hw1

Keyi Zhong

4/12/2020

#A

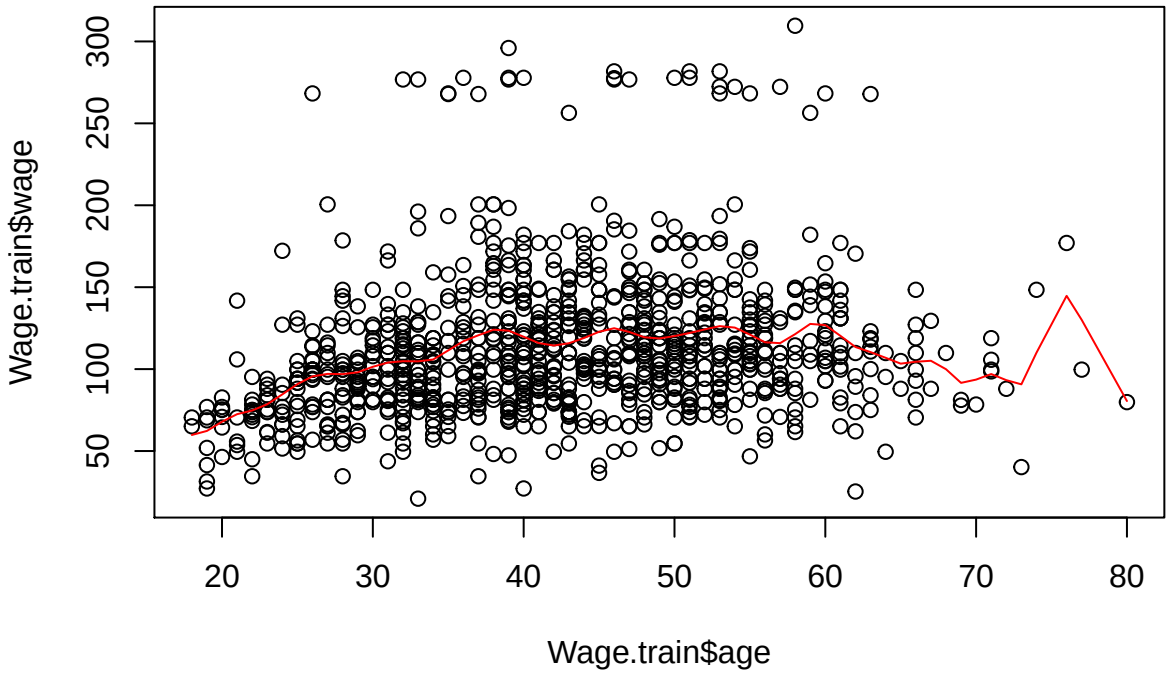
```
source('home1-part1-data.R')
Wage.test <- Wage.test[order(Wage.test$age),]
Wage.train <- Wage.train[order(Wage.train$age),]
## A
ksmooth.train <- function(x.train, y.train, kernel = c("box", "normal"),bandwidth = 0.5, CV = FALSE){
  ord <- order(x.train)
  x <- x.train[ord]
  y <- y.train[ord]
  if(kernel == 'box') {
    yhat.train <- rep(0,length(x))
    for (i in 1:length(x)) {
      if(CV) {
        yhat <- mean(y[-i][abs(x[i]-x[-i])<=bandwidth])
      }else {
        yhat <- mean(y[abs(x[i]-x)<=bandwidth])
      }
      yhat.train[i] <- yhat
    }
    return(list(x=x,y=yhat.train))
  }else {
    yhat.train <- rep(0,length(x))
    gaussian_sd <- -bandwidth/4/qnorm(0.25)
    for (i in 1:length(x)) {
      if(CV) {
        gaussian_sum <- sum(dnorm(x[i]-x[-i], 0, gaussian_sd))
        yhat <- sum(y[-i]*dnorm(x[i]-x[-i], 0, gaussian_sd))/gaussian_sum
      }else {
        gaussian_sum <- sum(dnorm(x[i]-x, 0, gaussian_sd))
        yhat <- sum(y*dnorm(x[i]-x, 0, gaussian_sd))/gaussian_sum
      }
      yhat.train[i] <- yhat
    }
    return(list(x=x,y=yhat.train))
  }
}
```

#B

```
ksmooth.predict <- function(ksmooth.train.out, x.query) {
  x <- x.query[order(x.query)]
  y.predict <- approxfun(ksmooth.train.out,rule = 2)(x)
  return(y.predict)
}
```

