Erasmus University Rotterdam

Bachelor Thesis

Business Analytics and Quantitative Marketing

Comparing the performance of Elastic Net and Least Angle Regression on simulated data sets

Author:
Frédérique Schellekens
447534

Supervisor:
Dr. P. Wan
Second assessor:
Dr. J. Fu

July 12, 2020

Abstract

This research focuses on the performance of Linear Regression methods. The Elastic Net (Zou & Hastie, 2005) and Least Angle Regression (Efron, Hastie, Johnstone, Tibshirani, et al., 2004) specifically. When choosing which method to apply to a data set, it is important to know the strengths and weaknesses of the methods. By comparing the discussed methods, this decision can be improved. In this research the methods are applied on different data sets, based on their strengths and shortcomings. A comparison based on the Median Mean Squared Error is made and the model performing best for the specific instance concluded. The Median Mean Squared Error for Elastic Net was shown to be smaller in both considered data sets and is concluded to perform better than Least Angle Regression on data sets with grouped variables, for both many and few variables considered.

Contents

1 Introduction **4**

2 Literature **5**

2.1 Criteria Linear Regression 6

2.2 Least Squares Regression 6

2.3 Forward Regression 7

2.4 Penalized Regression 7

2.5 Least Angle Regression 9

2.5.1 LASSO and Stagewise using LAR 9

2.6 Comparison 9

3 Data **10**

3.1 Diabetes data 10

3.1.1 Diabetes Simulation 11

3.2 Simulations 11

3.2.1 Simulation one 12

3.2.2 Simulation two 12

4 Methodology **12**

4.1 Implementation 12

4.1.1 LAR 12

4.1.2 LASSO 13

4.1.3 Stagewise and Stepwise 13

4.1.4 EN 13

4.2 Comparisons 14

4.2.1 Comparing LAR, LASSO and Stagewise 14

4.2.2 Comparing LAR and Elastic Net 15

5 Results **16**

5.1 Comparing LAR LASSO and Stagewise 16

5.2 Comparing LAR and Elastic Net 18

5.2.1 Simulation one 18

5.2.2 Simulation two . . . 19

6 Conclusion **19**

References **22**

Appendix A Simulation one **23**

A.1 MSE LAR . . . 23

A.2 MSE EN . . . 23

A.3 Coefficients LAR . . . 24

A.4 Coefficients EN . . . 25

Appendix B Simulation two **26**

B.1 MSE LAR . . . 26

B.2 MSE EN . . . 26

B.3 Coefficients LAR . . . 26

B.4 Coefficients EN . . . 47

Appendix C Code **67**

C.1 LAR and LASSO . . . 67

C.2 Stepwise and Stagewise . . . 72

C.3 Diabetes Simulation . . . 74

C.4 Simulation one . . . 76

C.5 Simulation two . . . 77

C.6 Train EN . . . 78

C.7 Optimize EN . . . 80

C.8 Validation . . . 80

C.9 LAR with Risk . . . 81

C.10 Run the LAR, EN simulation . . . 84

1 Introduction

Linear Regression is a method used oftentimes and in varying research subjects (Andrews, 1974). As stated by Weisberg (2005), Linear Regression is the basis for many newer methods. Linear Regression is easily understandable and often gives good results. The method often associated with Linear Regression is Least Squares. Specifically Ordinary Least Squares (OLS). For OLS, some assumptions have to hold. For instance, the error terms should all have an equal variance (Weisberg, 2005). If this assumption does not hold for the data set of interest, Weisberg (2005) states a different type of Linear Regression should be used. In this case it could be Weighted Least Squares (Weisberg, 2005). From this it can be concluded that data sets differ and may have different qualities. Based on these qualities, different methods should be used.

James, Witten, Hastie, and Tibshirani (2013) state that one of the downsides of OLS is the fact that for data sets with a high amount of variables compared to the amount of observations, the model produces a low bias but high variance. This is a result of over-fitting. OLS also includes all variables which are considered in the model (James et al., 2013). So if many irrelevant variables are considered, these will all be modelled. This makes the model more complex. It also allows for the irrelevant variables to influence the model (Hawkins, 2004). To fit the model better to data sets where these problems occur, variations and adaptions of the OLS are introduced. These adaptions include methods to select a set of relevant independent variables to be considered in the model, including for instance Stepwise or Stagewise Linear Regression (James et al., 2013). Or solutions to the case of over-fitting which include least absolute shrinkage and selection operator (LASSO) or Elastic Net (EN) (James et al., 2013). Some of these adaptations were found to be computational, this is an aspect on which Least Angle Regression (LAR) as introduced by Efron et al. (2004) performs well.

As can be seen, many different methods are available to model Linear Regression. The choice of method to use is based on the data set. Therefore it is important to know which method performs well on which type of data set. For many methods there have been comparisons of performance and the strengths and weaknesses of the methods are know. For instance, Segal, Dahlquist, and Conklin (2003) compare LAR and LASSO on a data set with a relatively high amount of variables compared to observations. A comparison which, to the author's knowledge, has not previously been made is the comparison of EN and LAR.

This leads to the following research question. **How does the performance of an EN compare to LAR, when considering varying data sets?**
With the sub question: *How do LAR, LASSO and Stagewise Forward Regression compare for the data set as used by Efron et al. (2004)?*

The qualities the data sets, on which the methods are compared, should have, are derived in Section 2. The different data sets will be derived in Section 3. They will be simulated to contain variables correlated in groups, as well as a data set with more variables than observations.

The objective of this research is to provide insight in the difference in terms of performance between the considered models. Which will enable fitting the model choice better to the data set and ultimately better modelling the dependent variable of interest. Although closely related models have been compared, to the author's knowledge, EN and LAR still lack a comparison.

The paper first finds the features of interest when comparing models and then implements the models on different data sets to compare their performance and conclude the difference in application.

This paper will firstly give a introduction of all methods of interest, with a theoretical background of the problem, in Section 2. The data will then be introduced in Section 3 after which the methodology will be explained in Section 4. Finally the results can be found in Section 5, from which a final conclusion will be drawn in Section 6.

2 Literature

In this section the relevant theory will be introduced. Followed by a summary of previous literature related to the research in the form of a model comparison. For the relevant theory, several linear regression methods will be introduced and they will be compared based on their qualities.

2.1 Criteria Linear Regression

This paper focuses on computing Linear Regression models. To be able to test how well a model performs and compare models, we have to define what is considered good. We follow the statement "Goodness is often defined in terms of prediction accuracy, but parsimony is another important criterion: simpler models are preferred for the sake of scientific insight into the $x - y$ relationship." (Efron et al., 2004, p. 407). These criteria are also supported by Zou and Hastie (2005). What can be concluded from this statement is that, as the goal is to predict a certain dependent variable using the linear model, the accuracy with which this is done correctly is of importance. Intuitively, to predict well, one could simply input as many variables as possible to get a better prediction. A result of doing this can be fitting the model too well to a specific data set, the model will then fit very well to the data set but is not general any more, which is called over-fitting (James et al., 2013, p. 29-34). Also with a large amount of variables, the actual effects working on the dependent variable are no longer clearly deductible from the model (Zou & Hastie, 2005). A smaller model would be preferred. How interpretable a model is, is referred to as 'parsimony'. Parsimony also is able to prevent over-fitting (Hawkins, 2004). As less variables are being considered in the model, it allows the model to fit less perfectly to the data set. How well a model fits to the data and the parsimony of this model are of importance when modeling a data set, as can be seen in the above quote of Efron et al. (2004).

Within the prediction accuracy another trade off occurs. This is the trade off between variance and the bias as explained more thoroughly in James et al. (2013, p. 33-37). When over-fitting occurs, a low bias and high variance are expected.

2.2 Least Squares Regression

Least Squares is a relatively simple linear regression, Least Squares is explained by Hastie, Tibshirani, and Friedman (2009, p. 44-49). The model is optimized by minimizing the euclidean distance between the prediction and the actual dependent variable value. For two-dimensional data, this method is often visualised as a line between the data points. The best line is the line for which the sum of the distance of all point to the line is minimized. Several problems arise when using Least Squares, as stated by James et al. (2013, p. 203-204) . Firstly when little observations are used to estimate many parameters, over-fitting can occur. Also if many variables are included, the parsimony will not hold. To solve this problem, only a few relevant variables can be used instead of all available variables.

2.3 Forward Regression

One way of only including a few variables and not all is by selecting only a few of the total amount of available variables (James et al., 2013, p.205-214). The objective is selecting a subset of all variables, containing only the most influential variables. Two types of Forward Regression are used in this paper. Firstly a method called Stepwise Forward regression, from here on referred to as 'Stepwise', as explained by James et al. (2013, p. 203-204) is introduced. This particular method continuously adds variables which are correlated to the dependent variable until all variables are added or some condition is fulfilled. As stated by Efron et al. (2004), in Stepwise, relatively large steps are taken towards the solution. This method often leads to selecting the best variable in the current step, but it is not able to look forward and consider the whole model. The best variable now, might not be part of the best model. So by choosing the variable now, the best model will not be reached. This feature is called 'greedy'.

A solution to this greedy feature is taking smaller steps, which is implemented in Stagewise Forward Linear Regression (from now on referred to as Stagewise) as described in Hastie et al. (2009, chapter 10.3). Efron et al. (2004) states this variation of the Forward Linear Regression method is less greedy, however the method is very computational due to the large amount of small steps being taken.

2.4 Penalized Regression

A Penalized Regression implements a certain restriction on the Least Squares Regression. As stated before, Least Squares has the risk of obtaining a high variance and overfitting. This is part of the bias-variance trade off. A method which is able to influence this trade off is the Ridge Regression as introduced by Hoerl and Kennard (1970). This method adds a penalty to the standard Least Squares Regression which restricts the sum of the squared coefficients (James et al., 2013, p. 215-219). This restricts the model and causes the coefficients to be lower than in Least Squares. The model is not able to fit as well to the data in the training set, which prevents over-fitting. The equation minimized in Ridge regression is stated in Equation 1. Where p equals the amount of variables.

$$L(\beta, \lambda) = |y - X\beta|^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{1}$$

The result of this restriction is a lower variance, but also a higher bias. The method is not able to set coefficients equal to zero and therefore does not create parsimony (Zou & Hastie, 2005).

