20152410 배형준 Data Mining HW1

Contents

load dataset and split train and test 2

(a) Polynomial regression 4

(1) validation set 4

(2) LOOCV 6

(3) 10 fold cv 8

(b) KNN regression 10

(1) validation set 10

(2) LOOCV 12

(3) 10 fold cv 14

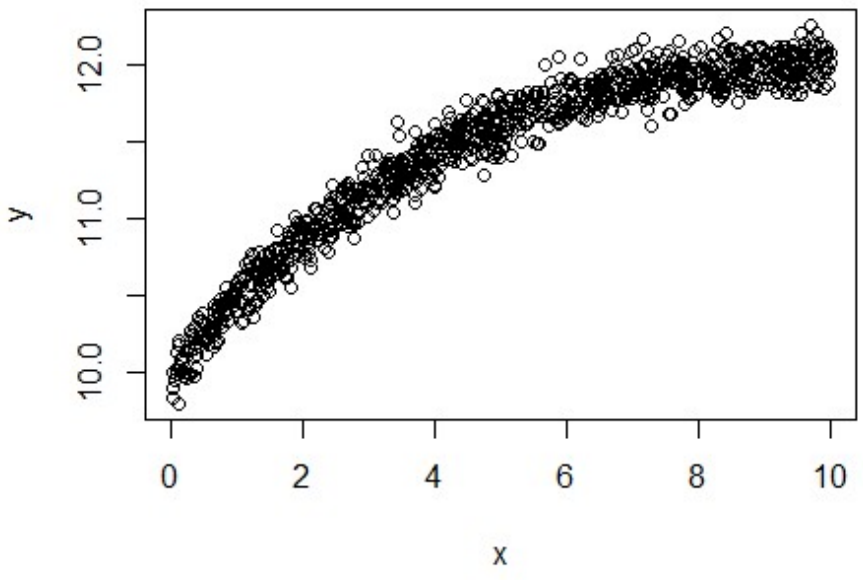
Use the attached “data2.txt”. Use set.seed(1) to make a test set of 300 observations from the data. Use the remaining 700 observations and apply (1) validation set, (2) LOOCV, and (3) 10-fold cross validation approaches setting the seed with your university ID to answer the following subproblems:

load dataset and split train and test

```
student = 20152410
dataset = read.csv('./data2.txt', header=TRUE, sep=' ')
head(dataset)
```

```
##          y        x
## 1 11.06616 2.655087
## 2 11.33982 3.721239
## 3 11.60248 5.728534
## 4 11.93088 9.082078
## 5 10.88337 2.016819
## 6 12.02761 8.983897
```

```
# reorder variables
data = data.frame(x = dataset[, 2], y = dataset[, 1])

# 데이터의 분포 시각적으로 확인
plot(data[, 1], data[, 2], xlab='x', ylab='y')
```



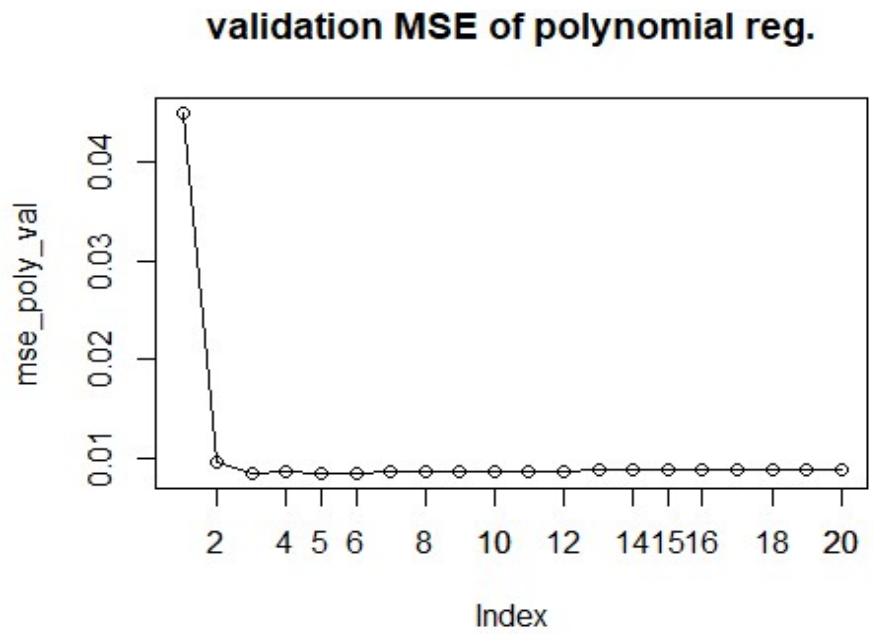
```r
# split trainset and testset
set.seed(1)
n = dim(data)[1]
train_size = 0.7
train_index = sample(1:n, n*train_size, replace=FALSE)
trainset = data[train_index, ]
testset = data[-train_index, ]

# for fitted curve in graph
x = seq(0, 10, 0.001)
x_linspace= data.frame(x = x)
```

(a) Polynomial regression

Determine the best polynomial regression model to predict y by x. Draw a scatter plot with the fitted curve. Report the test MSE of your final model using the test set.

