K-nn Regresion Assignment

Ignasi Mañé & Antoni Company

02/09/2019

The k nearest neighbor estimator of $m(t) = E(Y|X = t)$ is defined as

$$\hat{m}(t) = \frac{1}{k} \sum_{i \in N_k(t)} y_i$$

where $N_k(t)$ is the neighborhood of t defined by the k closest points x_i in the training sample.

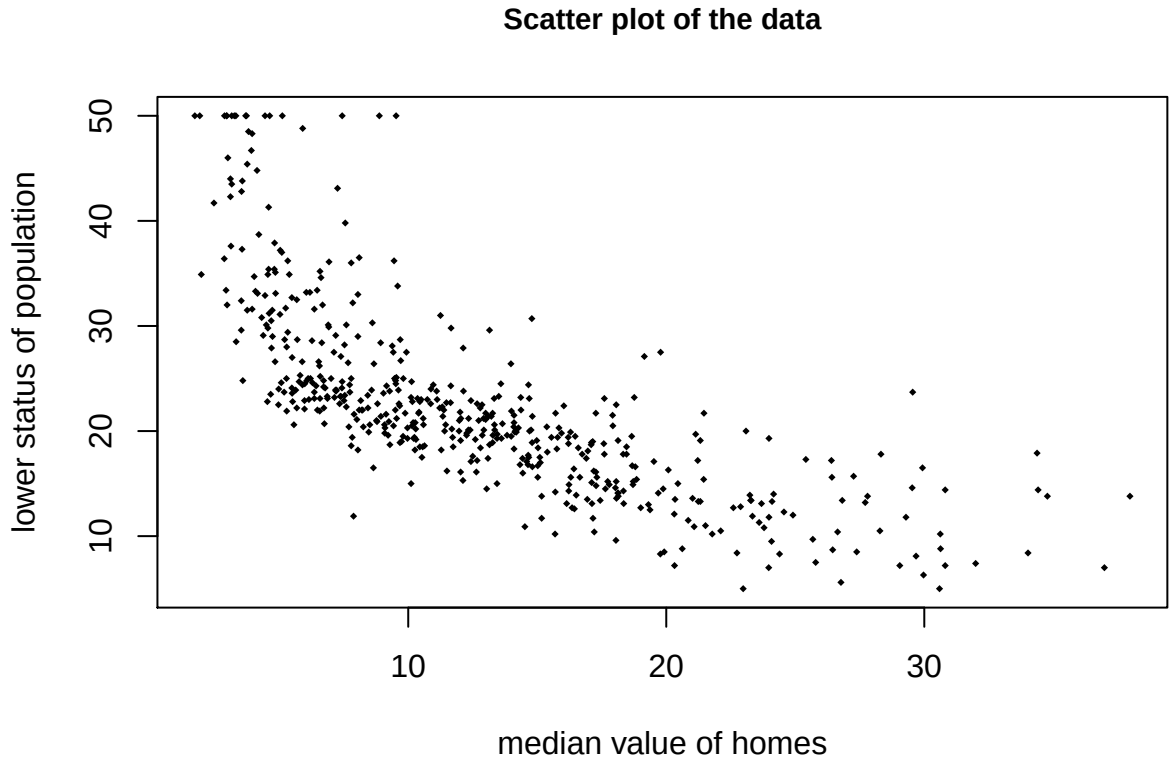
In this exercice we will use the knn estimator along with the Boston data set from the MASS package to express the response variable "medv" (median value of owner-occupied homes in $1000s) as a function of the covariate "lstat" (lower status of the population).

Fist, we load the Boston data set and create a data frame named df.xy containing the variables lstat and medv.

```
library(MASS)
data("Boston")
df.xy <- data.frame(x=Boston$lstat,y=Boston$medv)
```

Second, we plot both variables using a scatter plot to visualize how they are related.

```
plot(x=Boston$lstat,y=Boston$medv,col=1,pch=18, asp = 0, cex=.5,
     xlab="median value of homes",
     ylab = "lower status of population",
     main="Scatter plot of the data", cex.main=0.9)
```