A method also able to influence the bias-variance trade-off is the LASSO Regression as introduced by Tibshirani (1996). This method uses the penalty of restricting the sum of the absolute value of the coefficients. The equation minimized in LASSO is stated in Equation 2.

$$L(\beta, \lambda) = |y - X\beta|^2 + \lambda \sum_{j=1}^{p} |\beta| \tag{2}$$

Efron et al. (2004) explains that one of the results of this restriction is coefficients being able to be set to zero. This method therefore also is able to create parsimony. Several problems still arise for LASSO, as explained in Zou and Hastie (2005). The problem we will focus on is the fact that LASSO is not able to treat groups of correlated variables as such. A Ridge Regression in some cases then outperforms the LASSO (Tibshirani, 1996).

For the objective of keeping the benefits of LASSO, namely the parsimony and trade off capacities, and gaining the capacity to group variables, the Elastic Net is introduced by Zou and Hastie (2005). This method combines the restrictions of the Ridge and LASSO Regression in one method. The equation can be seen in Equation 3, where Equation 4 and Equation 5 hold (Zou & Hastie, 2005). With the solution in Equation 6 (Zou & Hastie, 2005).

$$L(\beta, \lambda_1, \lambda_2) = |y - X\beta|^2 + \lambda_1 |\beta|^2 + \lambda_2 |\beta|_1 \tag{3}$$

$$|\beta|^2 = \sum_{j=1}^{p} \beta_j^2 \tag{4}$$

$$\beta|_1 = \sum_{j=1}^{p} |\beta| \tag{5}$$

$$min(L(\beta, \lambda_1, \lambda_2)) \tag{6}$$

A big improvement in this method is the ability to treat variables correlated in groups as a group, instead of altering these variables independently (Zou & Hastie, 2005).

2.5 Least Angle Regression

As stated before, Stagewise contains a lot of little steps, which is very computational. To solve this problem and find a method, accomplishing the same objective with less steps, the Least Angle Regression (LAR) method was introduced by Efron et al. (2004). This method is a very intuitive way to use the variables to get to the objective of modelling the dependent variable. Namely by using the equiangular vector between parameters as a direction, instead of going in the direction of the variables separately. The method is able to reach the objective through as many steps as there are parameters whilst not becoming as greedy as Stepwise. The method chooses the highest correlated variable in each step and adds it to the set of variables it uses to calculate the direction to take a step in. By including multiple variables in the step, the stairwise steps of Stagewise are not necessary. Which allows LAR to compute the model much faster. For a thorough explanation of the mathematics and method, this paper refers to Efron et al. (2004).

2.5.1 LASSO and Stagewise using LAR

As explained in (Efron et al., 2004), the LAR method can be adjusted to capture the LASSO as well as Stagewise. This allows for both methods to be less computational compared to their original form. The adjustment for the LASSO allows for the model to not only gain a variable per step, but also to be able to lose a variable in a step. For the Stagewise, an adjustment when the direction is not feasible for the original method is implemented. This leads to the possibility to delete more than one variable per step from the model.

2.6 Comparison

Some of the before mentioned methods have already been compared in terms of performance. This section will give a brief insight in the conclusions drawn from these comparisons. Theoretically, Efron et al. (2004) shows LAR, LASSO and Stagewise to give nearly identical results.

When comparing with Elastic Net the following conclusions can be made based on before mentioned information. LASSO, and therefore also Elastic Net, is able to both control the variance-bias trade off and the parsimony.

A comparison of LASSO, Ridge and EN is performed by Zou and Hastie (2005). EN is shown to perform well on data sets with a relatively large amount of observations. As expected treating the variables as groups when they are group correlated is also a feature primarily visible in EN. For the

data sets and simulations used by Zou and Hastie (2005), EN outperformed LASSO.
For a theoretical comparison of LAR and LASSO, Efron et al. (2004) is followed. As explained here, the LAR algorithm is able to work for cases where the amount of variables exceeds the amount of observations. The LASSO algorithm is not able to do this.
To the knowledge of the author, no comparison of both Stagewise and LAR with Elastic Net has been documented to date.

3 Data

In this section the necessary data to answer the research question will be discussed and introduced. Firstly to answer the sub question, testing the performance of LAR, LASSO and Stagewise on the data sets as used by Efron et al. (2004). The data sets as used have been retrieved from Stanford University [1], the data sets consist of data on diabetes research. Secondly simulated data sets are introduced to compare the EN and LAR.

3.1 Diabetes data

The data used in this section follows Efron et al. (2004). When testing LAR and comparing LAR with LASSO and Stagewise, firstly a data set of 10 variables, concerning diabetes, is used. This data set consists of 442 patient observations. The variables include the categorical sex of the patient, the numerical age, BMI, Blood Pressure and finally several numerical 'serum measures'. The dependent variable is a quantitative indication of the disease of the patient (Efron et al., 2004). Secondly a data set is used which consists of the original variables, combines with several combinations and squares of the original 10 variables, which leads to 64 variables. The squares include all variables except for he categorical variable 2.
These data sets have been standardized. The amount of observations is much larger than the amount of variables. A visualisation of the dependent variable of the data set can be seen in Figure 1. As can be expected from a standardized data set, the values lay around zero. No normal distribution is seen.

[1] https://web.stanford.edu/ hastie/StatLearnSparsity$files/DATA/diabetes.html$

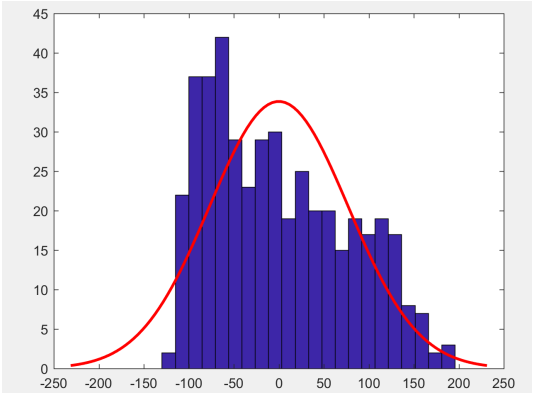

Figure 1: A histogram of the y.

3.1.1 Diabetes Simulation

Apart from the data set itself as introduced in Section 3.1, the larger data set of 64 variables is used in a Simulation. This Simulation again follows Efron et al. (2004). Firstly the model is trained on the actual data set of 10 variables, the mean and residuals from this model are used to construct the simulations. For each simulation, from the residuals, 442 residuals are picked randomly. A residual could be picked more than once. With the actual mean and the new set of residuals, the simulated y is generated by Equation 7 (Efron et al., 2004).

$$y^* = \mu + \epsilon^* \tag{7}$$

Where y^* is the simulated dependent variable, μ is the actual μ and ϵ^* is the simulated set of residuals. The simulated dependent variable and the set of dependent variables containing 64 variables, make up the data set used in each simulation.

3.2 Simulations

For a comparison using the Elastic Net, a data set with specific qualities is needed. As can be concluded from Section 2, the main difference between all models are their computational qualities, their performance on large or smaller data sets (relative to the amount of parameters) and their performance when dealing with highly correlated data sets. To best be able to compare the methods, this information will be used. Two data sets are simulated to capture the qualities discussed. Both Simulations are based on 'example 4' by Zou and Hastie (2005). This simulated data set contains 40 variables, where the first 15 variables consist of three groups of correlated variables with a coefficient of three. The last 25 variables are not informational in the actual model as the

coefficient for these variables is set to zero. 50 simulated observations are used as a training set, 400 observations are used as a testing set. As a result the training data set does capture the group correlated variables, but does not contain more variables than observations.

3.2.1 Simulation one

To alter the described simulation to allow for EN to be implemented, the amount of variables used in the training set was increased to 500. This was done because cross-validation as used in the code, creates subsets of the observations. In the case of 50 observations, for the code used, each subset would contain one observation, which is not desired. Also in Zou and Hastie (2005), three data sets are simulated. In this paper only two data sets will be simulated, the 'validation set' is omitted.

3.2.2 Simulation two

To also capture the instance where the amount of variables is larger than the amount of observations, a different simulation is introduced. This simulation is based on the previous simulation as introduced in Section 3.2.1, but contains 360 non-informational variables and 240 informational variables. Again the informational variables are divided in three correlated groups.

4 Methodology

To compare the different methods on their performance, firstly the methods have to be implemented. Afterwards comparison methods are discussed.

4.1 Implementation

All methods and Simulations are implemented using MATLAB R2017a (MATLAB, 2017). The code can be found in Appendix C.

4.1.1 LAR

To implement LAR, the formulas as introduced in Efron et al. (2004) were used. This method is able to take steps towards the solution by using several variables at the same time. To do this efficiently, it uses an equiangular vector. Every step a variable is added, however no variables can

be removed from the set used to calculate the vector. Therefore the parsimony decreases for every step, the method however does get closer to the least squares output with each step as well. A selection criterion for which step to take as output for the model will be introduced in Section 4.2.2.

4.1.2 LASSO

The LASSO method is implemented by using a modification of LAR, again following Efron et al. (2004). The modification allows for variables to also be removed from the active set when otherwise the assumptions of LASSO do not hold.

4.1.3 Stagewise and Stepwise

Both Stepwise and Stagewise are based on taking subsequent steps in the direction of the variables. For Stepwise, as explained in Efron et al. (2004), in every step a new variable is added to the set of used variables. This variable should not before be used and is chosen on having the highest correlation with the residuals. A large step in the direction of this variable is taken, namely a step of the size equal to the absolute highest correlation. The Stagewise implementation can be implemented following LAR as described by Efron et al. (2004). However for this paper it is based on Stepwise following Efron et al. (2004). Where Stepwise takes large steps in the direction of the variable with the highest correlation, Stagewise takes many small steps in the direction of the variable with the highest correlation (Efron et al., 2004).