(1) validation set

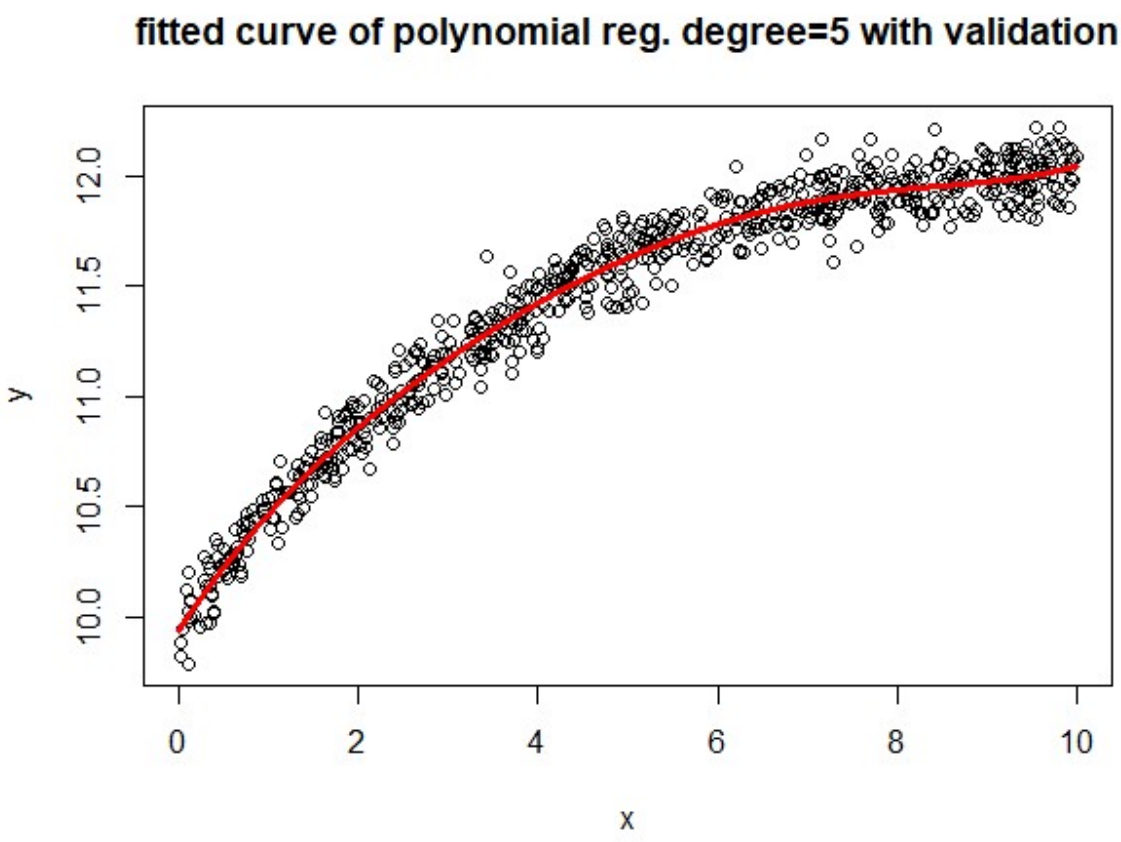
```
# make validation set
set.seed(student)
m = dim(trainset)[1]
val_size = 0.3
val_index = sample(1:m, m*val_size, replace=FALSE)
train = trainset[-val_index, ]
val = trainset[val_index, ]

mse_poly_val = c()


# model learning
for (i in 1:20) {
  model_poly = lm(y ~ poly(x, i), data=train)
  predict_value = predict(model_poly, newdata = val)
  mse = mean((val[, 'y'] - predict_value)^2)

  mse_poly_val[i] = mse
}

# 차수가 2 부터 validation mse 가 급격하게 감소하는 것을 확인할 수 있다.
plot(mse_poly_val, type='o', xlim=c(1, 20), main='validation MSE of polynomial reg.
')
axis(side=1, at=seq(0, 20, 2))
```



```r
# calculate test mse
degree = which.min(mse_poly_val)
model_poly = lm(y ~ poly(x, degree), data=trainset)
predict_value = predict(model_poly, newdata=testset)
test_mse_poly_val = mean((testset[, 'y'] - predict_value)^2)
cat('validation method 를 이용해 구한 polynomial regression 의 차수는',
    degree, '이고 test MSE 는', test_mse_poly_val, '이다.')
```

