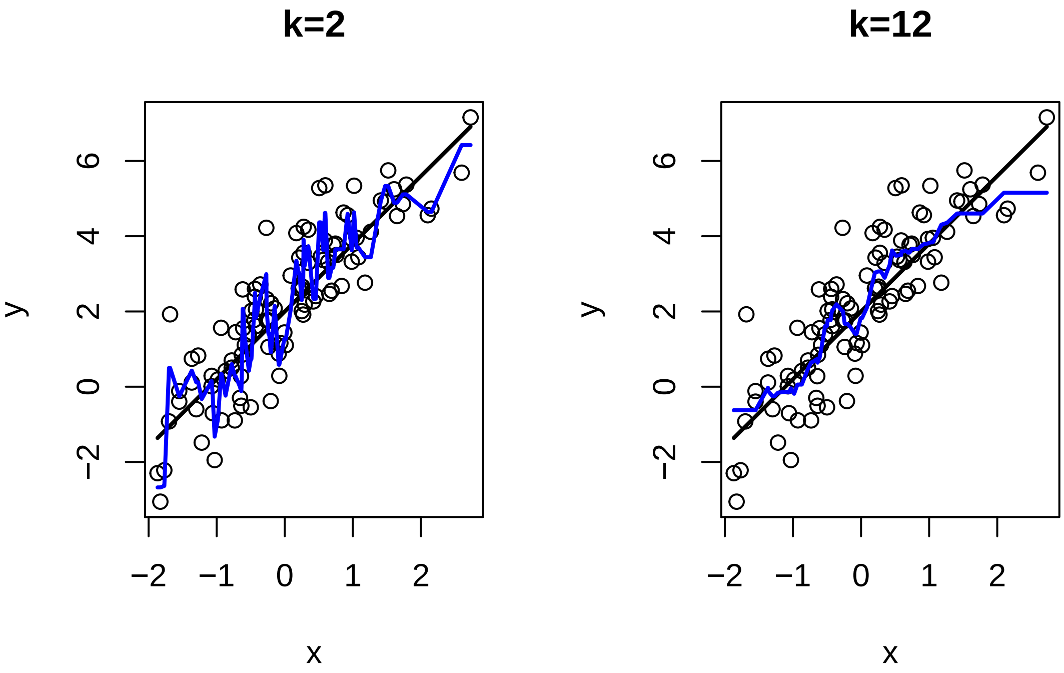
```
## integer(0)
```

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)
```



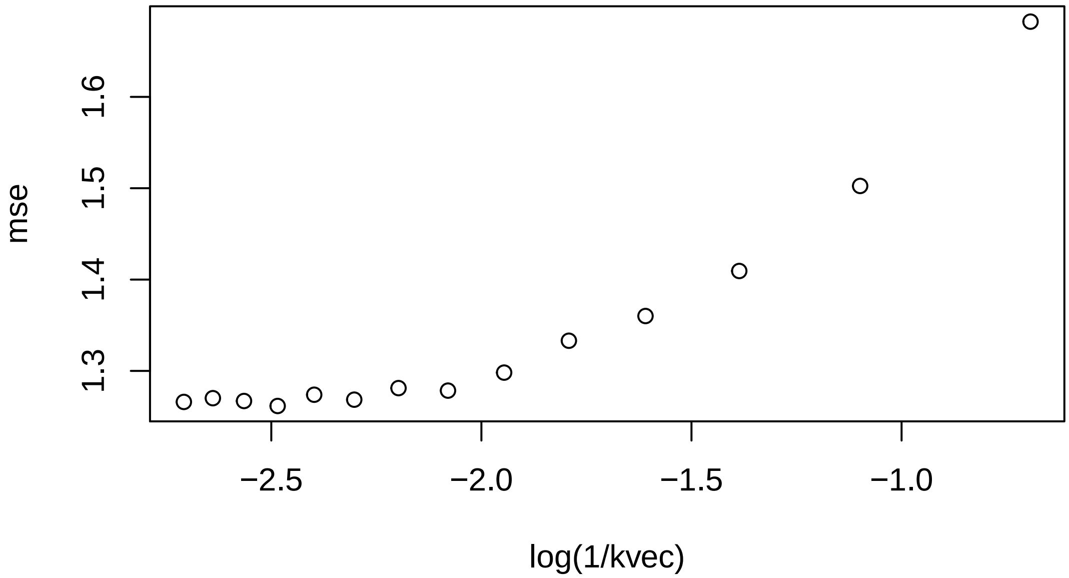
```
## integer(0)
```

Q1.5

