# Supervised Machine Learning HW 2

*Group 3: Maja Nordfeldt, Jakob Rauch, Timo Schenk, Konstantin Sommer*

*16-1-2020*

## Introduction

## Data

Table 1: Summary statistics

| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |

## Methods

Since we are dealing with a dataset with about 44 predictors and only 77 observations we use a methodology that induces both shrinkage of parameter estimates and variable selection. This is because a situation in which the ratio of observations to predictor variables is low, there is a danger of what is called "overfitting". In such a situation we might be able to find unbiased estimates of a coefficient for each of the variables using the standard multiple regression framework but since there are only few observations available per parameter, the model will do very poorly in out of sample prediction. We therefore use a combination of Ridge and Lasso regressions called elastic net. Both of these methods, as well as their combination, are aimed at driving some of the coefficients towards zero such that they are not used in prediction anymore. In order to do so we extend the loss function, which in OLS is only the residuals sum of square, by a penalty term, that increases with the size of the coefficients. So when minimizing the loss function one can also think about the penalty term as a constraint on the size of the coefficients. Since both of these methods have advantages and disadvantages, which we will describe in the following, we use a combination of the two, the elastic net method.

The loss function for ridge regression is $(y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta$. The first term is the sum of squared residuals, so the loss function of the multiple regression setting. As one can see we are here not only interested in minimizing this but in addition we also add the penalty term $\lambda\beta^T\beta$, where $\lambda$ is the penalty strength. One can see that the larger the chosen value of $\lambda$ the lower will be the optimal coefficients. The penalty term increases here quadratically in $\beta$, which is the key difference to the LASSO method in which the penalty term is $\lambda||\beta||_1$, where $||\beta||_1$ refers to the L1 norm so the sum of the absolute values of the vector. So the penalty term in LASSO increases with the absolute value of the coefficients and one can again state that the larger the chosen penalty term the smaller will be the resulting coefficient values.

The main difference between the two penalty terms is that while Ridge drives the coefficient values close to zero but mostly not exactly to zero, LASSO will assign an exact zero to many of the coefficients. Why this is the case becomes a bit clearer when looking at how the penalty term translate into a constraint on minimizing the RSS. While for Ridge this is $\beta^T\beta < \gamma$, for LASSO this is $||\beta||_1 < \gamma$. Both of these constraints are equivalent to minimizing the respective loss function and both span a space around the origin in which the the optimal coefficient values must be. While in Ridge this is a hyperbole, in LASSO this is a space with straight edges. One can see that when analysing the two dimensional case that it is mostly the case that for LASSO one of the coefficients will be zero and the other one will be relatively large, for Ridge, both coefficients are driven towards zero but not to exactly zero.

Driving some of the coefficients exactly to zero has the advantage that is basically translates into a variable selection exercise, since these coefficients will then not be used in prediction anymore. On the other hand LASSO might also drive coefficients towards zero that are actually not zero and that are then excluded eventhough they might actually explain some of the variation of the endogenous variable. So to use the advantages of both methods we will combine the two in an elastic net approach in which the penalty term is set up as a combination of the two previous ones: $\lambda(\alpha||\beta||_1 + (1 - \alpha\beta^T\beta)$, where $\lambda$ still determines the penalty strength and $\alpha$ determines the importance of the LASSO penalty relative to the Ridge penalty.

Any deviation from the OLS loss function will result in biased estimates for the coefficients, however, a penalty term leads to the resulting coefficients exhibiting smaller variances. Additionally, OLS does not work in the presence of exact or strong multicollinearity since the $X$ matrix of predictors will become singular or close to singular, making the inversion of the $X'X$ matrix impossible or very inexact (due to many rounding errors). More importantly, a sufficiently high penalty prevents overfitting and, in the presence of many predictors and few observations, will outperform a multiple regression estimation in predicting out of sample values. One therfore faces what is known as a bias variance trade of, determined by the penalty strength $\lambda$.

Using the elastic net method, we need to decide for values of the two hyperparameters: the penalty $\lambda$ and $\alpha$, the weight of LASSO relative to Ridge regression. There is an optimal combination of values for these hyperparameters, in the sense of delivering the best out-of-sample performance. To find this optimal combination, we use a method called k-fold cross-validation. In this procedure, we first arrange our dataset into random order (as the order of observations might be correlated with some features of the data, e.g. when neighborhoods are sampled in a spatial order. We then split the data into k bins of equal size. We use the last (k-1) bins as a training sample (i.e. we fit our model to this data and end up with parameter estimates) and use the first bin as test sample, where we calculate the out-of-sample mean squared error, a measure of the distance between predicted values given our parameter estimates and the observed values. We repeat this k times, using in the second/third/… iteration the second/third/… bin for testing and the remaining bins for training. We finally take the square root of the average of these k mean squared errors, which will be our measure of model performance given the hyperparameters. To find the optimal values, we do a grid search - calculating the model performance for each combination of $\alpha$ and $\lambda$ in the grid. The best values will lead to the lowest average mean squared error. These values lead to a model with good predictive performance.

## Results

Choose $k = 7$ to get equally large bins for 77 observations. Very close results glmnet() and our manual function when we specify k=7 for glmnet() - though the estimates are not exactly the same, as both glmnet() and our manual function have non-deterministic results because data is randomly rearranged. Best values are $\alpha = 0.6$ and $\lambda = 0.04$ for a grid search with $\alpha$ ranging between 0 and 1, with stepsize 0.1, and $\lambda$ ranging between 0.01 and 1 with stepsize 0.01.

## Conclusion

## References

## Code

```r
# Copy code chunks that create table output in here
library(pander)
pander(summary(0), caption = 'Summary statistics')
# Copy code chunks that create figure output in here
# This prints the code chunks at the end
```