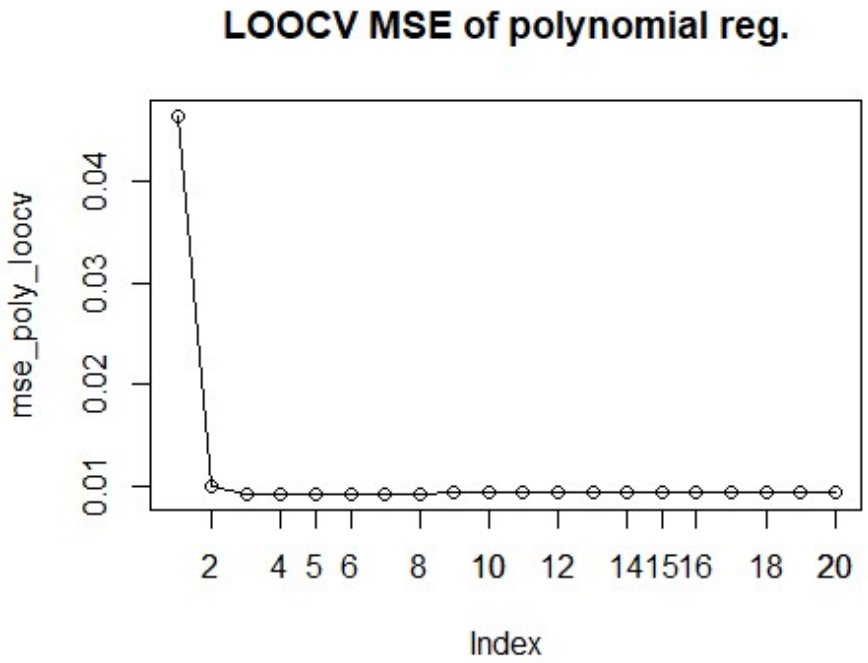
```
## validation method 를 이용해 구한 polynomial regression 의 차수는 5 이고 test MSE 는
0.01142621 이다.
```

```r
# fitted curve
plot(trainset[, 'x'], trainset[, 'y'], xlab='x', ylab='y',
     main='fitted curve of polynomial reg. degree=5 with validation')
points(x, predict(model_poly, newdata=x_linspace), col='red', type='l', lwd=3)
```



(2) LOOCV

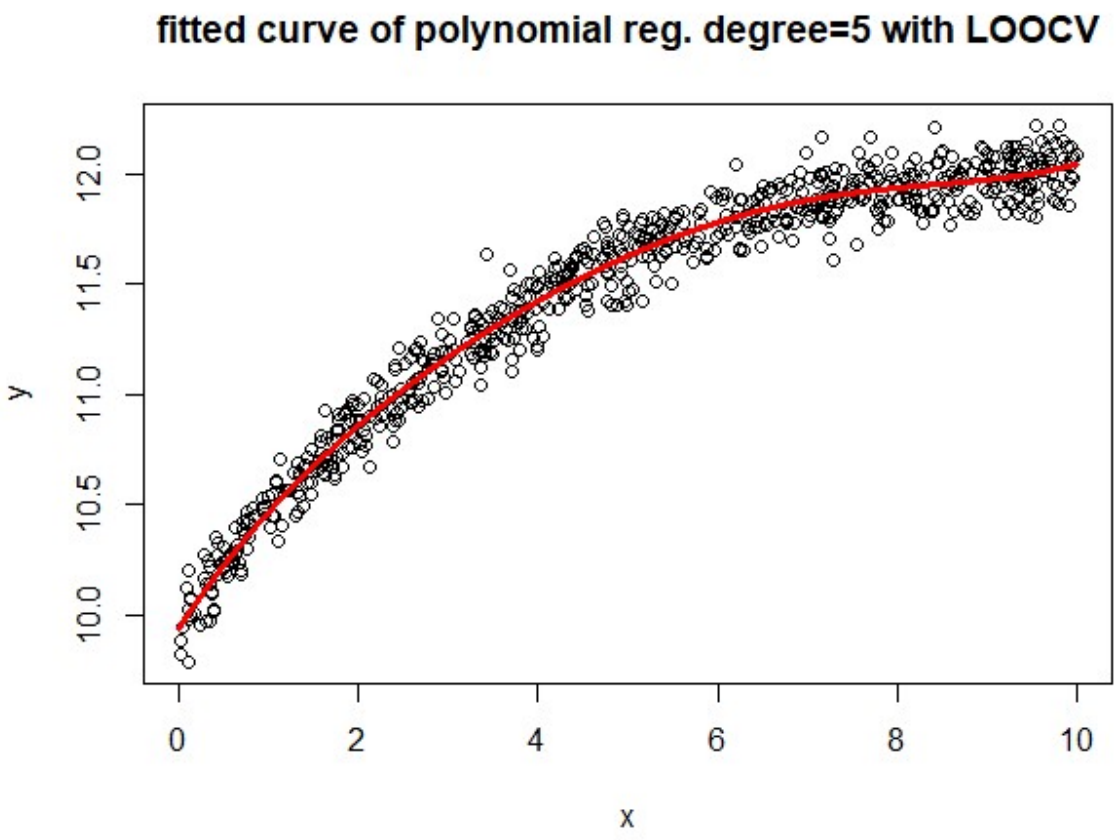
```
mse_poly_loocv = c()

# model learning
for (i in 1:20) {
  mse_loocv = 0

  for (j in 1:m) {
    temp_train = trainset[-j, ]
    temp_val = trainset[j, ]
    model_poly = lm(y ~ poly(x, i), data=temp_train)
    predict_value = predict(model_poly, newdata=temp_val)
    mse = (temp_val[, 'y'] - predict_value)^2

    mse_loocv = mse_loocv + mse
  }
  mse_poly_loocv[i] = mse_loocv / m
}

# 차수가 2 부터 validation mse 가 급격하게 감소하는 것을 확인할 수 있다.
plot(mse_poly_loocv, type='o', xlim=c(1, 20), main='LOOCV MSE of polynomial reg.')
axis(side=1, at=seq(0, 20, 2))
```



```r
# calculate test mse
degree = which.min(mse_poly_loocv)
model_poly = lm(y ~ poly(x, degree), data=trainset)
predict_value = predict(model_poly, newdata=testset)
test_mse_poly_loocv = mean((testset[, 'y'] - predict_value)^2)
cat('LOOCV method 를 이용해 구한 polynomial regression 의 차수는',
    degree, '이고 test MSE 는', test_mse_poly_loocv, '이다.')
```