4.1.4 EN

The implementation of EN is based on Zou and Hastie (2005). EN contains two parameters which should be chosen before being able to optimize the model. These variables are called hyperparameters. In the case of EN, these variables are the *alpha* and *lambda*. To find these hyperparameters, a method is used where several subsets are made of the observations. By doing optimizations of EN where every time a different subset is the validation set and all other subsets combined are the training set, the optimal hyperparameters can be concluded. This method is called 'cross-validation'. A training set is here a data set on which the model is trained. The validation set is the data set on which the trained model is implemented to be able to compare the different trained models. The comparison in this paper is done based on the Mean Squared Error (MSE) as will be explained in Section 4.2.2. Where ten subsets are created, this is called 'ten-fold cross-validation'. The optimization of EN by using cross-validation for the hyperparameters follows Zou and Hastie (2005).

Once the ten sets of hyperparameters are found, they are used to construct ten EN models on the entire data set. For these ten models, the optimal coefficients and the MSE are reported.
To concur with LAR, the ten models are not validated on a different data set. The optimal model is decided based on the MSE of the training set. This decision ensures not more data sets are used for EN, however by not validating on a different data set, the decision of optimal coefficients could be incorrect. This should be considered when drawing a conclusion from the results.

4.2 Comparisons

4.2.1 Comparing LAR, LASSO and Stagewise

For the implementation, firstly the data set has to be standardized. In this case, the used data set was already standardized. This data is as described in Section 3.1.
The entire following comparison is introduced by Efron et al. (2004). The LAR method is executed firstly using the data set with ten variables. To visualise the trajectory of the correlations for every separate variable, a graph is created mapping these correlations for each step of the algorithm. Also a graph of the coefficient for each variable plotted against the sum of the absolute coefficients is created. This is again done to analyse the trajectory of, in this case, the coefficients for each separate variable as the sum of absolute coefficients increases.
Then to compare the LAR, LASSO and Stagewise, using the data set consisting of 64 variables, 100 simulations of the data set were modelled using the methods. The methods had a maximum of 40 steps or 40 coefficients to be nonzero. Stepwise is also included in the graph. Stepwise is expected to behave differently to the other models and is included to give a reference point to different methods. To compare the performance of the methods, the 'average proportion explained' is used. The formula for the average proportion explained, is stated in Equation 8 (Efron et al., 2004). Where μ is the actual and $\hat{\mu}$ is the estimated prediction.

$$pe(\hat{\mu}) = 1 - \|\hat{\mu} - \mu\|^2 / \|\mu\|^2 \tag{8}$$

For each step the average proportion explained over all simulations is plotted against the step or average amount of coefficients nonzero. A higher proportion explained could be preferred, based on the objective of the model.

4.2.2 Comparing LAR and Elastic Net

The following comparison was implemented on both Simulations as introduced in Section 3.2. The objective is to be able to compare LAR and EN on a comparison measure. The Median Mean Squared Error (MMSE) will be used, following for instance Li, Lin, et al. (2010) and Zou and Hastie (2005).

The MMSE is calculated by taking the Median of the MSE. The MSE is calculated by Equation 9 (James et al., 2013, p. 29-30).

$$MSE = \frac{1}{n} * \sum_{i=1}^{n} (y_i - \hat{f}(x_i))^2 \tag{9}$$

Where n is the amount of observations, y_i is the actual value of the dependent variable for observation i and finally $\hat{f}(x_i)$ is the predicted value at observation i. A higher MMSE error, as with the MSE, implies a higher error and therefore a less preferred model.

To obtain the MMSE for the LAR and EN method, two Simulations as explained in Section 3.2 are used. For every simulation the MSE is concluded. The MMSE is concluded based on the MSE found in the simulations. For the Simulation as explained in Section 3.2.1, 10 simulations are used. This is done because the simulations were costly in terms of time. For the second Simulation as explained in Section 3.2.2, 3 simulations are used. Again because of a high cost in terms of time.

To compare using the MSE of every simulation, the MSE of each simulation is concluded using a validation set. This process follows Zou and Hastie (2005). The optimal model found by implementing Section 4.1 and concluding the optimal model, is implemented on a separate 'testing set' as found in Section 3.2. Which is used as a validation set from which the MSE for the optimal model is concluded.

The last step is to be able to conclude the optimal model within each simulation. The selection criterion for LAR will now be introduced. Here the parsimony and accuracy are relevant. Following Efron et al. (2004), a criterion based on a risk calculation is used. By minimizing this risk, the optimal solution is found. For EN the optimal model is found using Section 4.1.4.

In summary, for both Simulations as introduced in Section 4.2.2 a certain amount of simulations

is ran. Within each simulation the optimal model is concluded for the simulated data set. The MSE is concluded from these simulations and by using the MMSE, LAR and EN can be compared.

Aside from the MMSE, the values of the coefficients found in the optimal models are also of interest. Firstly the parsimony of the model can be concluded based on these coefficients. But also whether the model is able to identify irrelevant variables and set this coefficient to zero, can be seen in the coefficients. Finally, as the data sets contain group correlated variables, a preferred model is able to treat these variables as a group and does not treat them independently (Zou & Hastie, 2005).

The before mentioned comparisons are executed and the results are described in the following section.

5 Results

In this section the results will be introduced and discussed.

5.1 Comparing LAR LASSO and Stagewise

Firstly LAR itself is applied to the data set. The variables were added on each step in the order [3 9 4 7 2 10 6 5 8 1]. The regression coefficients for each variable are plotted against the sum of the absolute regression coefficients. The output of this plot can be seen in Figure 2. Also for each variable the correlations over every step are plotted against the steps. This plot can be seen in Figure 3.

The order in which the variables are added, agree with the order which occurred in Efron et al. (2004). Also when looking at the figures, not a big difference is seen. As the data set is the same as in the original research, no differences are expected. However, a difference in scaling in Figure 3 can be seen. This difference is expected to be a mistake in the paper by Efron et al. (2004). Also Figure 2 does not follow the figure in Efron et al. (2004), especially variable 7.

When LAR has been computed, a simulation is implemented for LAR, LASSO, and Stepwise. To be able to compare their performance in terms of 'proportion explained'. After 100 simulations, each consisting of 40 steps, Figure 4 is obtained. For both LAR, LASSO and Stagewise the figure does not differ much from the figure shown in Efron et al. (2004). However for Stepwise the plot does seem to differ greatly. The Stagewise results are slightly above the LAR and LASSO results, this would imply Stagewise to better explain the dependent variable for larger amounts of independent

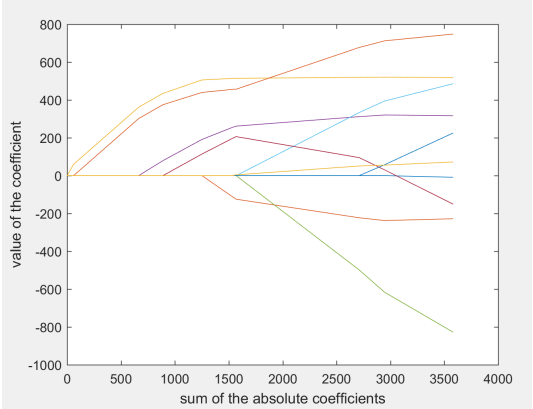

Figure 2: The coefficients for all variables, plotted against the sum of the absolute coefficients.

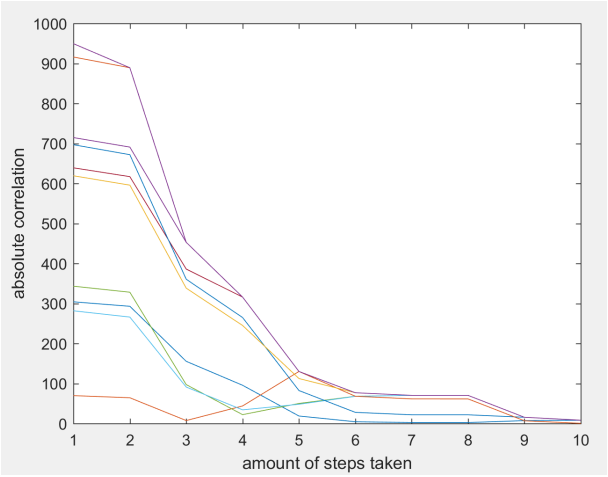

Figure 3: The correlations for each variable as well as the maximum correlation, plotted against the steps.

variables considered. The results given by the Stepwise algorithm seem unlikely. As stated by Efron et al. (2004), a fast increase due to the large steps taken, would be expected. Whereafter the proportion explained will decrease as a result of the algorithm not taking into consideration future steps and overfitting quickly. The obtained results do not show this behaviour, the proportion explained is seen to stay level over the increasing amount of variables included in the model. As this result does not agree with the behavior of a Stepwise model, a mistake in the programming might explain the results.

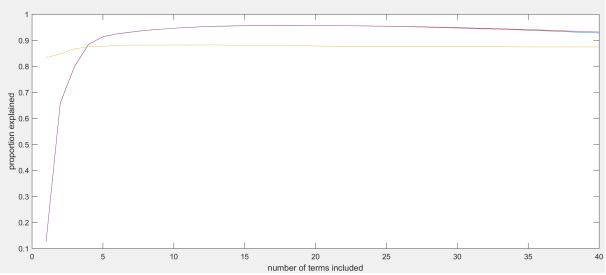

Figure 4: The proportion explained for each method is plotted against the Average number of terms.

5.2 Comparing LAR and Elastic Net

The objective of this paper is to compare the performance of LAR and EN on varying data sets. This subsection will describe the results of this comparison. The conclusions made will be based on the MMSE over several simulations as described in Section 3. As the amount of simulations run in each Simulation are small, no distribution and significance will be included. There is not enough information for these results. The coefficients of only the last simulation within each Simulation is returned, therefore this set of coefficients will be discussed. Because only one set of coefficients is considered, it should be noted the conclusions based on these coefficients are not trustworthy.