```r
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
```

```r
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

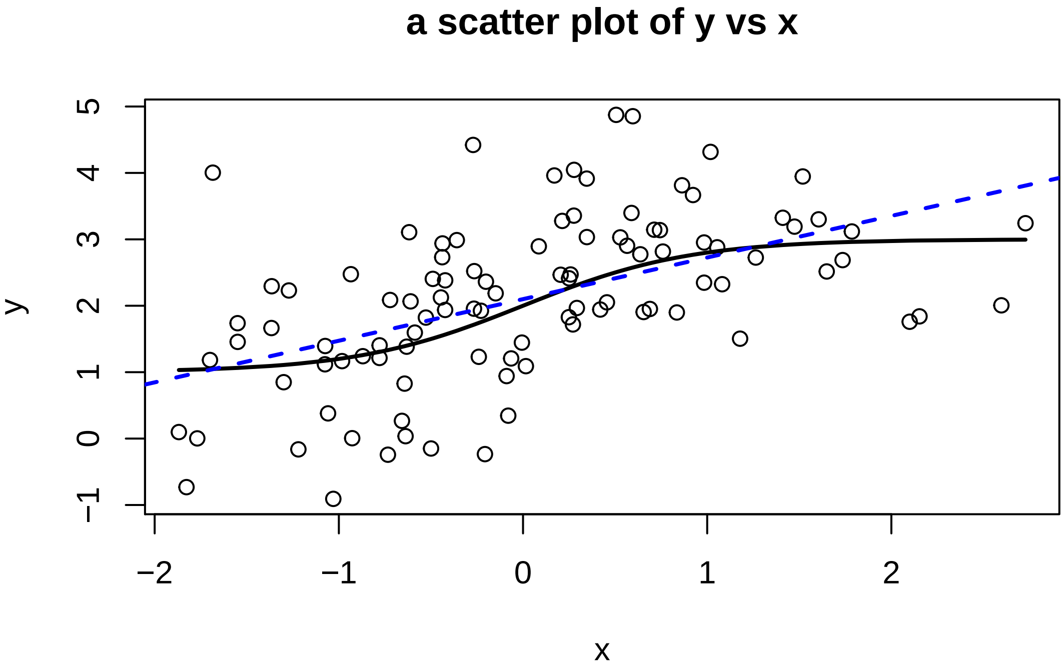
kknn(y~x, train, train, k = k, kernel = “rectangular”)$fitted gives you prediction of y in train dataset