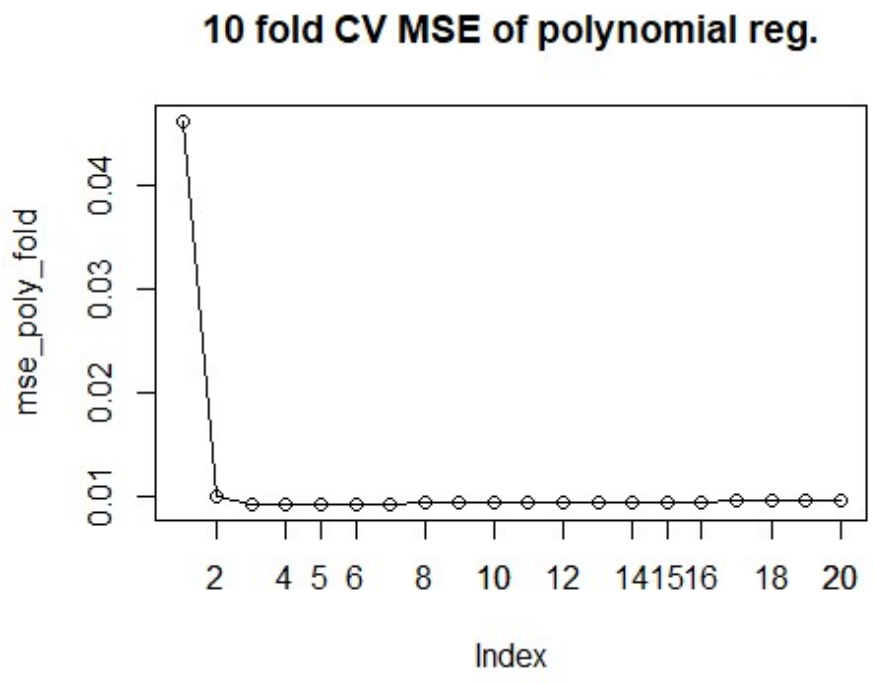
```
## LOOCV method 를 이용해 구한 polynomial regression 의 차수는 5 이고 test MSE 는 0.011
42621 이다.
```

```r
# fitted curve
plot(trainset[, 'x'], trainset[, 'y'], xlab='x', ylab='y',
     main='fitted curve of polynomial reg. degree=5 with LOOCV')
points(x, predict(model_poly, newdata=x_linspace), col='red', type='l', lwd=3)
```



