Homework 1

Advanced Methods for Data Analysis (36-402)

Due Friday January 25, 2019 *at 3:00pm* on Canvas

See the syllabus for general instructions regarding homework; note that you should **always show all your work** and submit both a writeup and R code.

1 Kernel regression

Recall that we discussed kernel regression in Lecture 1. In this problem you will use a different data set. The data come from a sample of automobiles and consist of pairs of engine displacement and miles-per-gallon. The data are in the file `engine.Rdata` on the Canvas website. It is going to be helpful for you to look over the code `Demo1_intro.R` that we used in Lecture 1 carefully.

(a) Download the file `engine.Rdata` from Canvas, and load it into your R session with `load("engine.Rdata")`. You can type `ls()` to see the R objects that have been loaded into memory. These will include four data matrices similar to the ones used in `Demo1_intro.R` plus a vector `x0` which has 411 equally-spaced values that cover the range of all of the observed predictor values. These should be used along the horizontal axis for plotting the fitted kernel regressions to reduce the effects of the gaps between the observed predictor values.

The matrices `engine.xtrain` and `engine.ytrain` are each 32×40, containing 40 training data sets each of 32 x or y points along its 40 columns. That is, the first column of `engine.xtrain` and the first column of `engine.ytrain` together make up a training data set of 32 (x, y) pairs. The x variable is engine displacement, and the y variable is miles-per-gallon.

Hence, amassing the train data sets together, there are $40 \times 32 = 1280$ x points and 1280 y points *in total*. Plot these 1280 y points versus the corresponding 1280 x points, on a single plot, to get an idea of the trend, if any. (Hint: there is an easy way to do this with a single call to the `plot` function.)

(b) For the next part, we will restrict our attention to just the first training set, i.e., the first columns of `engine.xtrain` and `engine.ytrain`. There is a standard R function that fits normal-kernel (Nadaraya-Watson) regression, but it has some serious issues. Hence, we will use the function `npreg`. To use this function you must first install the `np` package and then bring it into your workspace with `library(np)`. As with `lm`, `npreg` requires that you fit your kernel regression using data from a `data.frame`, so you will need to put `engine.xtrain[,1]` and `engine.ytrain[,1]` into a `data.frame` first. If you call that

`data.frame first`, with the two columns called `x` and `y`, the `npreg` syntax for fitting with bandwidth `h` is

```
kregobj=npreg(y~x,data=first,bws=h)
```

The output of `npreg` can then be used to create predictions and/or fitted values. We will not be using fitted values in this homework, but you can get them using `fitted(kregobj)`. If you want predictions at a new set of points `xnew`, use

```
predict(kregobj,newdata=data.frame(x=xnew))
```

Examples of such `xnew` would be the vector `x0` to be used for plotting or a column of `engine.xtest` for computing average squared test error. The normal kernel is used by default, but other kernels are available.

Fit kernel regressions with the normal kernel to the first set of training points, with 3 different values of the bandwidth parameter: 10, 50, and 100. For each bandwidth value, plot the estimated regression function from kernel regression as a line on top of the training points. To facilitate comparison, compute these lines using `x=x0` in the `predict` function. To be specific, if your command to compute the regression estimate is as stated earlier, then

```
lines(x0,predict(kregobj,newdata=data.frame(x=x0)),
```

will be a good way to *start* the appropriate command which adds a line to the plot. Make sure that a reader can tell which line is which.

(c) By inspection, what happens to the kernel regression fit as we vary the bandwidth parameter? What procedure does this remind you of (that we have already seen)?

(d) Using the same training set, i.e., the first columns of `engine.xtrain` and `engine.ytrain`, we are going to investigate the predictive performance on the first test set, i.e., the first columns of `engine.xtest` and `engine.ytest`. For a set of 30 bandwidth values, equally spaced from 5 to 150 (5 apart), fit a kernel regression to the training points and predict the regression function at the test x points (not `x0`.) Evaluate its test error, measured in terms of average squared distance between the predicted values and the test y values. Hence, you will have a curve of 30 test errors. Plot this test error curve as a function of the underlying bandwidth values.

(e) According to this test error curve, what is the optimal bandwidth value? What is its associated test error? Plot the kernel regression fit, on top of the training points, using this optimal bandwidth value (and using `x0` on the horizontal axis.) Looking at the plot, does your eye agree that this is really a good bandwidth value? Why or why not?

