MS-C1620 Statistical inference: Exercise set 12

1 Homework

To be solved at home before the exercise session

1. The dataset `cps71` from the package `np` contains ages and logarithmic wages of Canadian high school graduates.

(a) Try Nadaraya-Watson kernel regression with a normal kernel, with various bandwidth values and plot them. Decide, by visual inspection, which bandwidth you think best captures the meaningful features of the data. (Hint: You could first try h=1 and h=100.)

```
install.packages("np")
library(np)
library(KernSmooth)
data(cps71)
x <- cps71$age
y <- cps71$logwage
h <- 1
K <- ksmooth(x, y, kernel="normal", bandwidth=h)
plot(x, y)
lines(K)
```

(b) Perform cross-validation to choose the optimal bandwidth. Try 60 different bandwidths $0.5, 1.0, 1.5, 2.0, \ldots, 30.0$.

For starters you can take the cross-validation code from Lecture 11, but modify it to do leave-one-out cross-validation (instead of splitting the data half and half). That is, from the $n = 205$ data points, take each point in turn as the "test point", and fit the kernel regression model to the other 204 points. Average the squared error over the 205 points to get the MSE value. Find bandwidth that minimizes MSE.

Hint: If `j` is the index of the test point, then `x[-j]` gives all other points.

2 Class exercse

To be solved at the exercise session

2. Using the cps71 data, try local polynomial fits of degrees 0, 1 and 2. Since `ksmooth` does not support degrees other than zero, use now the function `locpoly`.

Use a bandwidth that corresponds to what you chose in 1b, but note that `locpoly` uses a different scale. In `ksmooth` bandwidth is measured as two times the interquartile range. Numerically this equals 2.7 standard deviations (in the normal distribution). In `locpoly` bandwidth is simply the standard deviation. This means that in `locpoly` you have to specify 1/2.7 of the value in `ksmooth` to get the same level of smoothing!

To get some idea of how the different fits would extrapolate, start with the following

```
xrange <- c(15,70)
plot(x, y, xlim=xrange)
```

and also specify `range.x=xrange` in `locpoly`. Compare the different fits (i) well inside the data range, (ii) near the ends of the data range, and (ii) outside the data range. Which of the fits seems best? How do they differ in extrapolation?

3. Take a random resample of the `cps71` data, of size $n = 205$, with replacement (just like in bootstrapping). Use local linear (degree=0) regression using `locpoly` on the resampled data and plot it. Do this 20 times, with the plots overlayed in the same picture using `lines`. This gives you a rough idea of how the result of the kernel regression is affected by the randomness in the data. What do you see?

4. *Optional exercise, requires R coding, slightly difficult.*

Refining problem 3. Do the resampling 1000 times. For each x point where the regression function is evaluated, collect the 1000 different values of the regression function, and then choose three quantiles (0.025, 0.500 and 0.975) from this set of 1000 values. The lower and upper values define in fact a 95% bootstrap confidence interval of the regression function (at that particular x value).

Plot three curves: One connecting the 0.025 quantiles (for all x), one connecting the 0.500 quantiles, and one connecting the 0.975 quantiles. The lower and upper curves are (pointwise) 95% *confidence bands* for the regression function.

Look critically at the curves. Where is the confidence band wider? What does this tell you about how reliable the regression function is? What do you think is causing the variations in the width?

Hint: Read the documentation of `locpoly`, and use the `gridsize` and `range.x` parameters. Note that `locpoly` calculates the regression function at equally spaced grid points, not at the original x points of the data. Use the same grid every time, for example 401 points from 21 to 65 (the data range).

For extra exercise, you can do this whole thing with various h values, try e.g. h=2 and h=100.