Machine learning II
Master Data Science
Winter Semester 2019/20

BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Prof. Tim Downie

Workshop 3
Non-Linear Regression Models II

Start R in the usual way.
In the 2nd section you will be using functions from the package `gam` and `akima` which you will probably need to install. You will also use the package `ISLR`, but if you are up-to-date with the workshops, this will already be installed. The other packages used `MASS` and `splines` were downloaded when R was installed.

Download there two script files `loess1.r` and `loess2.r` from Moodle, which contain template code for Section 1.

1 Local regression

The code in first script file `loess1.r` uses local regression to estimate the linear loess value at a specific point x_0. The code in `loess2.r` does the same over a given grid of x-values

Last week you used non-linear smoothing on the motorcycle helmet acceleration data `mcycle` in the MASS library. As a reminder, recreate the spline smoothing estimate using

```
library(MASS)
library(splines)
plot(mcycle)
fit1=smooth.spline(mcycle$times,mcycle$accel,df=10)
x.grid<-0:56
preds=predict(fit1,x.grid)
lines(x.grid,preds$y,lwd=2,col="black")
```

Use the code in `loess1.r` to obtain the local regression line at the point x_0=28 for the motorcycle data. You will need to complete some of the syntax. Once the code is working, try with span=0.75 and again with x_0=14

The code in `loess2.r` evaluates the whole loess curve over a given x-grid to get a complete loess curve.

Use the *R*-command `loess()` to replicate your algorithm in in one step. Because you have fitted a *linear* local regression, you need to specify the argument `degree=1` in the `loess` function call. The loess() function returns an list of class loess, which contains a vector called $fitted. These are the loess curve values evaluated at each of the data points. The predict() function is used to calculate the predicted values for any specified x-value(s).

2 Generalised additive Models

The Work data

Work through Lab 7.8.3 in James et al. starting on page 294, up to the command plot(gam.lo.i) on page 296.

The first example uses lm() to fit the model. Check that the function gam() with the same arguments outputs the same coefficients. Hints: if you have not already done so you need to install and start the gam package, and to obtain the coefficients of a statistical model use the function coef()
Important! The function plot.gam is now called plot.Gam

The College data

The College dataset is available in the ISLR package, the accompanying package to James et al.
An introduction to the data set is given on page 55.

We will fit the variable Accept, the number of applications accepted, as the outcome variable. Obtain the mean and median and a histogram of Accept. What do you notice about the distribution of these data?

Fit a first model with Private (a yes/no factor variable) and Apps as a smoothing spline using 5 effective degrees of freedom.

Both variables are highly significant (summary()), but Accept and Apps have very skewed distributions. Repeat the GAM fitting with the logarithm of both of these variables.

Continue the analysis by adding in turn each of the following variables using spline smoothing, and using the anova() function to see if this variable gives a better model fit.
F.Undergrad, Room.Board, Expend, PhD, and S.F.Ratio.

Obtain the response plot for each of the fitted variables in your final including the partial residuals (resid=TRUE).

Use you final model to obtain a prediction for Accept at *Harvard University* and compare this with the observed value, NB remember that the logarithm of accept was fitted in the model.