(f) Now repeat part (d), but do the same for each of the 40 training and test data sets in turn, and report the *average* test error curve at each bandwidth value over the 40 sets. Plot the average test error curve with respect to the bandwidth values. What do you see now? Has the optimal value of the bandwidth changed, and has its associated test error changed?

2 Optimism of the Training Error in Linear Regression

In this problem we will investigate the optimism of the training error in linear regression. We are given training data (X_i, Y_i), $i = 1, \dots n$, and test data (X'_j, Y'_j), $j = 1, \dots m$ which we will assume are *all* i.i.d., i.e., these observations are all independent and come from the same distribution.

Assuming that the predictors are p-dimensional, let $\hat{\beta} \in \mathbb{R}^p$ denote estimated linear regression coefficients based on the training data,

$$\hat{\beta} = (\mathbb{X}^\top \mathbb{X})^{-1} \mathbb{X}^\top \vec{Y},$$

where $\mathbb{X}$ is the $n \times p$ matrix with ith row equal to X_i, and $\vec{Y}$ is an n-dimensional vector with ith component Y_i; that is, $\vec{Y} = [Y_1, \dots, Y_n]^T$. We are going to prove that

$$\mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{\beta}^\top X_i)^2\Big] \leq \mathbb{E}\Big[\frac{1}{m}\sum_{j=1}^{m}(Y'_j - \hat{\beta}^\top X'_j)^2\Big],$$

where the expectations above are over all that is random, i.e., over the training set on the left hand side, and over both the training and testing sets on the right hand side. In words, we're proving that the expected training error is always less than or equal to the expected testing error (without many assumptions at all on the true model), meaning that the training error is naïvely optimistic.

(a) Argue that the expected test error is the same whether we have m test points or just 1 test point, i.e.,

$$\mathbb{E}\Big[\frac{1}{m}\sum_{i=1}^{m}(Y'_i - \hat{\beta}^\top X'_i)^2\Big] = \mathbb{E}\big[(Y'_1 - \hat{\beta}^\top X'_1)^2\big].$$

Hence argue that indeed it doesn't matter whether we have m test points or n test points,

$$\mathbb{E}\Big[\frac{1}{m}\sum_{i=1}^{m}(Y'_i - \hat{\beta}^\top X'_i)^2\Big] = \mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^{n}(Y'_i - \hat{\beta}^\top X'_i)^2\Big],$$

and so we may assume without a loss of generality that $m = n$ (the testing and training sets have the same number of samples). (*Hint:* For each $j = 1, \dots, m$, think about the conditional mean of $(Y'_j - \hat{\beta}^\top X'_j)^2$ given the training data $(X_1, Y_1), \dots, (X_n, Y_n)$. What does this tell you about $\mathbb{E}[(Y'_j - \hat{\beta}^\top X'_j)^2]$ for each j?)

(b) Now it is our task to compare the sizes of $\mathbb{E}[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{\beta}^\top X_i)^2]$ and $\mathbb{E}[\frac{1}{n}\sum_{i=1}^{n}(Y'_i - \hat{\beta}^\top X'_i)^2]$. First consider the random variables

$$A = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{\beta}^\top X_i)^2 \quad \text{and} \quad B = \frac{1}{n}\sum_{j=1}^{n}(Y'_j - \tilde{\beta}^\top X'_j)^2,$$

where $\tilde{\beta} \in \mathbb{R}^p$ denotes the estimated linear regression coefficients but fit from the *test set* of size n. Argue that A and B have the same distribution, so $\mathbb{E}(A) = \mathbb{E}(B)$.

(c) Argue that the random variable B defined in part (b) is always less than or equal to the observed test error,

$$B = \frac{1}{n}\sum_{i=1}^{n}(Y_i' - \tilde{\beta}^\top X_i')^2 \leq \frac{1}{n}\sum_{i=1}^{n}(Y_i' - \hat{\beta}^\top X_i')^2.$$

(*Hint:* don't try to plug in a formula for $\tilde{\beta}$, just recall that it is characterized by least squares ...)

(d) Use the result of part (c) to conclude that

$$\mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{\beta}^\top X_i)^2\Big] \leq \mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^{n}(Y_i' - \hat{\beta}^\top X_i')^2\Big],$$

as desired.