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

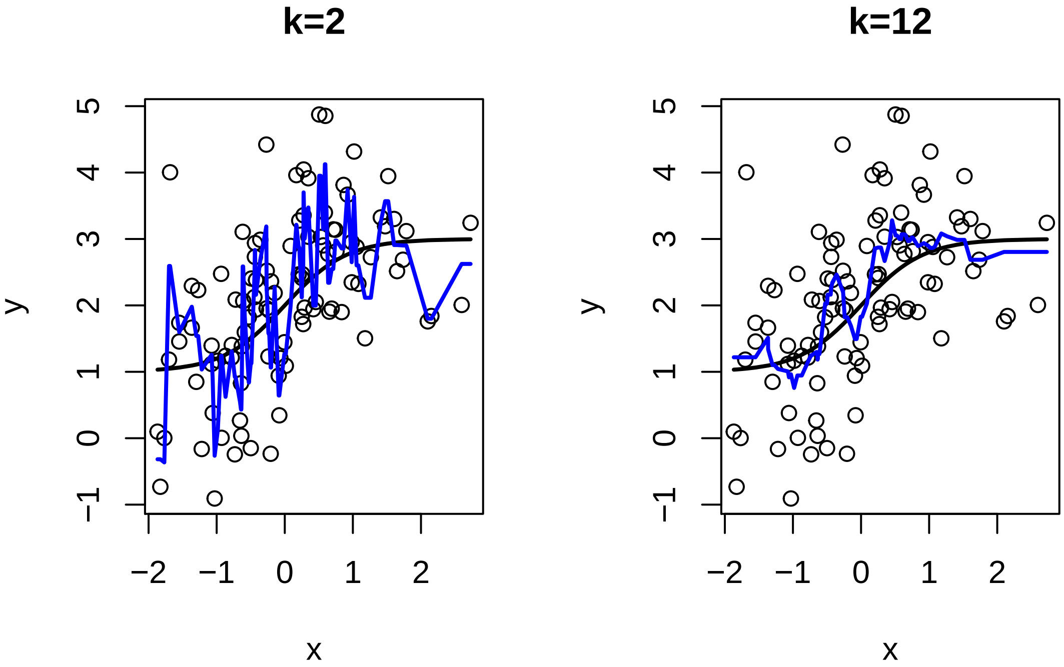
The solution should consider naming the columns of the two variables the same and be agnostic to who you are calling in the formula attribute. Otherwise the program will consider that the only information that is reliable is the one in the formula and will only use it and predict a database with its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

Q1.6

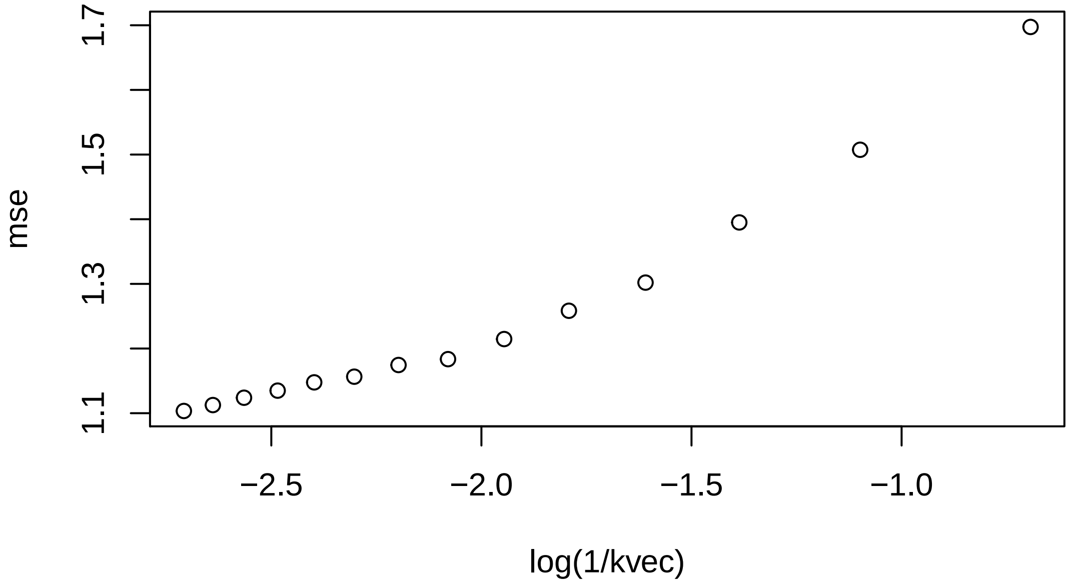
```
##
## Call:
## lm(formula = train$y ~ train$x, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.36223 -0.60754  0.07714  0.60121  2.95953
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.0999     0.1070   19.63  < 2e-16 ***
## 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
```