5.2.1 Simulation one

The first simulation consists of more observations than independent variables. Where three sets of variables with grouped correlation and several irrelevant variables exist in the independent variables. The MSE values for the LAR method can be found in Appendix A.1. The MSE values for the EN method can be found in Appendix A.2. MMSE for the LAR model is 239.60, the MMSE for the EN is 234.04. These results would lead to the conclusion of preferring EN over LAR for this data set. However it should be noted that the values are close to each other and therefore no definite conclusion can be made. The coefficients of one model of LAR can be found in Appendix A.3. The

coefficients of one model of EN can be found in Appendix A.4. In terms of parsimony, the LAR model used a beta with seven nonzero values. The EN model used a beta with 31 nonzero values. The EN model takes into account many of the irrelevant independent variables, this would indicate a downside to using EN. However this conclusion is in contrast with the benefits of an EN and the reliability of the results should therefore be considered. What can be seen is EN having values relatively close to the actual coefficient value of three for all informational variables, whereas LAR has coefficients varying more from the actual value.

5.2.2 Simulation two

For the second Simulation, a data set with more independent variables than observations is used. Again with three sets of grouped variables and several irrelevant variables. The MSE values for the LAR method can be found in Appendix B.1. The MSE values for the EN method can be found in Appendix B.2. The MMSE for the LAR model is $1.09 * 10^4$, the MMSE for the EN is 303.97. The MMSE of LAR is very high, which means the model does not fit to the data set well. The coefficients of one model of LAR can be found in Appendix B.3. The coefficients of one model of EN can be found in Appendix B.4. For EN the coefficients for the relevant variables are close to the actual value of three, whereas the coefficients for the irrelevant variables are either small or equal to zero. For LAR this is not the case, most variables are zero and only a very few are nonzero. EN however does contain many nonzero coefficients for the irrelevant variables which leads to bad parsimony. Only three simulations were run, so no conclusive decisions can be made. In terms of MMSE, EN is expected to be able to perform well on data sets with a high amount of variables compared to observations. For LAR the author did not find previous information on this topic. The results imply LAR is not able to perform well on this data set.

6 Conclusion

The objective of this paper is to compare several Linear Regression methods. Primarely EN and LAR. This is relevant because when the qualities of data sets on which specific methods work well are known, fitting a model to this data set becomes easier as less models have to be tried. So by comparing EN and LAR for future research, the selection between the two methods can possibly be based on theory instead of practice.

In terms of comparing the methods, EN is able to work well for data sets with grouped variables, as well as data sets with more variables than observations. Therefore the EN and LAR have been implemented on two simulated data sets, capturing these qualities to compare the performance of the methods. Their performance is compared based on the MMSE as well as the obtained coefficients for the optimal models.

For the data set simulated to have group correlated variables, but more observations than variables, the MMSE for EN was 234.04, the MMSE for LAR was 239.60. The coefficients showed EN to include many of the irrelevant variables, whereas LAR had less irrelevant variables with a coefficient of nonzero. For the data set simulated to have group correlated variables and more variables than observations, the MMSE for the LAR model is $1.09 * 10^4$, the MMSE for EN is 303.97. Also the coefficients for EN behave well, whereas very few of the coefficients for LAR are nonzero.

These results are used to answer the Research Question. **How does the performance of an Elastic net compare to a LAR, when considering varying data sets?**
With the sub question: *How do LAR, LASSO and Stagewise Forward Regression compare for the data set as used by Efron et al. (2004)?*

From the results, it can be concluded that for both data sets, EN had a lower MMSE. This implies EN to be preferred. However it should be noted for the first Simulation that MMSE values are very close, so no conclusive statement can be made when comparing LAR and EN. For the first Simulation, LAR had a better parsimony. The preference for the first Simulation can be based on the objective for the model. For the second Simulation however, the MMSE for LAR is much larger and the coefficients do not behave as well as for EN. Therefore for group correlated variables with more variables than observations, the EN can be concluded to perform better. It should be noted that very little simulations were ran, so no definite conclusions can be made.

Several implications of the research and suggestions for future research will now be introduced. Firstly, the fact that for EN a validation set was ommited, causes the method to reduce in reliability. A solution to this problem should be found to make the results more trustworthy. Also as LAR used non standardized data, whereas the EN did use standardized data, the results of both methods can not be formally compared. In the future the data for both methods should be standardized.

Furthermore only a small amount of simulations was used within each Simulation. To obtain more results and for instance obtain a standard error for the MMSE, more simulations should be ran. Finally the coefficients for all models used to obtain the MMSE should be registered. This would make conclusions based on these coefficients more reliable.

For future research; it would be useful to also compare the methods on real data. And possibly include other versions of EN in the comparison. For instance Adaptive Elastic Net as introduced by Zou and Zhang (2009) could be used. Finally more data sets could be used, with for instance non group correlated variables, to generalize the conclusions drawn from this research more.

References

Andrews, D. F. (1974). A robust method for multiple linear regression. *Technometrics*, *16*(4), 523–531.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *The Annals of statistics*, *32*(2), 407–499.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Hawkins, D. M. (2004). The problem of overfitting. *Journal of chemical information and computer sciences*, *44*(1), 1–12.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112). Springer.

Li, Q., Lin, N., et al. (2010). The bayesian elastic net. *Bayesian analysis*, *5*(1), 151–170.

MATLAB. (2017). *version 9.2.0.556344 (r2017a)*. Natick, Massachusetts: The MathWorks Inc.

Segal, M. R., Dahlquist, K. D., & Conklin, B. R. (2003). Regression approaches for microarray data analysis. *Journal of Computational Biology*, *10*(6), 961–980.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288.

Weisberg, S. (2005). *Applied linear regression* (Vol. 528). John Wiley & Sons.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, *67*(2), 301–320.

Zou, H., & Zhang, H. H. (2009). On the adaptive elastic-net with a diverging number of parameters. *Annals of statistics*, *37*(4), 1733.

Appendix A Simulation one

A.1 MSE LAR

Table 1: The MSE for ten simulations using LAR

simulation	MSE
1	265.12
2	191.91
3	207.04
4	218.24
5	250.69
6	235.69
7	269.33
8	243.75
9	271.32
10	233.69

A.2 MSE EN

Table 2: The MSE for ten simulations using EN

simulation	MSE
1	248.15
2	191.74
3	207.98
4	218.33
5	256.35
6	243.19
7	248.11
8	243.01
9	268.95
10	237.04

A.3 Coefficients LAR

Table 3: The coefficients for one of the simulations using LAR

variable	coefficient
1	0.0000
2	0.0000
3	0.9847
4	11.6007
5	0.0000
6	4.0335
7	3.5153
8	−0.0029
9	5.7153
10	0.0000
11	8.0058
12	2.3954
13	0.0000
14	0.0000
15	2.9870
16	0.0000
17	0.0000
18	0.0000
19	0.0000
20	0.0000
21	0.0000
22	0.0000
23	0.0000
24	0.0000
25	0.0000
26	0.0000
27	0.0000
28	0.0000
29	0.0000
30	0.0000
31	0.0000
32	0.0000
33	0.0000
34	0.0000
35	0.0000
36	0.0000
37	0.0000
38	0.0000
39	0.0000
40	0.0000

A.4 Coefficients EN

Table 4: The coefficients for one of the simulations using EN

variable	coefficient
1	2.4310
2	2.4522
3	2.4656
4	2.6377
5	2.4845
6	2.6377
7	2.6549
8	2.5236
9	2.6659
10	2.4957
11	2.7320
12	2.5922
13	2.5678
14	2.5531
15	2.6442
16	0.0160
17	0.5261
18	0.0000
19	0.0000
20	0.0000
21	−0.0198
22	−0.4916
23	−0.2803
24	0.0000
25	−0.2160
26	0.0000
27	−0.9168
28	0.1303
29	−0.0459
30	0.0000
31	−0.7669
32	0.0324
33	0.2030
34	−0.0483
35	−0.4617
36	0.0000
37	0.0000
38	0.0000
39	−0.0180
40	0.3485

Appendix B Simulation two

B.1 MSE LAR

Table 5: The MSE for three simulations using LAR

simulation	MSE
1	$7.4838 * 10^3$
2	$1.0880 * 10^4$
3	$1.2469 * 10^4$

B.2 MSE EN

Table 6: The MSE for three simulations using EN

simulation	MSE
1	303.97
2	316.26
3	301.364

B.3 Coefficients LAR

Table 7: The coefficients for one of the simulations using LAR

Begin of Table	
variable	coefficient
1	0
2	0
3	0
4	0
5	36,43701
6	0
7	0
8	23,09386
9	0,350746
10	0

Continuation of Table 7	
variable	coefficient
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	37,80286
26	0
27	0
28	0
29	0
30	0
31	0
32	0
33	0
34	0
35	0
36	0
37	0
38	0
39	0
40	0

Continuation of Table 7	
variable	coefficient
41	7,300507
42	0
43	0
44	0
45	0
46	0
47	0
48	0
49	9,676849
50	0
51	0
52	0
53	0
54	0
55	0
56	0
57	0
58	26,55317
59	0
60	38,51033
61	0
62	0
63	0
64	0
65	0
66	0
67	0
68	0
69	0
70	0

Continuation of Table 7	
variable	coefficient
71	0
72	0
73	0
74	0
75	0
76	0
77	0
78	0
79	0
80	0
81	0
82	0
83	0
84	0
85	0
86	0
87	0
88	0
89	0
90	0
91	0
92	0
93	0
94	0
95	0
96	0
97	0
98	0
99	0
100	0

Continuation of Table 7	
variable	coefficient
101	0
102	0
103	0
104	0
105	0
106	0
107	0
108	0
109	0
110	0
111	0
112	0
113	0
114	0
115	0
116	0
117	0
118	0
119	0
120	0
121	12,10556
122	0
123	0
124	0
125	0
126	0
127	17,74682
128	0
129	0
130	0