#C

```
plot(Wage.train$age, Wage.train$wage)
trained <- ksmooth.train(Wage.train$age, Wage.train$wage,'normal',3)
x <- trained$x
y <- trained$y
lines(trained,col='red')
```

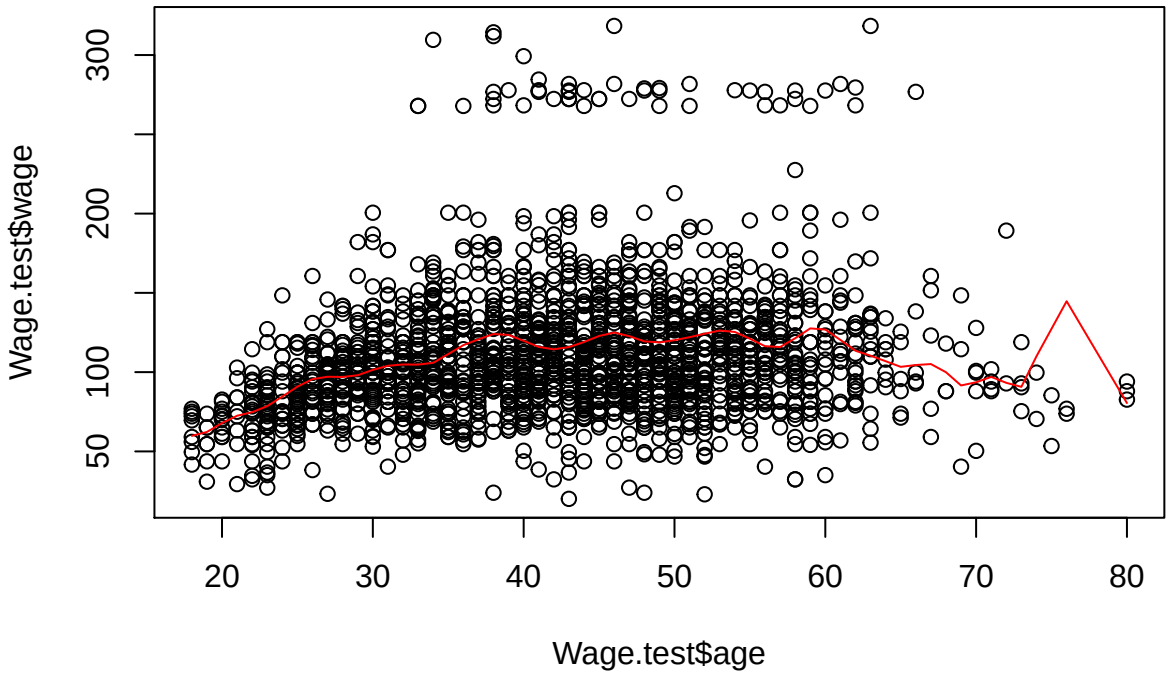



```
sum <- 0
for(i in 1:length(x)) {
  sum <- sum + (Wage.train$wage[order(Wage.train$age)][i]-y[i])^2
}
sum
```

```
## [1] 1625121
```

#D

```
trained <- ksmooth.train(Wage.train$age, Wage.train$wage,'normal',3)
test.predict <- ksmooth.predict(trained, Wage.test$age)
plot(Wage.test$age,Wage.test$wage)
lines(Wage.test$age[order(Wage.test$age)] ,test.predict, col = 'red')
```




```
sum <- 0
for(i in 1:length(Wage.test$age)) {
  sum <- sum + (Wage.test$wage[order(Wage.test$age)][i]-test.predict[i])^2
}
sum
```