(3) 10 fold cv

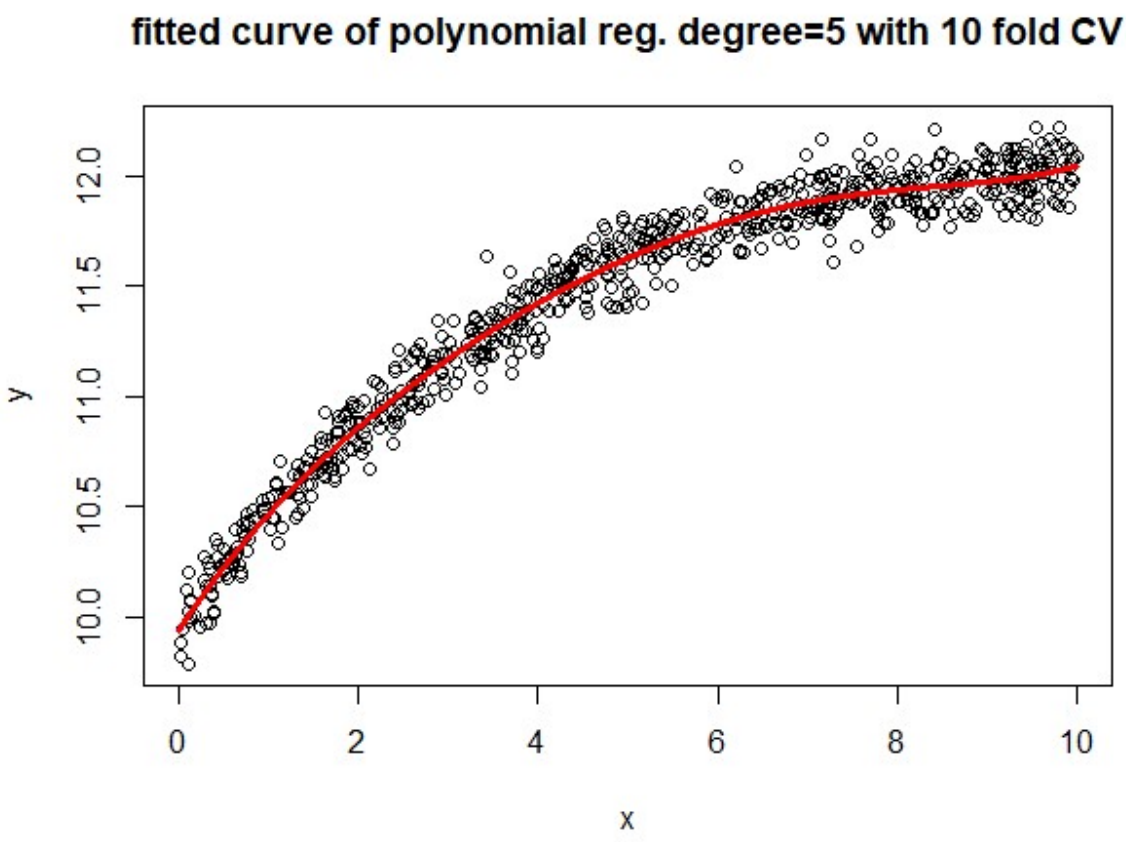
```
set.seed(student)
fold = 10
fold_index = sample(1:fold, m, replace=TRUE)

mse_poly_fold = c()

# model learning
for (i in 1:20) {
  mse_fold = 0

  for (j in 1:fold) {
    temp_train = trainset[fold_index!=j, ]
    temp_val = trainset[fold_index==j, ]
    model_poly = lm(y ~ poly(x, i), data=temp_train)
    predict_value = predict(model_poly, newdata=temp_val)
    mse = mean((temp_val[, 'y'] - predict_value)^2)

    mse_fold = mse_fold + mse
  }
  mse_poly_fold[i] = mse_fold / fold
}

# 차수가 2 부터 validation mse 가 급격하게 감소하는 것을 확인할 수 있다.
plot(mse_poly_fold, type='o', xlim=c(1, 20), main='10 fold CV MSE of polynomial re
g.')
axis(side=1, at=seq(0, 20, 2))
```



```r
# calculate test mse
degree = which.min(mse_poly_fold)
model_poly = lm(y ~ poly(x, degree), data=trainset)
predict_value = predict(model_poly, newdata=testset)
test_mse_poly_10foldcv = mean((testset[, 'y'] - predict_value)^2)
cat('10 fold CV method 를 이용해 구한 polynomial regression 의 차수는',
    degree, '이고 test MSE 는', test_mse_poly_10foldcv, '이다.')
```

```
## 10 fold CV method 를 이용해 구한 polynomial regression 의 차수는 5 이고 test MSE 는
0.01142621 이다.
```

```r
# fitted curve
plot(trainset[, 'x'], trainset[, 'y'], xlab='x', ylab='y',
     main='fitted curve of polynomial reg. degree=5 with 10 fold CV')
points(x, predict(model_poly, newdata=x_linspace), col='red', type='l', lwd=3)
```



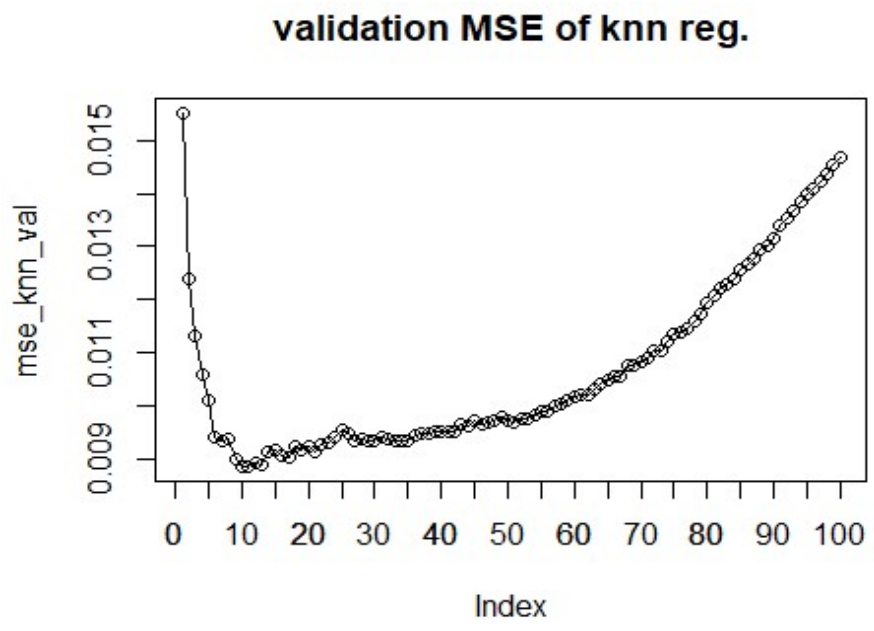
(b) KNN regression

Determine the best knn regression model to predict y by x. Draw a scatter plot with the fitted curve. Report the test MSE of your final model using the test set.