Continuation of Table 7	
variable	coefficient
131	0
132	0
133	20,83736
134	0
135	0
136	0
137	0
138	0
139	0
140	0
141	0
142	0
143	0
144	35,50126
145	0
146	0
147	36,65461
148	0
149	0
150	0
151	0
152	0
153	0
154	0
155	0
156	51,97355
157	0
158	0
159	0
160	0

Continuation of Table 7	
variable	coefficient
161	0,480041
162	55,94665
163	1,559043
164	16,46237
165	0
166	0
167	0
168	0
169	0
170	0
171	0
172	0
173	0
174	0
175	0
176	0
177	0
178	16,92194
179	0
180	0
181	0
182	0
183	0
184	0
185	0
186	0
187	0
188	0
189	0
190	0

Continuation of Table 7	
variable	coefficient
191	0
192	0
193	0
194	0
195	0
196	15,26166
197	0
198	8,261605
199	0
200	0
201	0
202	0
203	0
204	0
205	0
206	0
207	0
208	0
209	0
210	0
211	0
212	0
213	0
214	0
215	25,08781
216	0
217	0
218	0
219	0
220	0

Continuation of Table 7	
variable	coefficient
221	0
222	0
223	5,599927
224	0
225	0
226	0
227	0
228	0
229	0
230	0
231	0
232	4,529976
233	0
234	0
235	22,27213
236	0
237	0,444717
238	0
239	0
240	0
241	0
242	0
243	0
244	0
245	0
246	0
247	0
248	0
249	0
250	0

Continuation of Table 7	
variable	coefficient
251	0
252	0
253	0
254	0
255	0
256	0
257	0
258	0
259	0
260	0
261	0
262	0
263	0
264	0
265	0
266	0
267	0
268	0
269	0
270	0
271	0
272	0
273	0
274	0
275	0
276	0
277	0
278	0
279	0
280	0

Continuation of Table 7	
variable	coefficient
281	0
282	0
283	0
284	0
285	0
286	0
287	0
288	0
289	0
290	0
291	0
292	0
293	0
294	0
295	0
296	0
297	0
298	0
299	0
300	0
301	0
302	0
303	0
304	0
305	0
306	0
307	0
308	0
309	0
310	0

Continuation of Table 7	
variable	coefficient
311	0
312	0
313	0
314	0
315	0
316	0
317	0
318	0
319	0
320	0
321	0
322	0
323	0
324	0
325	0
326	0
327	0
328	0
329	0
330	0
331	0
332	0
333	0
334	0
335	0
336	0
337	0
338	0
339	0
340	0

Continuation of Table 7	
variable	coefficient
341	0
342	0
343	0
344	0
345	0
346	0
347	0
348	0
349	0
350	0
351	0
352	0
353	0
354	0
355	0
356	0
357	0
358	0
359	0
360	0
361	0
362	0
363	0
364	0
365	0
366	0
367	0
368	0
369	0
370	0

Continuation of Table 7	
variable	coefficient
371	0
372	0
373	0
374	0
375	0
376	0
377	0
378	0
379	0
380	0
381	0
382	0
383	0
384	0
385	0
386	0
387	0
388	0
389	0
390	0
391	0
392	0
393	0
394	0
395	0
396	0
397	0
398	0
399	0
400	0

Continuation of Table 7	
variable	coefficient
401	0
402	0
403	0
404	0
405	0
406	0
407	0
408	0
409	0
410	0
411	0
412	0
413	0
414	0
415	0
416	0
417	0
418	0
419	0
420	0
421	0
422	0
423	0
424	0
425	0
426	0
427	0
428	0
429	0
430	0

Continuation of Table 7	
variable	coefficient
431	0
432	0
433	0
434	0
435	0
436	0
437	0
438	0
439	0
440	0
441	0
442	0
443	0
444	0
445	0
446	0
447	0
448	0
449	0
450	0
451	0
452	0
453	0
454	0
455	0
456	0
457	0
458	0
459	0
460	0

Continuation of Table 7	
variable	coefficient
461	0
462	0
463	0
464	0
465	0
466	0
467	0
468	0
469	0
470	0
471	0
472	0
473	0
474	0
475	0
476	0
477	0
478	0
479	0
480	0
481	0
482	0
483	0
484	0
485	0
486	0
487	0
488	0
489	0
490	0

Continuation of Table 7	
variable	coefficient
491	0
492	0
493	0
494	0
495	0
496	0
497	0
498	0
499	0
500	0
501	0
502	0
503	0
504	0
505	0
506	0
507	0
508	0
509	0
510	0
511	0
512	0
513	0
514	0
515	0
516	0
517	0
518	0
519	0
520	0

Continuation of Table 7	
variable	coefficient
521	0
522	0
523	0
524	0
525	0
526	0
527	0
528	0
529	0
530	0
531	0
532	0
533	0
534	0
535	0
536	0
537	0
538	0
539	0
540	0
541	0
542	0
543	0
544	0
545	0
546	0
547	0
548	0
549	0
550	0

Continuation of Table 7	
variable	coefficient
551	0
552	0
553	0
554	0
555	0
556	0
557	0
558	0
559	0
560	0
561	0
562	0
563	0
564	0
565	0
566	0
567	0
568	0
569	0
570	0
571	0
572	0
573	0
574	0
575	0
576	0
577	0
578	0
579	0
580	0

Continuation of Table 7	
variable	coefficient
581	0
582	0
583	0
584	0
585	0
586	0
587	0
588	0
589	0
590	0
591	0
592	0
593	0
594	0
595	0
596	0
597	0
598	0
599	0
600	0
End of Table	

B.4 Coefficients EN

Table 8: The coefficients for one of the simulations using EN

Begin of Table	
variable	coefficient
1	3,010461
2	2,892294
3	3,052786
4	3,015276
5	3,068465
6	2,964138
7	2,941848
8	2,963363
9	2,928604
10	2,999172
11	2,994384
12	2,98641
13	2,90279
14	2,948346
15	2,968901
16	2,8029
17	2,876565
18	3,063118
19	2,939367
20	3,016215
21	2,998622
22	2,922078
23	2,924082
24	2,916623
25	2,98439
26	2,908217
27	2,88573

Continuation of Table 8	
variable	coefficient
28	3,033446
29	2,934459
30	3,046321
31	2,860964
32	2,999917
33	2,972034
34	2,917284
35	2,937175
36	2,92007
37	2,944916
38	2,933949
39	3,014088
40	3,050735
41	2,927839
42	3,002232
43	3,00225
44	2,970086
45	2,877009
46	2,977308
47	2,933847
48	2,981263
49	3,040637
50	2,943181
51	3,03909
52	2,8974
53	2,891068
54	2,876949
55	2,90019
56	2,915214
57	2,993491

Continuation of Table 8	
variable	coefficient
58	3,021326
59	3,057585
60	3,03826
61	2,935406
62	2,952229
63	2,898145
64	2,973293
65	2,903468
66	2,920825
67	2,949535
68	2,958478
69	2,997151
70	2,867791
71	2,933351
72	2,918357
73	2,855118
74	2,958202
75	2,857739
76	2,982726
77	2,958387
78	2,976453
79	3,042642
80	2,911211
81	2,985867
82	2,978118
83	2,997991
84	3,013152
85	2,895777
86	2,983619
87	2,982586

Continuation of Table 8	
variable	coefficient
88	2,856795
89	2,910953
90	2,905985
91	2,913591
92	2,944458
93	2,960007
94	2,93827
95	2,925755
96	3,056504
97	2,951523
98	3,063621
99	2,988078
100	2,865823
101	2,843676
102	3,020708
103	2,959377
104	2,876056
105	2,882448
106	2,926122
107	2,997335
108	2,929825
109	2,854674
110	2,889643
111	2,844441
112	2,907244
113	2,955446
114	2,834533
115	3,006253
116	2,94365
117	2,981617

Continuation of Table 8	
variable	coefficient
118	2,985007
119	2,959468
120	2,928587
121	2,968854
122	2,999826
123	3,002867
124	2,963211
125	2,956698
126	2,948719
127	3,015117
128	2,806303
129	2,938958
130	3,020763
131	2,876651
132	2,968913
133	2,955707
134	3,033085
135	2,960786
136	2,956754
137	2,876555
138	2,942852
139	2,987294
140	2,973469
141	2,863541
142	3,024311
143	2,904927
144	3,014495
145	2,988884
146	3,018683
147	2,94596

Continuation of Table 8	
variable	coefficient
148	2,956849
149	3,03122
150	2,88745
151	3,120243
152	2,981694
153	2,913896
154	2,813139
155	3,014122
156	2,984541
157	3,011163
158	2,955683
159	3,06966
160	2,99613
161	2,939308
162	3,003036
163	2,953774
164	3,08788
165	2,966253
166	2,878796
167	2,905681
168	2,875967
169	3,019383
170	2,873456
171	2,925379
172	2,926154
173	2,956233
174	2,953092
175	2,953851
176	2,933813
177	2,91201

Continuation of Table 8	
variable	coefficient
178	2,978436
179	2,901817
180	2,958651
181	3,026574
182	2,949417
183	3,019499
184	2,903373
185	2,96074
186	3,019514
187	2,885832
188	2,940148
189	2,98227
190	2,886392
191	2,982398
192	2,950849
193	3,010815
194	2,91771
195	2,885805
196	2,839181
197	3,052135
198	3,07469
199	3,039092
200	2,925175
201	2,992543
202	2,872627
203	2,924435
204	2,829511
205	2,884708
206	2,994074
207	2,952915

Continuation of Table 8	
variable	coefficient
208	2,860276
209	2,840492
210	2,98791
211	2,973946
212	2,932092
213	2,965684
214	2,916517
215	2,985981
216	2,933049
217	2,955849
218	2,928604
219	2,968552
220	2,924461
221	2,980004
222	2,988634
223	2,942022
224	2,97661
225	2,879341
226	2,870667
227	2,900159
228	2,954743
229	2,951056
230	2,875161
231	2,96942
232	2,95408
233	2,91694
234	2,998633
235	3,021529
236	3,006506
237	2,92947