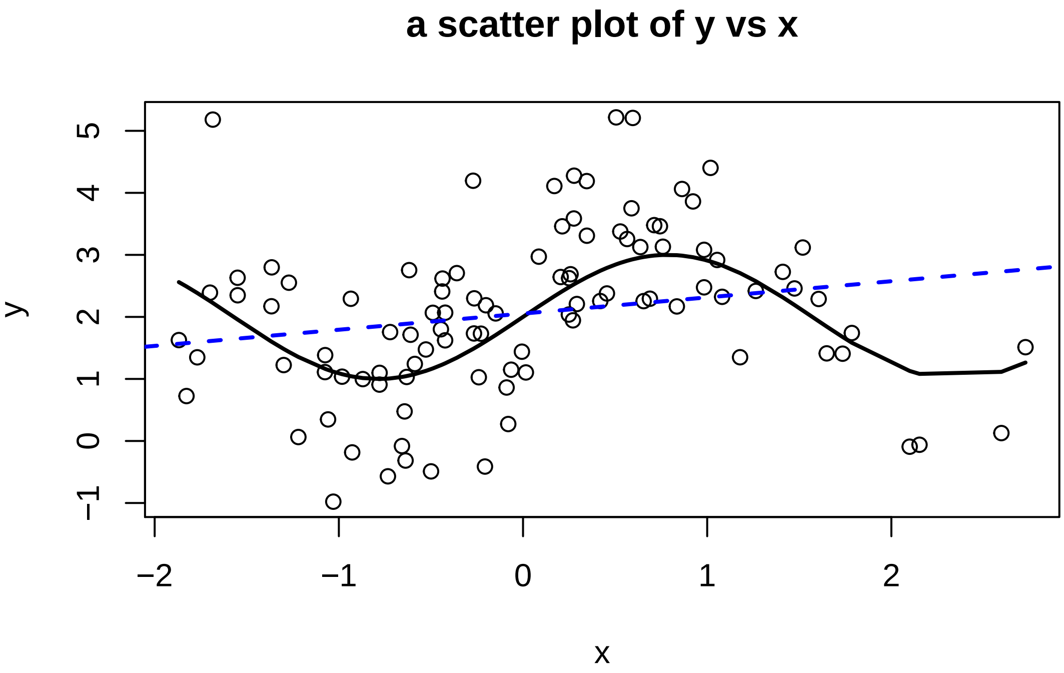


```
## integer(0)
```

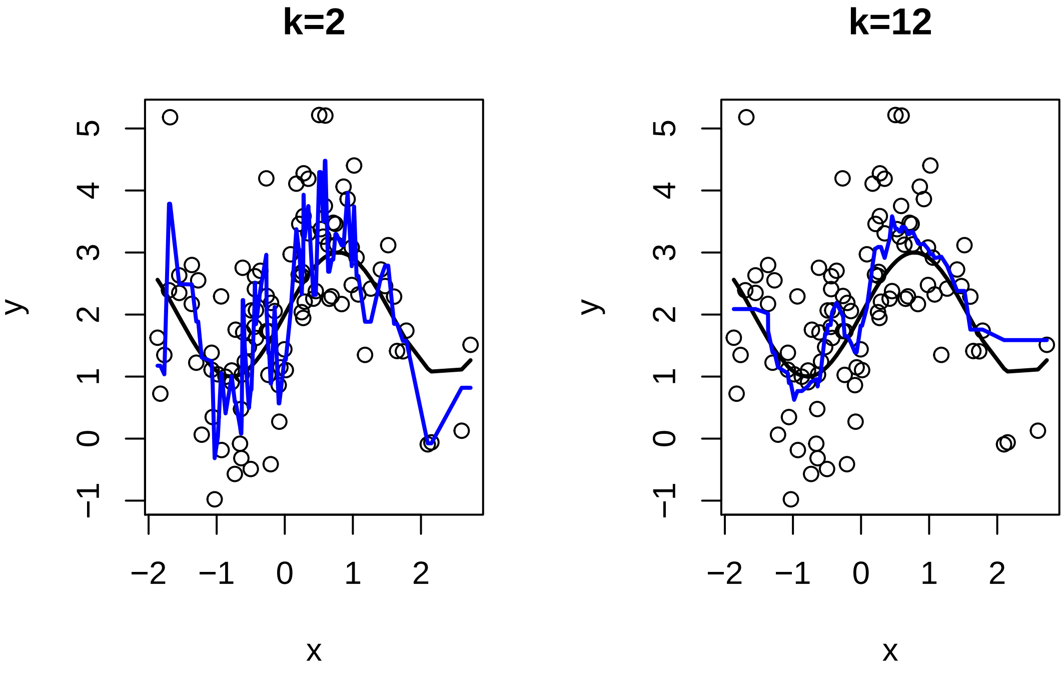
```
## the smallest MSE of knn is 1.10342 ,while the MES of linear gression is 1.058055 .
##  linear gression model fits better in this case.
```

