Advanced Statistical Modeling

Non-parametric models - Smoothing splines

Haoran Mo, Alexandra Yamaui

December 28th, 2017

In this excercise we will use a family of nonparametric estimators for the regression function $m(x)$, that express the estimation problem as an optimization problem. One member of this family of estimators are the Splines, which are based in the penalized least squares nonparametric regression, which penalizes the lack of smoothness. The most common ones are the cubic splines, with degree 3.

We are going to use the `meat` data set, which contains information of 240 samples of chopped meat that have been analyzed by two methods: a chemical procedure and a near infrared (NIR) absorbance spectroscopy. Because the chemical analysis is expensive and time consuming we want to predict the fat content from the spectral data. There are three main variables Fat: Fat content (in percentage) as it was determined buy the chemical analysis. abs.850: Near-infrared absorbance spectra for 850 nm wavelength. abs.957: Near-infrared absorbance spectra for 975 nm wavelength.

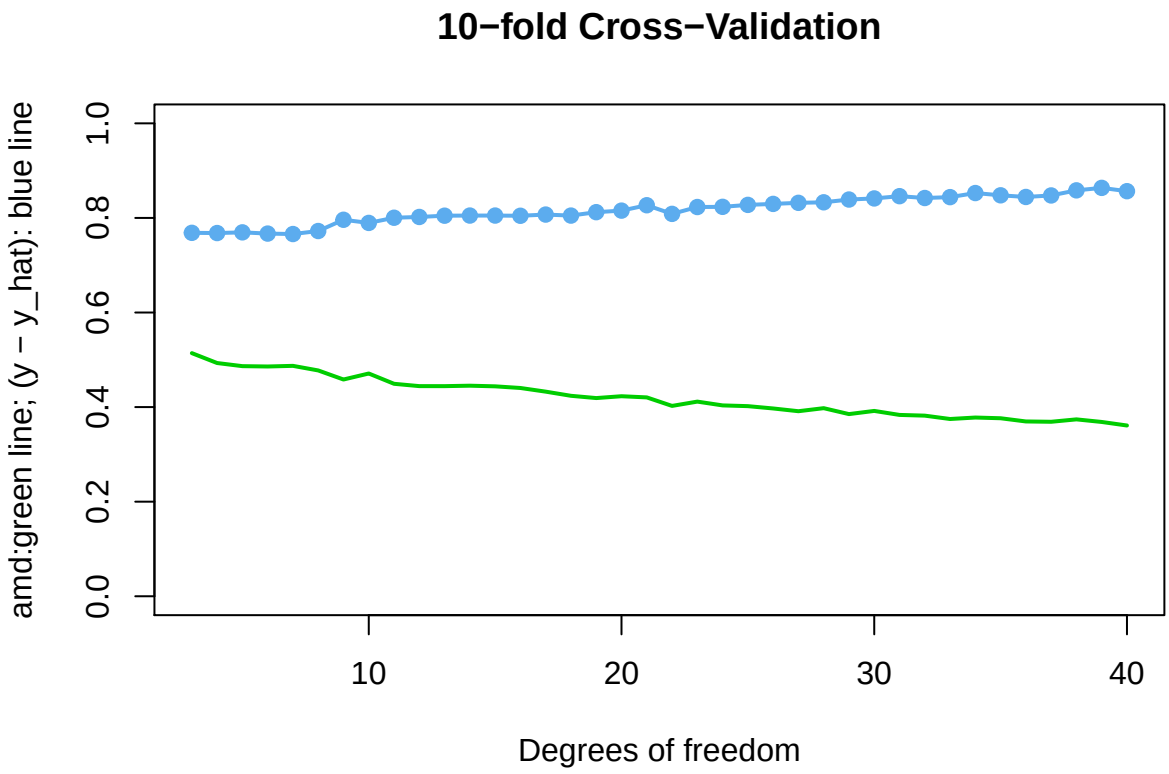
We are going to use a linear model to predict the log(Fat) content as a function of `abs.850` using cubic (degree 3) B-spline basis. To generate this basis matrix we need to determine the right number of knots or degrees of freedom, for which we will use cross-validation, fitting several models for each value of degrees of freedom. As a reference for the right number of knots we will use the default value used in the `smooth.spline` function for number of observations (n) greater than 50, this is $k = O(n^{1/5})$.

```
load("meat.Rdata")
attach(meat)

abs.850.s <- sort(abs.850,index.return = TRUE)
lgfat <- log(Fat)
lgfat <- lgfat[abs.850.s$ix]

# Reference degrees of freedom
n <- nrow(meat)
q <- ceiling(n^(1/5))
```