Continuation of Table 8	
variable	coefficient
238	2,973618
239	2,911312
240	2,882442
241	0,485848
242	0,036963
243	0,031225
244	0
245	0
246	0,147985
247	0,322436
248	0
249	0
250	0,235152
251	0
252	-0,00427
253	-0,55052
254	-0,2177
255	-0,36979
256	0,191486
257	0,068344
258	0,47195
259	0
260	0,221672
261	0,056852
262	-0,30205
263	0
264	0
265	0,016594
266	0
267	0

Continuation of Table 8	
variable	coefficient
268	0,170121
269	0
270	0,375281
271	0
272	0,23447
273	0
274	-0,55482
275	0
276	0,198654
277	0,265071
278	-0,11135
279	0
280	0,115517
281	0
282	0,182227
283	0
284	0,001973
285	0,305637
286	0
287	-0,02888
288	0
289	0,026231
290	0,202679
291	-0,07844
292	-0,15836
293	0
294	-0,20078
295	-0,02295
296	0
297	0,271805

Continuation of Table 8	
variable	coefficient
298	0,042074
299	0
300	0
301	0
302	0,176349
303	-0,14947
304	0,025728
305	-0,03698
306	0
307	0,041203
308	0
309	0,42749
310	-0,13551
311	-0,16721
312	-0,03397
313	-0,06788
314	0,285926
315	0,004618
316	-0,41809
317	-0,01042
318	0
319	0,141345
320	0
321	0,037057
322	-0,11321
323	0
324	0
325	0
326	0,278423
327	0

Continuation of Table 8	
variable	coefficient
328	0
329	0,10475
330	0,414752
331	0
332	-0,0053
333	-0,19779
334	0,058385
335	0
336	0
337	-0,0977
338	-0,0924
339	-0,40403
340	0,060374
341	0,034917
342	-0,00563
343	0
344	0,25889
345	-0,12896
346	-0,31567
347	-0,02636
348	0,219246
349	0,169034
350	0,080167
351	0,080688
352	-0,25698
353	-0,03801
354	-0,3925
355	0
356	-0,11257
357	0,15444

Continuation of Table 8	
variable	coefficient
358	0,271639
359	0
360	0
361	0
362	0
363	-0,23097
364	0,517474
365	-0,00996
366	-0,10974
367	-0,43221
368	0
369	0,400111
370	-0,16148
371	0,38106
372	-0,15082
373	0,431767
374	0,145242
375	-0,15882
376	-0,12928
377	0,12196
378	-0,26362
379	0,192737
380	0
381	-0,00287
382	-0,01077
383	0
384	-0,43369
385	0
386	0,350631
387	0,228229

Continuation of Table 8	
variable	coefficient
388	0,217288
389	-0,30143
390	0,031279
391	0
392	-0,00492
393	-0,23773
394	0,348685
395	0
396	0,501141
397	-0,37553
398	0
399	0,434365
400	-0,02467
401	0,203849
402	-0,1485
403	0
404	-0,52765
405	0
406	-0,07627
407	-0,09352
408	-0,33268
409	-0,01815
410	0
411	0
412	-0,47246
413	0
414	0,411065
415	0
416	0,267553
417	0

Continuation of Table 8	
variable	coefficient
418	-0,05274
419	0,102379
420	0
421	-0,13652
422	0,12445
423	-0,05329
424	0
425	-0,42676
426	-0,03907
427	0,282265
428	0
429	0
430	0
431	-0,32184
432	0,475829
433	0,049012
434	-0,44127
435	0,062859
436	0,296227
437	-0,46183
438	0
439	0
440	0
441	-0,09644
442	0
443	-0,2002
444	-0,36145
445	-0,2086
446	-0,66476
447	-0,21326

Continuation of Table 8	
variable	coefficient
448	-0,15619
449	0,166938
450	0,450434
451	-0,51778
452	0
453	0,001925
454	-0,10025
455	-0,38954
456	-0,46612
457	0
458	-0,01197
459	0,232006
460	-0,28049
461	-0,11519
462	0
463	-0,43118
464	0
465	0,0857
466	0,171302
467	0,187612
468	0
469	-0,16285
470	-0,13689
471	0,002417
472	0
473	-0,39498
474	0
475	-0,26821
476	0
477	-0,19958

Continuation of Table 8	
variable	coefficient
478	0
479	-0,07049
480	0
481	0
482	0
483	0,093534
484	0
485	-0,31097
486	-0,06825
487	0
488	-0,01207
489	0,042761
490	0
491	0
492	0,525652
493	-0,08927
494	-0,02443
495	0
496	0
497	0,057819
498	-0,04797
499	-0,16824
500	0,055251
501	-0,00598
502	0,014395
503	0
504	0
505	0
506	0,022903
507	0,118074

Continuation of Table 8	
variable	coefficient
508	0
509	0,231741
510	-0,103
511	0
512	0,193854
513	0
514	0
515	-0,05554
516	0,574329
517	0
518	0,015882
519	0,290309
520	0,394619
521	-0,01009
522	0
523	0
524	0,352417
525	0
526	0
527	0
528	0
529	0,253079
530	-0,14627
531	0
532	0
533	-0,09088
534	-0,05712
535	0,528117
536	0
537	0,006824

Continuation of Table 8	
variable	coefficient
538	-0,40107
539	-0,08014
540	0
541	0
542	0,301237
543	0
544	-0,26827
545	0,780553
546	-0,53792
547	0,064107
548	0
549	-0,56041
550	0
551	-0,16017
552	0
553	0
554	0,104999
555	0,056844
556	-0,68657
557	-0,13411
558	-0,04211
559	0,014022
560	0
561	0
562	-0,13146
563	-0,49893
564	-0,46833
565	-0,04299
566	0
567	-0,32787

Continuation of Table 8	
variable	coefficient
568	0,109658
569	0,510489
570	0
571	0
572	0
573	0,303123
574	-0,25498
575	0
576	0,035122
577	0,18463
578	0
579	0,186214
580	-0,54806
581	0,266758
582	0,26623
583	-0,07693
584	0
585	-0,07835
586	0,118636
587	0,203333
588	0,11666
589	-0,1205
590	0,071783
591	-0,02563
592	0,056049
593	-0,1454
594	0
595	0,215203
596	0
597	-0,23105

Continuation of Table 8	
variable	coefficient
598	0
599	0
600	1,009665
End of Table	