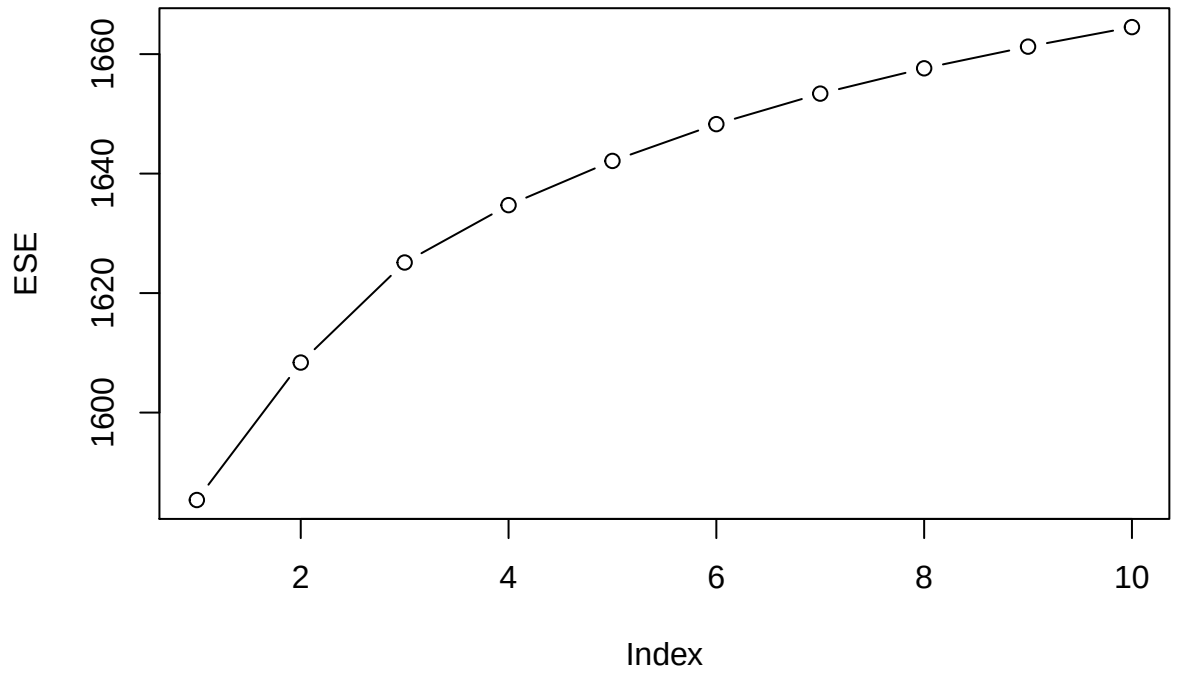
```
## [1] 3168000
```

#E

```
ESE <- rep(0,10)
for(i in 1:10) {
  yhat <- ksmooth.train(Wage.train$age,Wage.train$wage,'normal',i)$y
  ESE[i] <- mean((Wage.train$wage[order(Wage.train$age)]-yhat)^2)
}
ESE
```

```
##  [1] 1585.364 1608.370 1625.121 1634.722 1642.120 1648.282 1653.387 1657.624
##  [9] 1661.252 1664.519
```

```
plot(ESE,type = 'b')
```