1. Write a function that computes the k-nn estimator of m(t) for a given value of t R.

The function that computes the k-nn estimator of m(t) is:

```
#Knn estimator
knn.medv <- function (t, k, data) {
  dist.t.x <- abs(data[,1]-t) #distance fromn each point to point t
  dist.t.x.k <- sort(dist.t.x)[k] #we get the k nearest distance after soarting the distance vector
  lstat.under.k <- which(dist.t.x<=dist.t.x.k) #which data points are within the radius=dist.t.x.k
  medv.estim <- sum(data[lstat.under.k,2])/length(data[lstat.under.k,2])
  #we compute the average to estimate the value of E(Y|X=x)
  return(medv.estim) #retourn the estimated value
  }
```

2. Then, define t as a sequence form 1 to 40: t <- 1:40.

```
t <- seq(1,40,by=1)
```

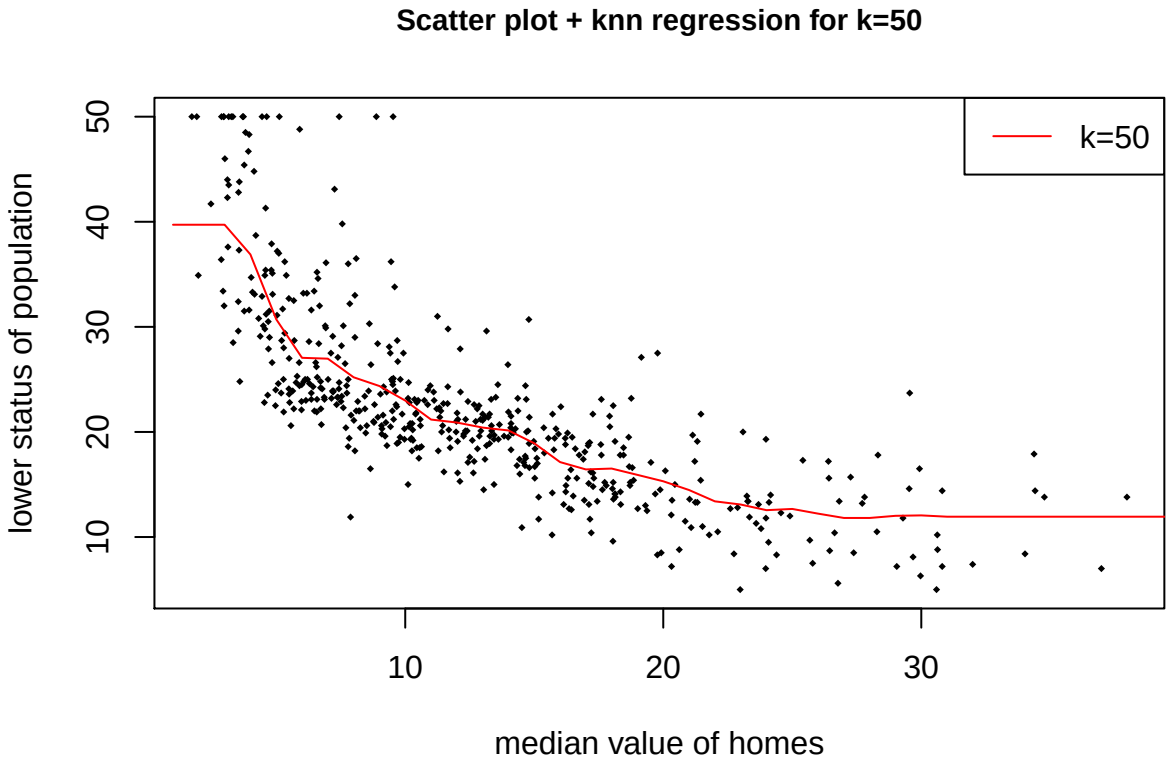
3. Estimate m(t[i]) for i = 1, . . . , 40 using k = 50.