Appendix C Code

C.1 LAR and LASSO

Listing 1: Matlab code: programming LAR and LASSO

```
function [figure1, figure2, mu, mus, step, size_alpha] = LARS(X, y, maxsteps, type, figure1, figure2)
%This function receives the dependent and independent variables and a
%maximum amount of steps to take.
%For a type which is either lasso or lar, the function outputs the
%coefficients as well as either figure 1 or 2
%figure 1 and 2 correspond to the table 3a or b in the paper by
%Efron et al. (2004)
%the output consists of the optimal mu for every step, the amount of steps
%taken and the size of alpha for every step.
%this function implements the lars algorithm as introduced by Efron et al.
%(2004). All named formula's correspond to the formulas named in this
%paper.
%also the modifications to capture all LASSO regression outputs are
%implemented.

[k,n]= size(X);                                                    %initialising
X_trans= X.';
y_trans= y.';
mu=zeros([k 1]);
beta= zeros([n 1]);
beta_previous = zeros([n 1]);
alpha= [];
g_alpha =[];
A_alpha= [];
w_alpha =[];
u_alpha =[];
gamma=[];
a=[];
size_alpha=[];
step = [];
Absolutec= [];
Maximum_correlation=[];
betas=[];
betas=[betas beta];
betasum=[];
betasum=[betasum 0];
all_C=[];
mus=[];
```

```
i=1;
while i<=maxsteps && length(alpha)< n    %loop until the desired amount of
                                         %steps is reached or all variables
                                         %are used (inspired by R code)

    c_hat= X_trans*(y-mu);               %formula 2.8, calculates the
        '                                %current correlations for all variables

    C_optional=[1:n];
    r=length(alpha);
    q=1;
                                    %create the vector of correlation for
                                    %all variables not in alpha
    while q<=n
        u=1;
        in_alpha=0;
        while u<=r
            if q == alpha(u)
                in_alpha=in_alpha+1;
            end
            u=u+1;
        end
        if in_alpha==0
            C_optional(q)= c_hat(q);
        else
            C_optional(q)= 0;
        end
        q=q+1;
    end



    [C_hat, j]=  max(abs(C_optional));     %formula 2.9, obtain the maximum
                                %correlation and new variable to add to alpha
    alpha= [alpha j];

    r= length(alpha);
    ones= [];
    sj=[1:r];
    u=1;
    X_alpha=[];
    while u<=n
        q=1;
        while q<=r                      %update sj, X_alpha and the ones vector
            if u==alpha(q) && c_hat(u)>=0
                sj(q)= 1;                                          %2.10
                X_alpha(:,q)= 1*X(:,u);                            %2.4
                ones=[1 ones];
            elseif u==alpha(q)
                sj(q)= -1;                                %2.10
                X_alpha(:,q)= -1*X(:,u);                  %2.4
                ones=[1 ones];
            end
            q=q+1;
        end
        u=u+1;
    end

    X_alpha_trans=X_alpha.';
    ones_trans=ones.';

    g_alpha=  X_alpha_trans*X_alpha;                        %formula 2.5
```

```
invert_g_alpha= inv(g_alpha);

A_alpha1= (ones*invert_g_alpha*ones_trans);                %formula 2.5
A_alpha= 1/(sqrt(A_alpha1));

w_alpha= A_alpha*invert_g_alpha*ones_trans;  %formula 2.6, obtain the weights vector



u_alpha= X_alpha*w_alpha;                                  %formula 2.6


a= X_trans*u_alpha;                                %formula 2.11, the inner product



gamma= [];
q=1;
test=[];
while q<=(n -1)                                    %formula 2.13, calculate the gamma
    u=1;
    in_alpha=0;
    while u<=r
        if q == alpha(u)
            in_alpha=in_alpha+1;
        end
        u=u+1;
    end

    if in_alpha==0

        derde= (C_hat-c_hat(q))/(A_alpha - a(q));
        test= [test derde];
        if derde>0                                 %include only positive components
            gamma=[gamma derde];
        end
        zesde= (C_hat+c_hat(q))/(A_alpha +a(q));
        test= [test zesde];
        if zesde>0                                  %include only positive components
            gamma=[gamma zesde];
        end
    end
    q=q+1;
end

gamma_estimated=min(gamma);
if r==n                                      %on the last step gamma is different
    gamma_estimated=C_hat/A_alpha;
end




    q=1;
    while q<=n
        u=1;
        while u<=r
            if q == alpha(u)
                                               %calculate the coefficients
                beta(q)=beta(q) + gamma_estimated*sj(u)*w_alpha(u);
            end
            u=u+1;
```

```
        end
        q=q+1;
    end
    betas=[betas beta];

    beta_summed= sum(abs(beta));

    betasum=[betasum beta_summed];




if strcmp(figure2,'true')==1        %create a matrix with the absolute
                                    %correlation for all variables at each step
    currents=[];
    q=1;
    while q<=n                              %calculate the correlations in step i
        correlation= X_trans(q,:)*(y-mu);
        correlation=abs(correlation);
        currents= [currents correlation];
        q=q+1;
    end

    %create a vector of the maximum correlation per step
    Maximum_correlation= [Maximum_correlation ; max(currents)];

    Absolutec= [Absolutec ; currents];
    if i==maxsteps
        Absolutec=[Absolutec Maximum_correlation];
    end
end

if strcmp(type, 'LASSO')==1                         %The LASSO adaption
    D=zeros([n 1]);
    q=1;
    while q<=r
        D(alpha(q))=sj(q)*w_alpha(q);               %obtained from formula 3.2
        q=q+1;
    end

    track=[];
    gammas=[];
    q=1;
    count=0;
    while q<=n
        gammaj= (-1*beta(q))/D(q);     %formula 3.4, calculate the point of sign change
        if gammaj >0
            gammas=[gammas gammaj];
            count=count+1;
            track=[track q];
        end
        q=q+1;
    end

    [Gamma_test, j]=min(gammas);         %formula 3.5, obtain the alternative gamma
    if Gamma_test<gamma_estimated                   %or count==0
        q=1;
        to_remove=track(j);
        new_alpha=[];
        while q<=r                  %formula 3.4, remove the variable
                                    %where the sign changed occured from alpha
```

```
                if alpha(q) ~= to_remove
                    new_alpha= [new_alpha alpha(q)];
                end
                q=q+1;
            end
            alpha=new_alpha;
            mu= mu + Gamma_test*u_alpha;  %formula 3.6, updating mu if a LASSO step was made
        else                              %formula 2.12, updating mu if no LASSO step was made
            mu= mu + gamma_estimated* u_alpha;
        end
    else


        mu= mu + gamma_estimated* u_alpha;       %update mu if not LASSO
        end







    r=length(alpha);                                %creating output for the function
    size_alpha=[size_alpha r];
    step= [step i];
    mus=[mus mu];

    i=i+1;
end


betasum_trans= betasum.';
if strcmp(figure2 ,'true')==1                       %output for figure 3.2
    figure2 = plot(step,Absolutec,'DisplayName','Absolutec');
    xlabel('amount_of_steps_taken')
    ylabel('absolute_correlation')
end
if strcmp(figure1,'true')==1                        %output for figure 3.1
    figure1 = plot(betasum_trans,betas,'DisplayName','beta');
    xlabel('sum_of_the_absolute_coefficients')
    ylabel('value_of_the_coefficient')
end

end
```

C.2 Stepwise and Stagewise

Listing 2: Matlab code: programming Stepwise and Stagewise

```
function [mus, step] = stepwise(X, y, maxsteps, type)
%this function performs stepwise or stagewise regression
%this function receives the dependent and independent variables as input.
%As well as the maximum amount of steps to take and the type which is
%either stepwise or stagewise.
%The function outputs the predicted value at each reported step
%(every time a new variable is added to the model), as well as a vector
%containing 1 to 40.


[k,n]= size(X);                                 %initiating
X_trans= X.';
mu=zeros([k 1]);
mus=[];
epsilon=1;                                     %setting the stepsize for stagewise
step=[];
alpha=[];
size_alpha=[];
alpha_amount=0;
i=1;
while i<=maxsteps && length(alpha_amount)<=40  %a maximum of 40 variables
                                                %should be concluded in the model
    c_hat= X_trans*(y-mu);                      %calculating the current correlation
    C_optional=[1:n];
    q=1;
    r=length(alpha);
    while q<=n                                  %creating a vector containing
                            %only the correlations for not yet used variables
        u=1;
        in_alpha=0;
        while u<=r
            if q == alpha(u)
                in_alpha= 1;
            end
            u=u+1;
        end
        if in_alpha==0
            C_optional(q)= c_hat(q);
        else
            C_optional(q)= 0;
        end
        q=q+1;
    end

    if strcmp(type,'stepwise')==1
        [C_hat, j]=  max(abs(C_optional));      %if stepwise, then only a
                                        %variable not already used can be chosen
        alpha=[alpha j];
    else
        [C_hat, j]=  max(abs(c_hat)); %if stagewise, any variable can be chosen
        alpha=[alpha j];                        %alpha should be updated to
                            %keep track of the amount of variables in the model
    end
    if c_hat(j)>=0
        sign=1;
    else
        sign = -1;
```

```
    end

    if strcmp(type,'stepwise')==1                %setting the stepsize
        epsilon=C_hat;
    end


    mu=mu+epsilon.*sign.*X(:,j);                 %updating mu

    alpha_test= unique(alpha);
    if length(alpha_test)>length(alpha_amount)   %testing if a variable was
                                                 %added, then updating stagewise
        mus=[mus mu];
    end

    alpha_amount=alpha_test;

    step=[1:40];


    i=i+1;
end
end
```

C.3 Diabetes Simulation

Listing 3: Matlab code: The Simulation for Diabetes data

```
%This code creates 100 simulations of a data set an calculates the
%proportion explained for each computational step for LAR, LASSO, Stagewise
%and Stepwise for every simulation. Which is printed in a graph.

%obtaining the actual mu
[~, ~, mu_set, ~, ~, ~] = LARS(X1, y, 10, 'lars', 'false', 'false');
mut=mu_set;
epsilon = y - mut;

avg_proportion_explained_lars=[];                                    %initialising
avg_proportion_explained_LASSO=[];
avg_explained_stagewise=[];
avg_proportion_explained_stepwise=[];
avg_proportion_explained_stagewise=[];
avg_size_alpha_LASSO=[];
avg_size_alpha_stagewise=[];
average_size_alpha_stagewise=[];
randoms=[];
question=[];

i=1;                                                                 %simulate 100 times
while i<=100

    proportion_explained_lars=[];                          %initialising within the loop
    proportion_explained_LASSO=[];
    proportion_explained_stagewise=[];
    proportion_explained_stepwise=[];
    epsilon_star=[];

    random = randi([1 442],1,442);       %finding a vector of 442 random integers between 1, 442
    randoms=[randoms ; rng];             %saving the random values for the sake of reproduction

    q=1;
    while q<=442                    %by using the random vector, create a simulated error vector
        epsilon_star=[epsilon_star ; epsilon(random(q))];
        q=q+1;
    end
    y_star= mut+epsilon_star;                              obtain the simulated y

    [~, ~, ~, mus, ~, ~] = LARS(X, y_star, 40, 'lars', 'false', 'false');
                          %run 40 steps of lars and save the mu obtained in each step
    q=1;
    while q<=40                            %calulate the proportion explained for each step
        mu_minus=mus(:, q)-mut;
        pe= sum(mut.'*mut);
        se= sum(mu_minus.'*mu_minus);
        pe=1-se/pe;
        proportion_explained_lars=[proportion_explained_lars pe];
        q=q+1;
    end
    avg_proportion_explained_lars=[avg_proportion_explained_lars ; proportion_explained_lars];

    [~, ~, ~, mus, ~, size_alpha] = LARS(X, y_star, 40, 'LASSO', 'false', 'false');
    %run 40 steps of LASSO and save the mu and the amount of variables in alpha, in each step
    q=1;
    while q<=40                                  %calculate the proportion explained for each step
        mu_minus=mus(:, q)-mut;
```

```
        mu_minus_trans=mu_minus.';
        pe= sum(mut.'*mut);
        se= sum(mu_minus_trans*mu_minus);
        pe=1-se/pe;
        proportion_explained_LASSO=[proportion_explained_LASSO pe];
        q=q+1;
    end
    avg_proportion_explained_LASSO=[avg_proportion_explained_LASSO ; proportion_explained_LASSO];
    avg_size_alpha_LASSO=[avg_size_alpha_LASSO ; size_alpha];


    [mus, step] = stepwise(X, y_star, 6000, 'stagewise');
    %run untill 40 variables in the model
    q=1;

    while q<=40                          %calculate the proportion explained for each step
        mu_minus=mus(:, q)-mut;
        mu_minus_trans=mu_minus.';
        pe= sum(mut.'*mut);
        se= sum(mu_minus_trans*mu_minus);
        pe=1-se/pe;
        proportion_explained_stagewise=[proportion_explained_stagewise pe];
        q=q+1;
    end
    avg_proportion_explained_stagewise=[avg_proportion_explained_stagewise ; proportion_explained_stagewise];
    avg_size_alpha_stagewise=[avg_size_alpha_stagewise ; step];


    [mus, step] = stepwise(X, y_star, 1000, 'stepwise');
    %run 40 steps of stepwise and save the mu and the amount of variables in alpha, in each step

    q=1;
    while q<=40                              %calculate the proportion explained for each step
        mu_minus=mus(:, q)-mut;
        mu_minus_trans=mu_minus.';
        pe= sum(mut.'*mut);
        se= sum(mu_minus_trans*mu_minus);
        pe=1-se/pe;
        proportion_explained_stepwise=[proportion_explained_stepwise pe];
        q=q+1;
    end
    avg_proportion_explained_stepwise=[avg_proportion_explained_stepwise ; proportion_explained_stepwise];




    i=i+1;
end

final_proportion= [];                    %create the average vectors for 'figure5'

final_proportion=[mean(avg_proportion_explained_lars, 1) ; mean(avg_proportion_explained_LASSO, 1)] ;
final_proportion=[final_proportion ;  mean(avg_proportion_explained_stepwise, 1) ];
final_proportion=[final_proportion ; mean(avg_proportion_explained_stagewise, 1)];
final_steps=[];
final_steps= [step ; mean(avg_size_alpha_LASSO, 1) ;   step ; step];

plot(final_steps.', final_proportion.','DisplayName','final_proportion');  %figure5
xlabel('number_of_terms_included')
    ylabel('proportion_explained')
```

