Summer school: modeling high dimensional data (practical)

A. Bar-Hen

December 13, 2017

1 Study of housing values

The Boston dataset from the MASS package reports the median value of owner-occupied homes in about 500 U.S. census tracts in the Boston area, together with several variables which might help to explain the variation in median value across tracts. The corresponding data.frame contains the following columns

crim: crime rate

zn: proportion of 25,000 square feet residential lots

indus: proportion of nonretail business acres

chas: is the tract bounds the Charles River ?

nox: annual average nitrogen oxide concentration in parts per hundred million

rm: average number of rooms

age: proportion of owner units built prior to 1940

dis: weighted distances to five employment centers in the Boston area

rad: index of accessibility to radial highways

tax: full value property tax rate (/10,000)

ptratio: pupil/teacher ratio

black: proportion of blacks in the population

lstat: proportion of population that is lower status

medv: median value of owner-occupied homes

The objective is to construct a linear model that has good predictive properties for the variable `medv`. Furthermore, we would prefer to only use predictors that are directly tied to the response of the model, and therefore we shall proceed to perform a selection of variables.

1.1 First part: introduction

1.1.1 Preliminaries

Load the Boston dataset

```
library(MASS)
data(Boston)
```

Based on this table create two new matrices named `Boston.train` and `Boston.test`. The first one will be used to train your models and the second one will serve as a validation dataset.

1.1.2 Descriptive analysis

Briefly describe the data: give in broad strokes the main tendencies contained in the dataset, especially those connected to the variable we are attempting to explain. Are there any predictors that appear redundant?

2 Second part: multilinear regression and variable selection

2.1 Multilinear regression

Generate a multilinear model containing all the predictors (we will refer to this model as full). Run a diagnostic of the model. Do the same with the model containing only the intercept (we will refer to this model as null). Compare the two models and comment.

2.2 Exhaustive search

Use the `regsubset` function of the `leaps` package to generate all the possible models containing up to 10 predictors. Represent, in relation to the number of predictors used, the evolution of the square error, of the adjusted R2, the BIC and the C_p for the best 500 models. These measures are accessible by using the `summary` function on the outputs of the `regsubset` function.

2.3 Stepwise Selection

Propose two models, named `step.AIC` and `step.BIC`, that use the stepwise regression procedure in the forward/backwards mode. Briefly study the outputs of these models (anova and summary functions).

3 Third part: penalisation methods

In this part we are interested in the regularised methods ridge and lasso in order to constrain the variance of our estimator and control the variance of our estimator and eventually improve our prediction error. To generate these models we shall use the `glmnet` package. You will mostly need the `glmnet`, `predict`, `cv.glmnet` and plot functions of this package. Type `help(glmnet)`, `help(plot.glmnet)`, `help(cv.glmnet)` and `help(predict.glmnet)` to get help on these functions.

3.1 Ridge Regression

Generate the Ridge regression model on the 10 predictors ensemble. Trace the obtained regularisation path and comment on it. Select a λ through 10-fold cross-validation with the minimum and a ”1 standard error” rules (the most penalized model with a 1 std distance from the model with the least error). We shall name the corresponding models `ridge.min` and `ridge.1se` Take care with the input format for the `glmnet` function.

3.2 Lasso Regression

Generate the Lasso regression model on the 10 predictors ensemble. Trace the obtained regularisation path and comment on it. Select a λthrough 10-fold cross-validation with the minimum and a ”1 standard error” rules (the most penalized model with a 1 std distance from the model with the least error). Also trace the BIC et mBIC criteria, whose analytic expression is reminded to be:

$$\begin{aligned} \text{BIC} &= n\log(\text{err}_{\mathcal{D}} + \log(n)\text{df}(\lambda) \\ \text{mBIC} &= n\log(\text{err}_{\mathcal{D}} + (\log(n) + 2\log(p))\text{df}(\lambda) \end{aligned}$$

We name `lasso.min`, `lasso.1se`, `lasso.BIC` and `lasso.mBIC` the corresponding models.

4 Forth part: Evaluating the quality of the models obtained

For each of your models, i.e. :

- The model corresponding to a constant (`null`),
- The model with all the predictors (`full`),
- The models obtained by using the stepwise methodology (`step.AIC` and `step.BIC`),
- The models obtained by using the ridge methodology (`ridge.min` and `ridge.1se`),
- The models obtained by using the lasso methodology (`lasso.min`, `lasso.1se`, `lasso.BIC` and `lasso.mBIC`),

Estimate its precition errors with the help of the test dataset. You can use the predict functions associated to the different objects you are manipulating. For the ridge and lasso regressions, plot the evolution of the prediction error calculated over the test ensemble versus the values of the regularization parameters (use a logarithmic scale). Highlight the λ values corresponding to the cross-validation and penalization criteria. Comment about the model finally retained.