We call the function knn.medv with paraameter k=50 for every value i = {1,. . . ,40} and store the results in the vector estim.medv.

```
estim.medv <- c()
for (i in 1:40){
  estim.medv[i] <- knn.medv(t = i,k = 50 ,data = df.xy)
}
```

4. Plot y against x. Then represent the estimated regression function.

```
plot(Boston$lstat,Boston$medv,col=1,pch=18, asp = 0, cex=.5,
     xlab="median value of homes",
     ylab = "lower status of population",
     main="Scatter plot + knn regression for k=50", cex.main=0.9)
lines(x=t,y=estim.medv,col="red")
legend("topright",legend=paste("k=",50,sep = ""),lty = c(1), col=c("red"))
```

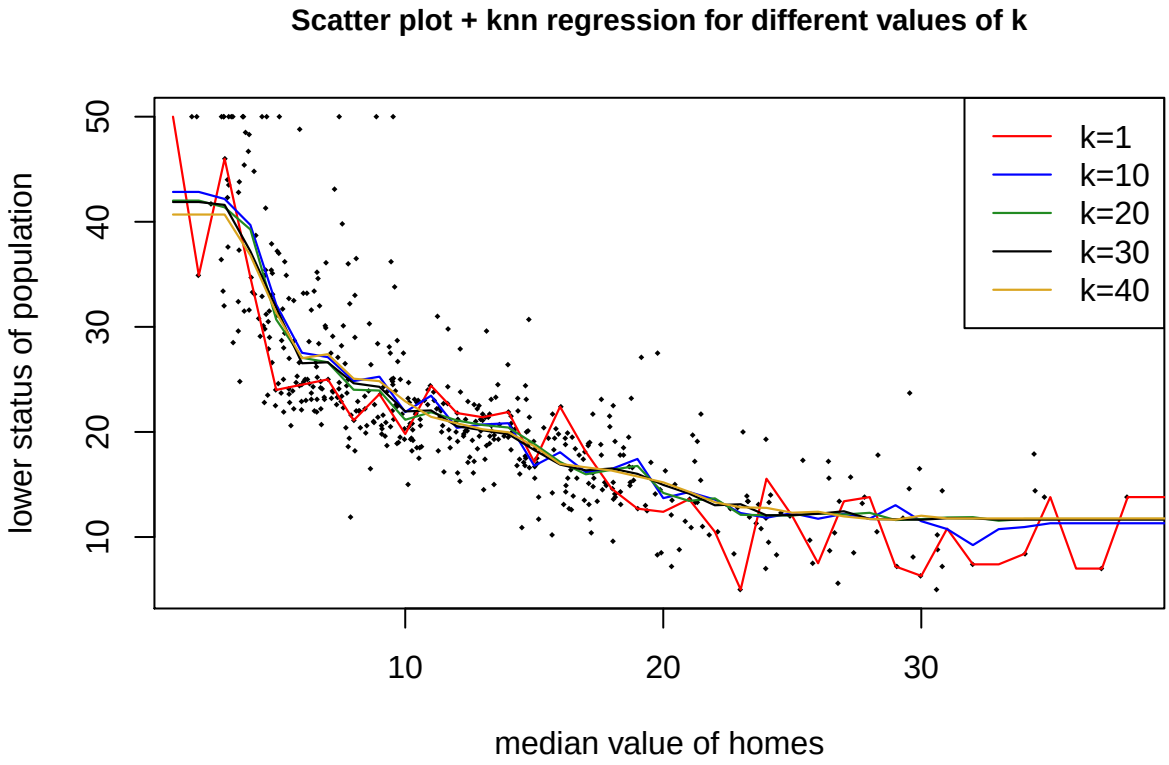


5. Repeat the same exercise using different values of k.

```
k.1 <- c(1,seq(10,40,by=10))
t.1 <- seq(1,40,by=1)
estim.medv.m <- matrix(0,nrow = length(t.1),ncol = length(k.1))

for (k in 1:length(k.1)){
  for(t in t.1){
    estim.medv.m[t,k] <- knn.medv(t = t,k = k.1[k], data = df.xy)
  }
}

plot(Boston$lstat,Boston$medv,col=1,pch=18, asp = 0, cex=.4,
     xlab="median value of homes",
     ylab = "lower status of population",
     main="Scatter plot + knn regression for different values of k", cex.main=0.9)
matplot(estim.medv.m,type="l",add = TRUE,
        col=c("red","blue","forestgreen","gray0","goldenrod"), lty = c(1),
        lwd=c(1.1))
legend("topright",legend=paste("k=",k.1,sep = ""),
       col=c("red","blue","forestgreen","gray0","goldenrod"), lty = c(1),
       lwd=c(1.1))
```



As we can see in the chart, it seems that the greater the value of k is the less variability the estimated regression function has.