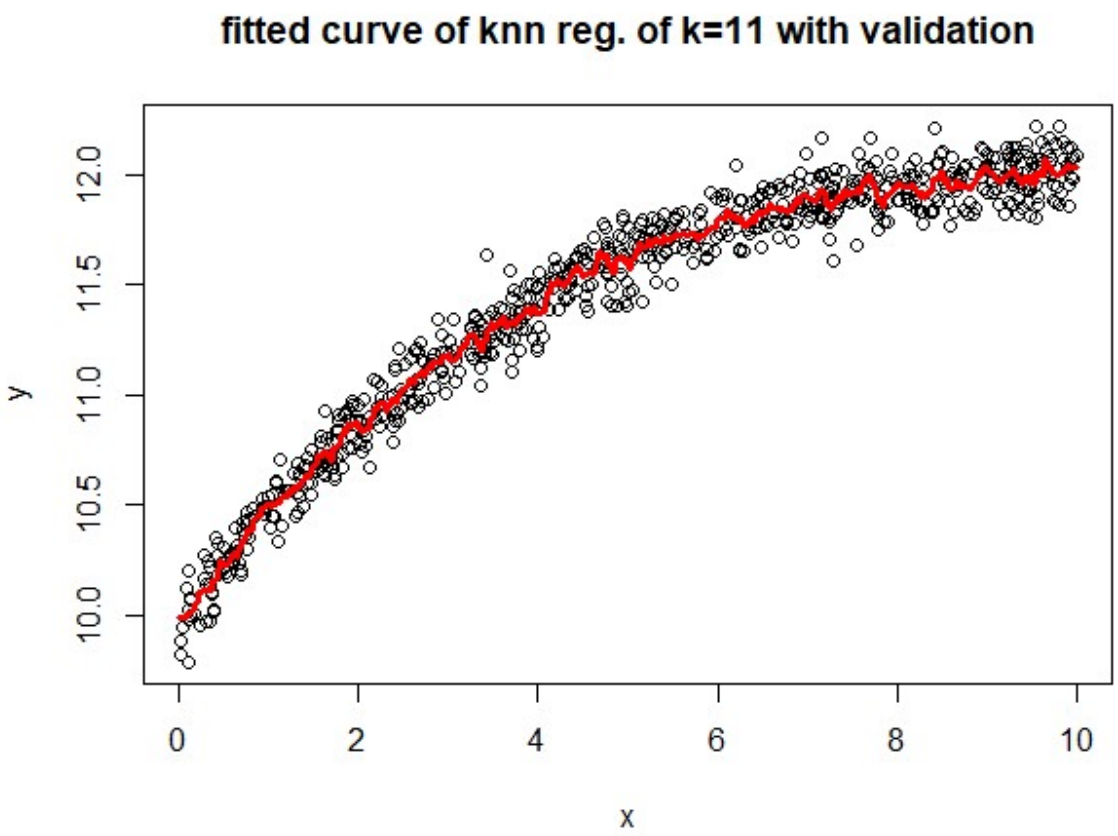
```
library(caret)
```

(1) validation set

```
# make validation set
set.seed(student)
m = dim(trainset)[1]
val_size = 0.3
val_index = sample(1:m, m*val_size, replace=FALSE)
train = trainset[-val_index, ]
val = trainset[val_index, ]

mse_knn_val = c()

# model learning
for (i in 1:100) {
  model_knn = knnreg(y ~ x, k=i, data=train)
  predict_value = predict(model_knn, newdata=val)
  mse = mean((val[, 'y'] - predict_value)^2)

  mse_knn_val[i] = mse
}

# k 가 증가하면서 val mse 가 감소하다가 10 근처부터 다시 증가하는 것을 확인할 수 있다.
plot(mse_knn_val, type='o', xlim=c(1, 100), main='validation MSE of knn reg.')
axis(side=1, at=seq(0, 100, 5))
```