We can see that for 215 observations the reference for the degrees of freedom is 3. Now we will use cross-validation and plot the error $(y - \hat{y})$ in blue for each value of degrees of freedom. Additionally, just a matter of comparisson, we will plot the adjusted mean deviance in green.

```
# 10-fold cross-validation to choose the righ degrees of freedom (number of knots)
#seed <- 1800
#set.seed(seed)

# step 1, find the relation between trainning error ~ df
train_set <- createDataPartition(lgfat, p = 0.8, list = FALSE)
train_df <- meat[train_set,]
test_df <- meat[-train_set,]

# Degrees of freedom
degrees.freedom <- 3:40

x <- sort(train_df$abs.850, index.return = TRUE)
y <- log(train_df$Fat)[x$ix]
```

```r
x <- x$x

# Polynomial degree
q <- 3

# Models fitted with different values of degrees of freedom
fitted_models_bs <- apply(t(degrees.freedom), 2,
                          function(deg_freed) lm(y ~ bs(x = x, df = deg_freed, degree = q)))

# Adjusted Mean Deviance
amd <- sapply(fitted_models_bs, function(model) deviance(model)/nobs(model))

# folds
n_folds <- 10
folds_i <- sample(rep(1:n_folds, length.out = length(meat$Fat)))

cv_results <- matrix(NA, nrow = n_folds, ncol = length(degrees.freedom))

for (k in 1:n_folds) {
  test_i <- which(folds_i == k)
  train_xy <- meat[-test_i, ]
  test_xy <- meat[test_i, ]

  x.train <- sort(train_xy$abs.850, index.return=TRUE)
  y.train <- log(train_xy$Fat)[x.train$ix]
  x.train <- x.train$x
  fitted_models <- apply(t(degrees.freedom), 2, function(deg_freed) lm(y.train ~ bs(x = x.train, df = de
  x.test <- sort(test_xy$abs.850, index.return=TRUE)
  y.test <- log(test_xy$Fat)[x.test$ix]
  x.test <- x.test$x

  # Fit different linear models for every value of degrees of freedom


  # Prediction over validation (test) set
  pred <- mapply(function(obj, deg_freed)
                 predict(obj, data.frame(bs(x = x.test, df = deg_freed, degree = q))),fitted_models, de

  # Calculate the Mean Square Error
  cv_results[k, ] <- sapply(as.list(data.frame(pred)),
                            function(y_hat) mean((y.test - y_hat)^2))
}

# Take the mean of the cross-validation mean square error
cv <- colMeans(cv_results)

plot(degrees.freedom, amd, type = "l", lwd = 2, col = 3,
     ylab = "amd:green line; (y - y_hat): blue line",
     xlab = "Degrees of freedom", main = "10-fold Cross-Validation", ylim = c(0, 1))
cv_sd <- apply(cv_results, 2, sd)/sqrt(n_folds)
# errbar(degrees.freedom, cv, cv + cv_sd, cv - cv_sd, add = TRUE, col = "steelblue2", pch = 19, lwd = 0
lines(degrees.freedom, cv, lwd = 2, col = "steelblue2")
points(degrees.freedom, cv, col = "steelblue2", pch = 19)
```

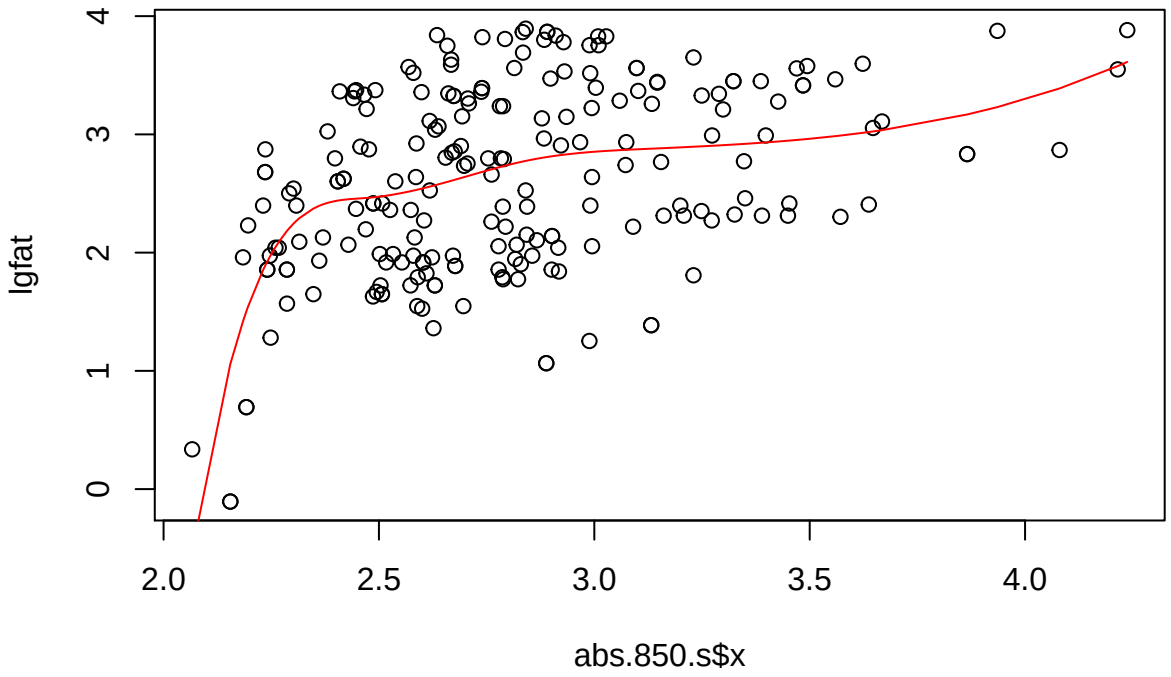



```
# Best degrees of freedom
min.idx <- which.min(cv)
best.deg_freed <- degrees.freedom[min.idx]
```