Q1.7

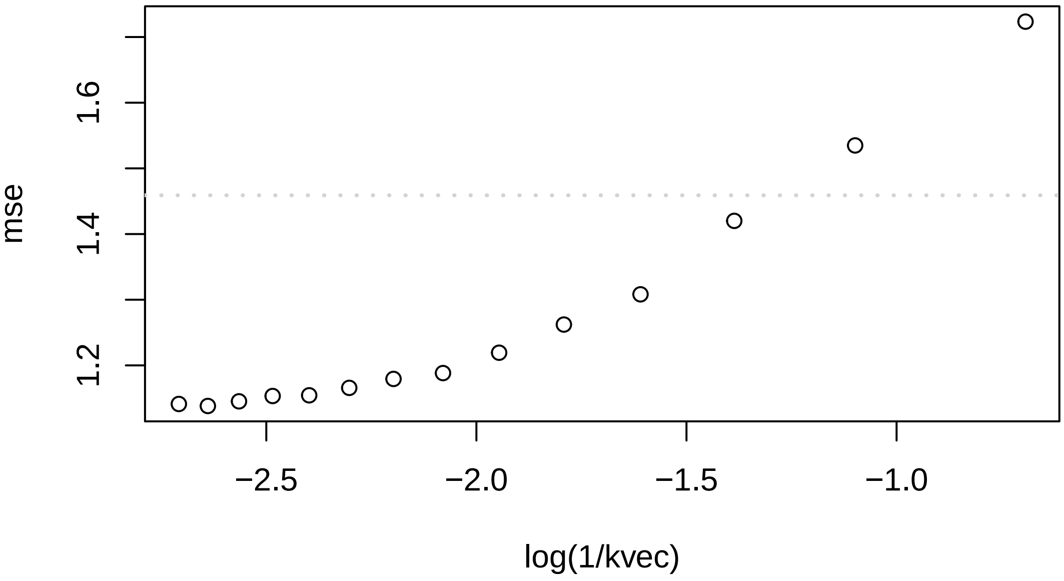
```
##
## Call:
## lm(formula = train$y ~ train$x, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7639 -0.8357  0.0382  0.7927  3.5662
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.0539     0.1311  15.670   <2e-16 ***
## train$x       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
```