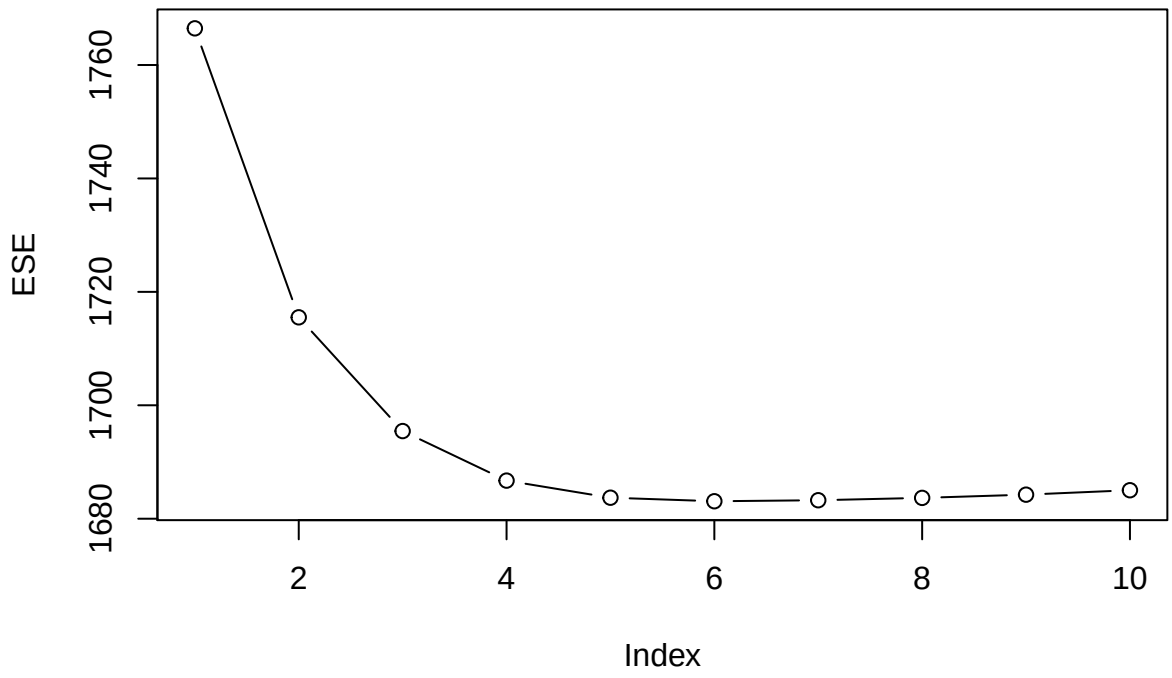


#F

```
ESE <- rep(0,10)
for(i in 1:10) {
  fhat <- ksmooth.train(Wage.train$age,Wage.train$wage,'normal',i,TRUE)$y
  ESE[i] <- mean((Wage.train$wage[order(Wage.train$age)]-fhat)^2)
}
ESE
```

```
##  [1] 1766.466 1715.508 1695.453 1686.715 1683.705 1683.079 1683.256 1683.671
##  [9] 1684.239 1685.021
```

```
plot(ESE,type = 'b')
```

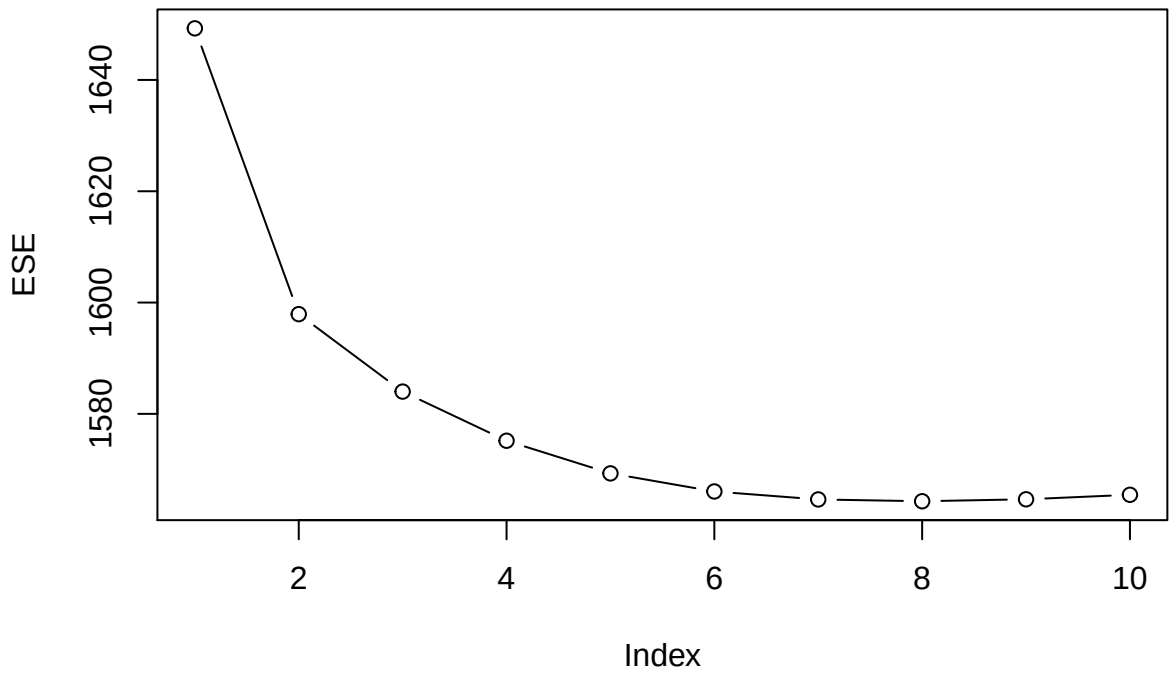


#G

```
ESE <- rep(0,10)
for(i in 1:10) {
  fhat <- ksmooth.predict(ksmooth.train(Wage.train$age,Wage.train$wage,'normal',i), Wage.test$age)
  ESE[i] <- mean((Wage.test$wage[order(Wage.test$age)]-fhat)^2)
}
ESE
```

```
##  [1] 1649.268 1597.913 1584.000 1575.168 1569.304 1566.052 1564.621 1564.297
##  [9] 1564.647 1565.453
```

```
plot(ESE,type = 'b')
```