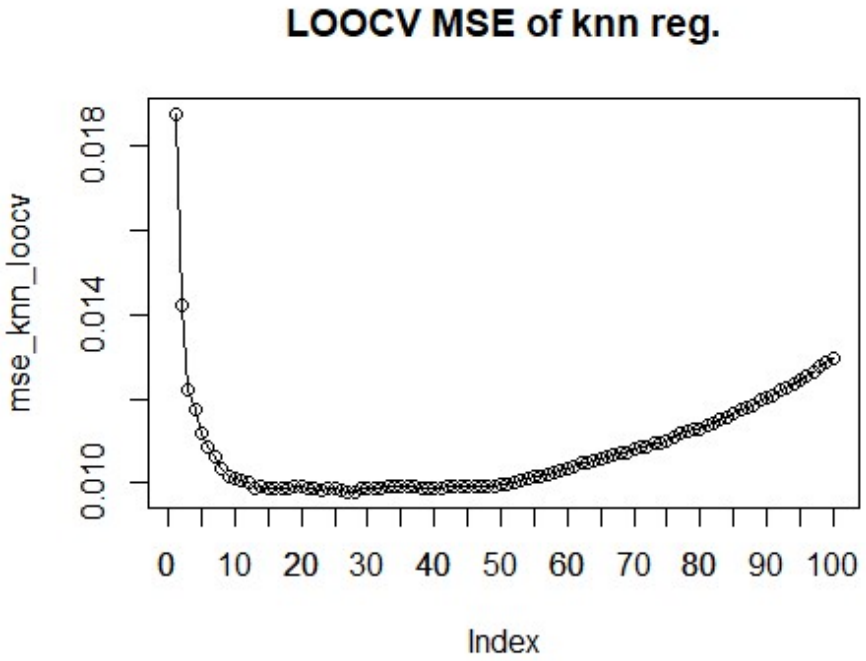


```r
# calculate test mse
k = which.min(mse_knn_val)
model_knn = knnreg(y ~ x, k=k, data=trainset)
predict_value = predict(model_knn, newdata=testset)
test_mse_knn_val = mean((testset[, 'y'] - predict_value)^2)
cat('validation method 를 이용해 구한 k 는',
    k, '이고 test MSE 는', test_mse_knn_val, '이다.')

## validation method 를 이용해 구한 k 는 11 이고 test MSE 는 0.01234136 이다.

# fitted curve
plot(trainset[, 'x'], trainset[, 'y'], xlab='x', ylab='y',
     main='fitted curve of knn reg of k=11 with validation')
points(x, predict(model_knn, newdata=x_linspace), col='red', type='l', lwd=3)
```

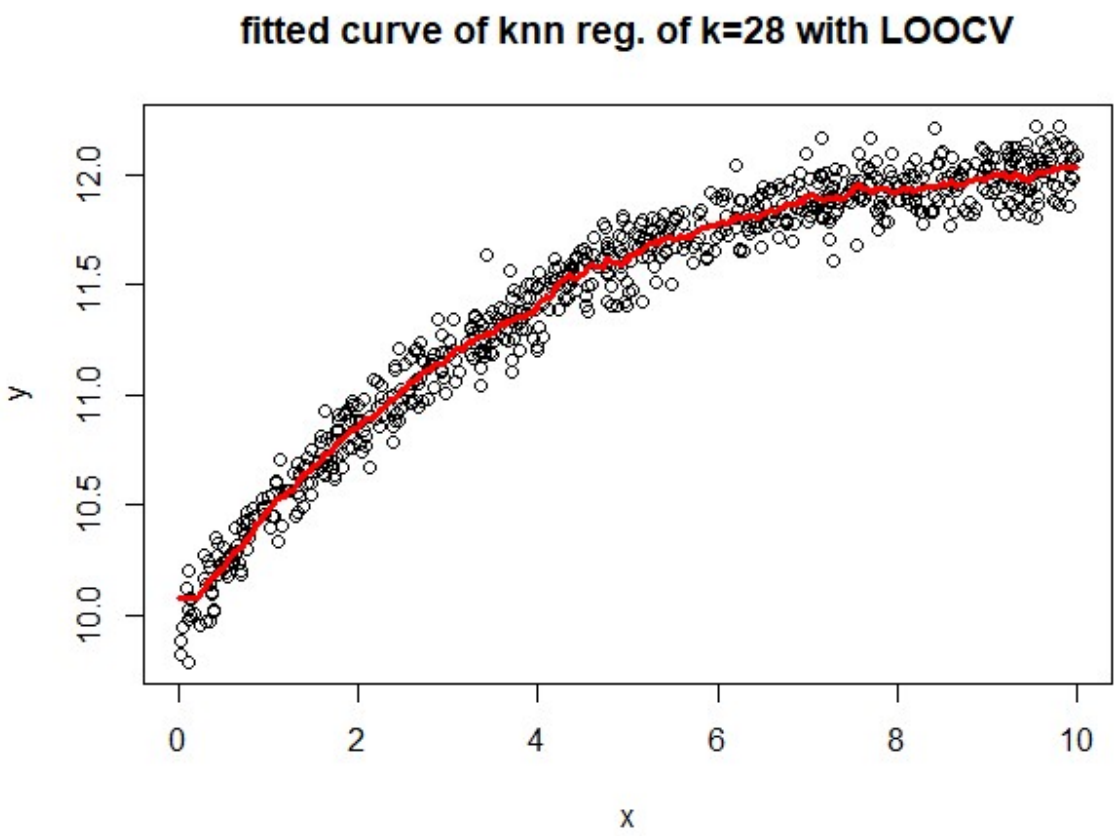


(2) LOOCV

```
mse_knn_loocv = c()

# model learning
for (i in 1:100) {
  mse_loocv = 0

  for (j in 1:m) {
    temp_train = trainset[-j, ]
    temp_val = trainset[j, ]
    model_knn = knnreg(y ~ x, k=i, data=temp_train)
    predict_value = predict(model_knn, newdata=temp_val)
    mse = (temp_val[, 'y'] - predict_value)^2

    mse_loocv = mse_loocv + mse
  }
  mse_knn_loocv[i] = mse_loocv / m
}

# k 가 감소하다가 50 을 지나면서부터 증가하는 것을 확인할 수 있다.
plot(mse_knn_loocv, type='o', xlim=c(1, 100), main='LOOCV MSE of knn reg.')
axis(side=1, at=seq(0, 100, 5))
```

