Problem Set 3 – Ridge and Lasso Regression

2016-03-15

Simulating Data

To simulate data we must draw random variables with some prespecified distribution. In R for every distribution usually four functions are implemented which are useful for working with distributions and differ by their prefix: r (random), d (density), p (probability), and q (quantile). The prefix is combined with a name for the distribution, e.g. *norm* for the normal distribution: *dnorm* for the density of a normal distribution, *pnorm* for the probability, *qnorm* for the quantiles, and *rnorm* to draw from a normal distribution. (Check out the help page of the functions!)

Here we want to simulate a linear model of the form

$$y_i = x_i'\beta + \varepsilon_i, i = 1, \ldots, n$$

with β a p-dimensional coefficient vector and x_i p-dimensional vector of regressors. In vector notation:

$$y = X\beta + \varepsilon$$

with y and ε n-dimensional vectors and X a $n \times p$-design matrix.

Here the task is to simulate from this model, where we assume that the coefficient vector β has s entries equal to one and all others are zero.

- Set $n = 100$, $p = 10$, $s = 3$
- Create the coefficient vector β. Useful functions: *c()*, *rep()*
- Simulate a design matrix and the error. Useful functions: *matrix()*, *rnorm*
- Construct the model from above. Useful function: %%* for matrix multiplication

Ridge Regression I

- Estimate a ridge regression on simulated data from Exercise 1. Useful function: *glmnet* from the package *glmnet* with default $\alpha = 1$. Also check out the option *lambda* in *glmnet* and the function *cv.glmnet* to perform cross-validation to determine λ.
- Simulate new data from the same model and make predictions both in- and out-of-sample. Calculate the MSE for the predictions (also for the in-sample fit). Useful function: *predict*
- Repeat the previous steps with different settings on n, p, and s.
- Compare the results with ols regression!

Lasso Estimation I

- Redo the calculations from above but with Lasso with varying n, p, and s. Hint: Set option α in *glmnet* to 0
- Compare the results! (In particular compare a "sparse" with a "dense" setting)
- The package *hdm* contains the function *rlasso* which determines the penalization parameter by some theoretical grounded method. Look up the function in the man pages and / or vignette and analyze now the data set using this function. Compare the results.