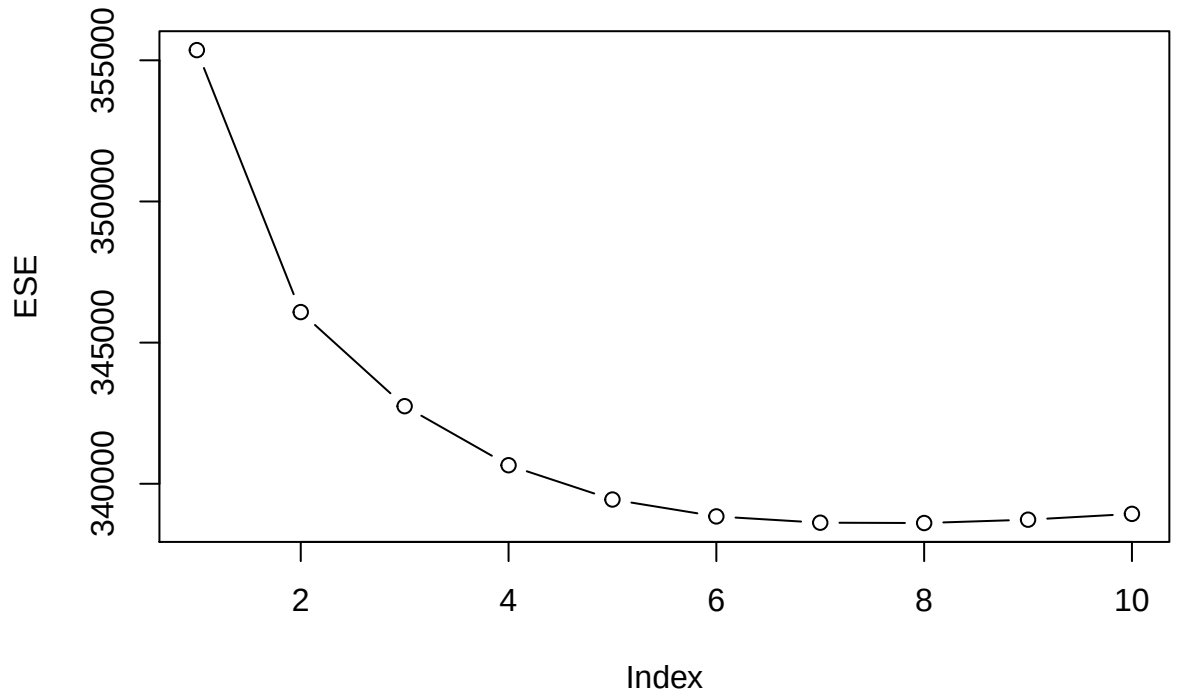


#H

```
ESE <- rep(0,10)
for(i in 1:10) {
  RSS <- 0
  for(j in 1:5) {
    train.age <- Wage.train$age[fold != j]
    train.wage <- Wage.train$wage[fold != j]
    trained <- ksmooth.train(train.age, train.wage, 'normal', i)
    test.age <- Wage.train$age[fold == j]
    test.wage <- Wage.train$wage[fold == j]
    fhat <- ksmooth.predict(trained, test.age)
    RSS <- RSS + sum((test.wage[order(test.age)] - fhat)^2)
  }
  ESE[i] <- RSS / 5
}
ESE
```

```
##  [1] 355360.8 346081.7 342748.4 340657.0 339442.5 338844.3 338621.2 338610.3
##  [9] 338727.6 338933.7
```

```r
plot(ESE,type = 'b')
```



#Part2 A

$D = E(\frac{1}{n}||f - \hat{f}||^2)$

Let R be $(\epsilon_1, \epsilon_2...)$

$D = E(\frac{1}{n}||f - Wy||^2)$

$D = \frac{1}{n}E(||f - W(f + R)||^2)$

$D = \frac{1}{n}E(||(I - W)f - WR||)$

$V(\epsilon_i) = \sigma^2$ and $E(\epsilon_i) = 0$ sp $||E||^2 = \sigma^2$

$||WE||^2 = \sum_i \sum_j \epsilon_i^2 w_{ij}^2 = \sigma^2 \sum_i \sum_j w_{ij}^2$