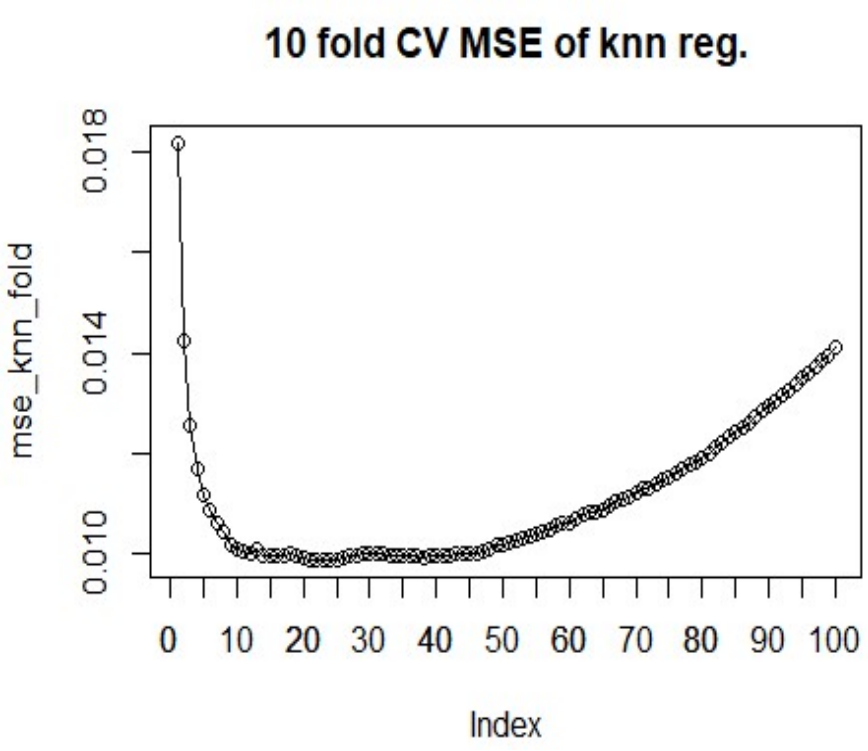


```r
# calculate test mse
k = which.min(mse_knn_loocv)
model_knn = knnreg(y ~ x, k=k, data=trainset)
predict_value = predict(model_knn, newdata=testset)
test_mse_knn_loocv = mean((testset[, 'y'] - predict_value)^2)
cat('validation method 를 이용해 구한 k 는',
    k, '이고 test MSE 는', test_mse_knn_loocv, '이다.')

## validation method 를 이용해 구한 k 는 28 이고 test MSE 는 0.01171555 이다.

# fitted curve
plot(trainset[, 'x'], trainset[, 'y'], xlab='x', ylab='y',
     main='fitted curve of knn reg of k=28 with LOOCV')
points(x, predict(model_knn, newdata=x_linspace), col='red', type='l', lwd=3)
```



(3) 10 fold cv

```
set.seed(student)
fold = 10
fold_index = sample(1:fold, m, replace=TRUE)

mse_knn_fold = c()

# model learning
for (i in 1:100) {
  mse_fold = 0

  for (j in 1:fold) {
    temp_train = trainset[fold_index!=j, ]
    temp_val = trainset[fold_index==j, ]
    model_knn = knnreg(y ~ x, k=i, data=temp_train)
    predict_value = predict(model_knn, newdata=temp_val)
    mse = mean((temp_val[, 'y'] - predict_value)^2)

    mse_fold = mse_fold + mse
  }
  mse_knn_fold[i] = mse_fold / fold
}

# k 가 감소하다가 50 을 지나면서부터 증가하는 것을 확인할 수 있다.
plot(mse_knn_fold, type='o', xlim=c(1, 100), main='10 fold CV MSE of knn reg.')
axis(side=1, at=seq(0, 100, 5))
```



```r
# calculate test mse
k = which.min(mse_knn_fold)
model_knn = knnreg(y ~ x, k=k, data=trainset)
predict_value = predict(model_knn, newdata=testset)
test_mse_knn_10foldcv = mean((testset[, 'y'] - predict_value)^2)
cat('validation method 를 이용해 구한 k 는',
    k, '이고 test MSE 는', test_mse_knn_10foldcv, '이다.')

## validation method 를 이용해 구한 k 는 25 이고 test MSE 는 0.01172869 이다.

# fitted curve
plot(trainset[, 'x'], trainset[, 'y'], xlab='x', ylab='y',
     main='fitted curve of knn reg of k=25 with 10 fold CV')
points(x, predict(model_knn, newdata=x_linspace), col='red', type='l', lwd=3)
```