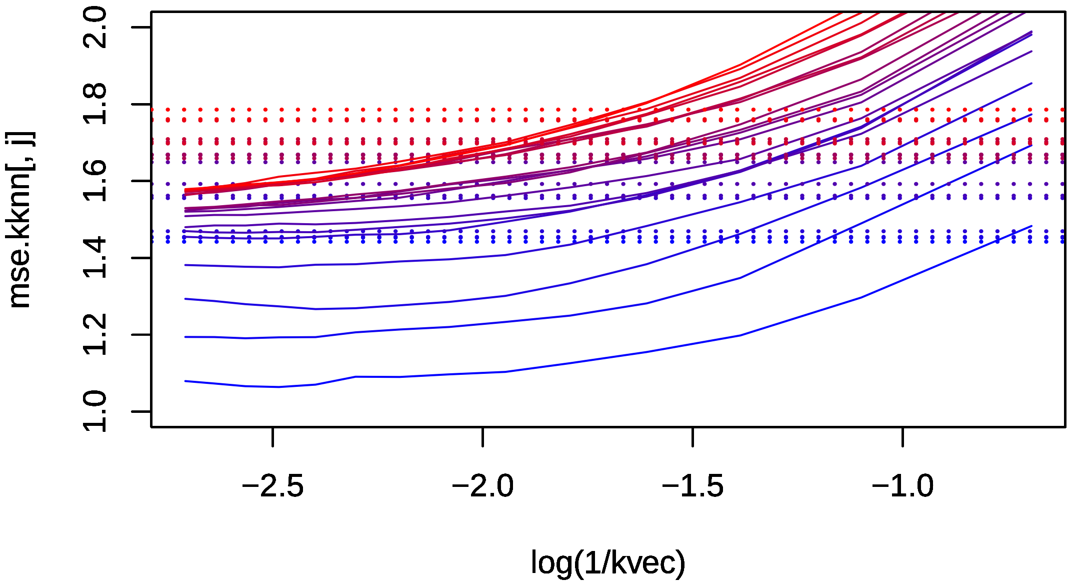


```
## integer(0)
```

```
## 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

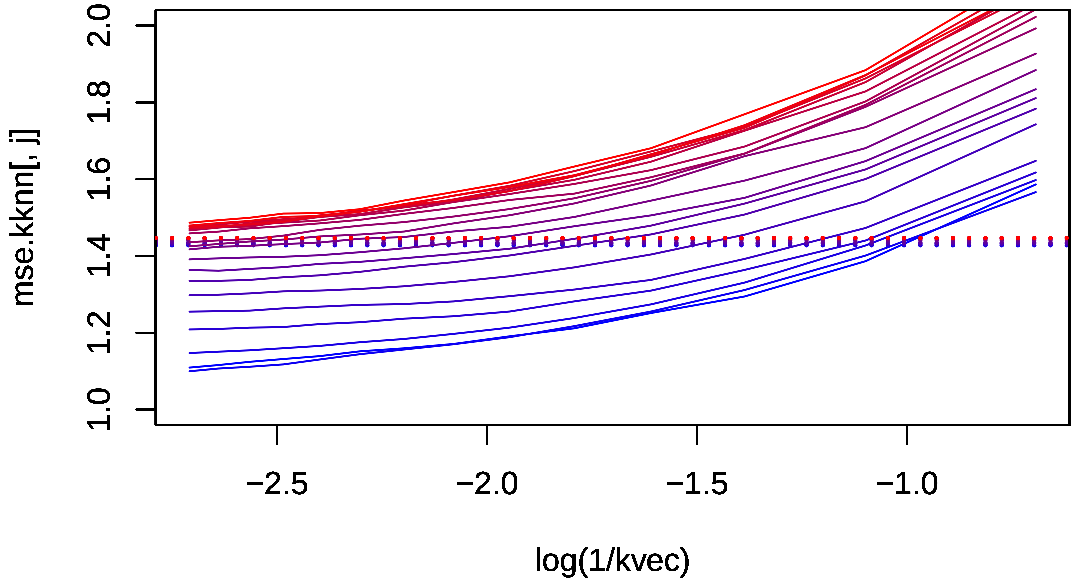
```
## Warning in par(new = TRUE): calling par(new=TRUE) with no plot
```



```
## 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
```