From the plot we can see that the error (blue line) gets slighly higher as the number of degrees of freedom increases. If we check numerically, we can see that minimum value for the error is obtained using 7 degrees of freedom, which is consequent to the reference value calculated previously.

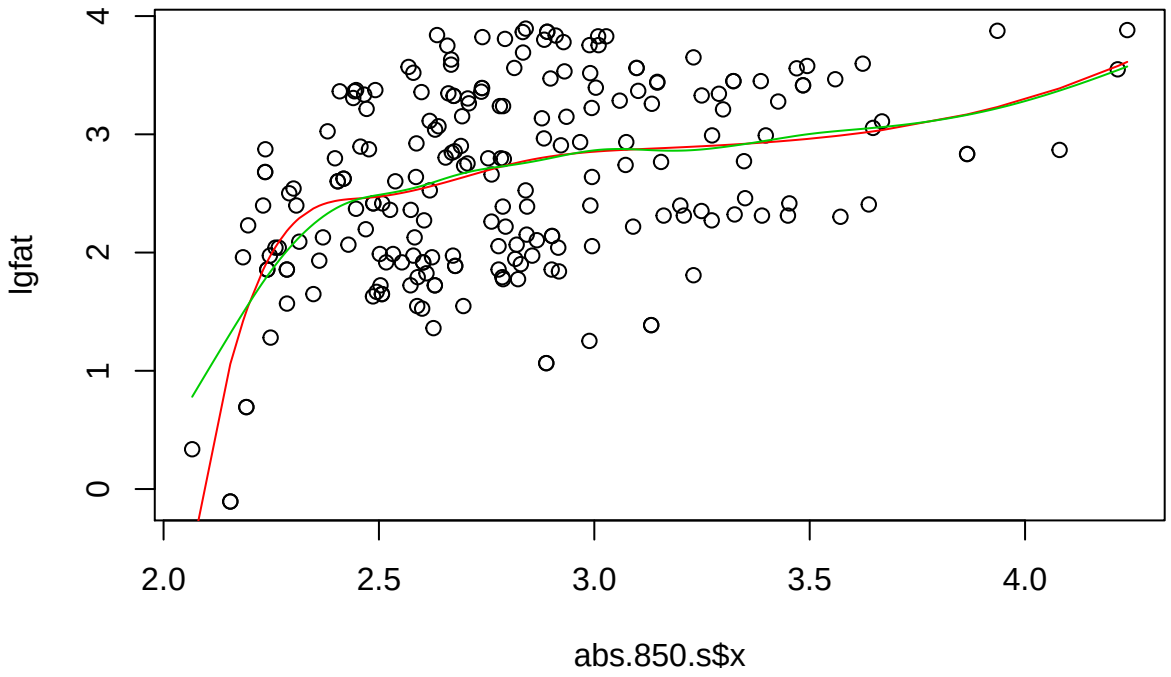
Having calculated the righ value of degrees of freedom we can build our final linear model using a B-spline basis for `abs.850`.

```
x.basis <- bs(x = abs.850.s$x, df = best.deg_freed, degree = q)
lm.spl <- lm(lgfat~x.basis)
plot(abs.850.s$x, lgfat)
lines(abs.850.s$x, lm.spl$fitted.values, col=2)
```



Additionally, we will use `smooth.spline` function for the same regression using the same number of effective parameters to compare both fits.

```
new.abs.850 <- seq(min(abs.850.s$x), max(abs.850.s$x), length=length(abs.850.s$x))
sm.spl <- smooth.spline(abs.850.s$x, lgfat, df=best.deg_freed)
pred.lgfat <- predict(sm.spl, x = new.abs.850)
plot(abs.850.s$x, lgfat)
lines(abs.850.s$x, lm.spl$fitted.values, col=2)
lines(pred.lgfat,col=3)
```




```
# we can find that two lines match well.
```

We san see that both curves follow more or less the same pattern, however, the one produces by `smooth.spline` (green) function is smoother than the one fitted with linear regression (red).