$D = \frac{1}{n}(||(W - I)f||^2 + \sigma^2 trace(W^T W))$

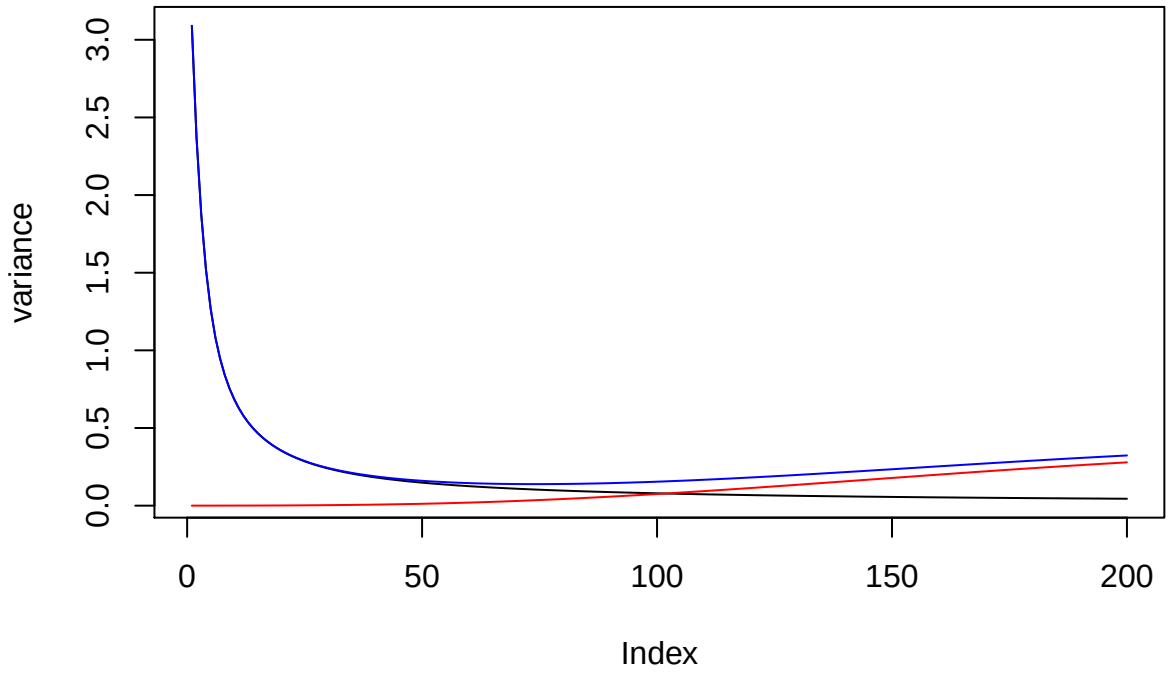
#Part2 B

```r
source('home1-part2-data.R')
squared.bias <- rep(0,200)
variance <- rep(0,200)
sds <- seq(0.01,2,by=0.01)
weights <- matrix(nrow = length(x.train),ncol=length(x.train))
for(i in 1:200) {
  for(j in 1:200) {
    gaussian_sum <- sum(dnorm(x.train[j]-x.train, 0, sds[i]))
```

```
    weight <- dnorm(x.train[j]-x.train, 0, sds[i])/gaussian_sum
    variance[i] <- variance[i] + sum(weight^2)
    weight[j] <- weight[j] - 1
    squared.bias[i] <- squared.bias[i] + abs(sum(weight*f))^2
  }
  squared.bias[i] <- (squared.bias[i])/200
  variance[i] <- noise.var*variance[i]/200
}
sums <- variance + squared.bias
sds[which(sums==min(sums))]
```

```
## [1] 0.74
```

```
plot(variance,type='l')
lines(squared.bias,col='red')
lines(sums,col='blue')
```



#Part2 C

```
sigma <- which(sums==min(sums))*0.01
fhat <- rep(0,200)
for(i in 1:200) {
  gaussian_sum <- sum(dnorm(x.train[i]-x.train, 0, sigma))
  yhat <- sum(y.train*dnorm(x.train[i]-x.train, 0, sigma))/gaussian_sum
  fhat[i] <- yhat
```

```
}
plot(x.train,y.train)
lines(x.train, f,col='blue')
lines(x.train,fhat,col='red')


legend("topleft",
       c("train","fhat", "f"),
       fill=c("black","red", "blue")
)
```