C.4 Simulation one

Listing 4: Matlab code: Generating Simulation one

```
    function [y, X]= simulation(step)
    %this function returns a simulated data set, for a given step in the
    %process. This simulation concurs with Simulation one
beta_1=3*ones([1 15]);   %relevant variables
beta_2=zeros([1 25]);     %irrelevant variables
beta=[beta_1 beta_2];
sigma=15;


    if step==1               %set the amount of observations
        n=500;
    elseif step==2
        n=50;
    else
        n=400;
    end
X=[];
epsilon= randn([n 1]);
i=1;
Z_1=randn([n 1]);            %create three sets of grouped variables
Z_2=randn([n 1]);
Z_3=randn([n 1]);
while i<=40
    if i<=5
        epsilon_i=0.1*randn([n 1]);
       x= Z_1 +epsilon_i;
    elseif i<=10
       epsilon_i=0.1*randn([n 1]);
       x= Z_2 +epsilon_i;
    elseif i<=15
       epsilon_i=0.1*randn([n 1]);
       x= Z_3 +epsilon_i;
    else
       x= randn([n 1]);
    end

    X=[X x];
    i=i+1;
end

if step==1
X=X;
y=X*beta.' +sigma*epsilon;                    %create the data set
elseif step==2
    X=X;
    y=X*beta.' +sigma*epsilon;
else
    X=X;
    y=X*beta.' +sigma*epsilon;
end


end
```

C.5 Simulation two

Listing 5: Matlab code: Generating Simulation two

```
    function [y, X]= simulation(step)
    %this function returns a simulated data set, for a given step in the
    %process. This simulation concurs with Simulation two
beta_1=3*ones([1 240]);          %relevant variables
beta_2=zeros([1 360]);           %irrelevant variables
beta=[beta_1 beta_2];
sigma=15;


    if step==1                   %set the amount of observations
        n=500;
    elseif step==2
        n=50;
    else
        n=400;
    end
X=[];
epsilon= randn([n 1]);
i=1;
Z_1=randn([n 1]);                %create three sets of grouped variables
Z_2=randn([n 1]);                %this procedure follows Hastie et al.
Z_3=randn([n 1]);
while i<=600
    if i<=80
        epsilon_i=0.1*randn([n 1]);
       x= Z_1 +epsilon_i;
    elseif i<=160
       epsilon_i=0.1*randn([n 1]);
       x= Z_2 +epsilon_i;
    elseif i<=240
       epsilon_i=0.1*randn([n 1]);
       x= Z_3 +epsilon_i;
    else
       x= randn([n 1]);
    end

    X=[X x];
    i=i+1;
end

if step==1                                   %the validation set/training/testing set
X=X;
y=X*beta.' +sigma*epsilon;
elseif step==2
    X=X;
    y=X*beta.' +sigma*epsilon;
else
    X=X;
    y=X*beta.' +sigma*epsilon;
end


end
```

C.6 Train EN

Listing 6: Matlab code: Training EN

```
function [alpha, lambda] = EN_training(X, y)
%This function trains EN on a data set and outputs optimal combinations of
%alpha and lambda for simulation two
alpha_mse=[];
betas=[];
lambdas=[];
y_1=[];
X_1=[];
y_2=[];
X_2=[];
y_3=[];
X_3=[];
y_4=[];
X_4=[];
y_5=[];
X_5=[];
y_6=[];
X_6=[];
y_7=[];
X_7=[];
y_8=[];
X_8=[];
y_9=[];
X_9=[];
y_10=[];
X_10=[];

random = randi([1 500],1,500);          %split the data set for crossvalidation
    i=1;
    while i<=500
        if random(i) <=50
            y_1=[y_1 ; y(i)];
            X_1=[X_1 ; X(i, :)];
        elseif 50< random(i) <=100
            y_2=[y_2 ; y(i)];
            X_2=[X_2 ; X(i, :)];
            elseif 100< random(i) <=150
            y_3=[y_3 ; y(i)];
            X_3=[X_3 ; X(i, :)];
            elseif 150< random(i) <=200
            y_4=[y_4 ; y(i)];
            X_4=[X_4 ; X(i, :)];
            elseif 200< random(i) <=250
            y_5=[y_5 ; y(i)];
            X_5=[X_5 ; X(i, :)];
            elseif 250< random(i) <=300
            y_6=[y_6 ; y(i)];
            X_6=[X_6 ; X(i, :)];
            elseif 300< random(i) <=350
            y_7=[y_7 ; y(i)];
            X_7=[X_7 ; X(i, :)];
            elseif 350< random(i) <=400
            y_8=[y_8 ; y(i)];
            X_8=[X_8 ; X(i, :)];
            elseif 400< random(i) <=450
            y_9=[y_9 ; y(i)];
            X_9=[X_9 ; X(i, :)];
```

```
        else
             y_10=[y_10 ; y(i)];
            X_10=[X_10 ; X(i, :)];
        end
        i=i+1;
    end
    X=[X_1;X_2;X_3;X_4;X_5;X_6;X_7;X_8;X_9;X_10];
    y=[y_1;y_2;y_3;y_4;y_5;y_6;y_7;y_8;y_9;y_10];

    alpha_val=[];
    lambda_val=[];
k=1;
while k<=10                          %crossvalidation
    alpha_test=[];
    lambda_test=[];
    mse_test=[];

    train_x=[];
    train_y=[];
    validate_X=X((k-1)*50+1 : k*50, :);
    validate_y=y((k-1)*50+1 : k*50);

    train_x=[train_x ; X(1:(k-1)*50, :) ; X(k*50: 500, :)];
    train_y=[train_y ; y(1:(k-1)*50) ; y((k)*50:500)];

    i=1;
    while i<=10                      %optimize lambda on 10 types of alpha
        alpha=i/10;
    [~,FitInfo] = lasso(train_x,train_y,'Alpha',alpha,'CV',10);
    %EN for given alpha and ten-fold cross validation
idxLambda1SE = FitInfo.Index1SE;
alpha_test=[alpha_test alpha];
lambda_test=[lambda_test FitInfo.Lambda1SE];

%validate the alpha lambda combination
[~,FitInfo] = lasso(validate_X,validate_y,'Alpha',alpha,'Lambda', FitInfo.Lambda1SE);
  %EN for given lambda and alpha
MSE_est= FitInfo.MSE;
mse_test=[mse_test MSE_est];         %find the MSE
        i=i+1;
    end

      [~, index]= min(mse_test);         %save the optimal lambda and alpha
        lambda_val=[lambda_val lambda_test(index)];
        alpha_val=[alpha_val alpha_test(index)];
   k=k+1;
end

alpha=alpha_val;
lambda=lambda_val;

end
```

C.7 Optimize EN

Listing 7: Matlab code: Optimizing EN

```
function [beta_en]=EN_validation(X,y, alpha, lambda)
 %this function inputs a data set, and combinations of alpha and lambda
 %and outputs the optimal EN coefficients for the data set
      i=1;
      betas=[];
      MSE=[];
      while i<=10    %perform EN for all alpha, lambda combinations
      [B,FitInfo] = lasso(X,y,'Alpha',alpha(i),'Lambda', lambda(i));
      MSE=[MSE FitInfo.MSE];
      betas=[betas B];
      i=i+1;
      end
      [~, index]= min(MSE);            %find the optimal combination
      beta_en=betas(:, index);         %output the coefficients

 end
```

C.8 Validation

Listing 8: Matlab code: Generating the MSE for models

```
function [mse] = validation(X, y, beta)
%using a data set and given coefficients to output the MSE
[k,~]= size(X);
estimate= X*beta ;
mse=1/k *sum((y-estimate).^2);
end
```

C.9 LAR with Risk

Listing 9: Matlab code: Generating The optimal LAR

```
function [beta]=LAR(y, X, maxsteps)
%this function works the same as the previous lar method
%however all irrelevant code is removed, only LAR remains
%it inputs a data set and a maximum of steps and outputs the optimal
%coefficients
[k,n]= size(X);
risk =[];
standard_y=std(y);
X_trans= X.';
y_trans= y.';
mu=zeros([k 1]);
beta= zeros([n 1]);
beta_previous = zeros([n 1]);
alpha= [];
g_alpha =[];
A_alpha= [];
w_alpha =[];
u_alpha =[];
gamma=[];
a=[];
size_alpha =[];
step = [];
Absolutec= [];
Maximum_correlation =[];
betas =[];
betas=[betas beta];
betasum =[];
betasum=[betasum 0];
all_C =[];
mus=[];

i=1;
while i<=maxsteps && length(alpha)< n

    c_hat= X_trans*(y-mu);

    C_optional=[1:n];
    r=length(alpha);
    q=1;

        while q<=n
            u=1;
            in_alpha=0;
            while u<=r
                if q == alpha(u)
                    in_alpha=in_alpha+1;
                end
                u=u+1;
            end
            if in_alpha==0
                C_optional(q)= c_hat(q);
            else
                C_optional(q)= 0;
            end
            q=q+1;
        end
```

```
[C_hat, j]=  max(abs(C_optional));
alpha= [alpha j];

r= length(alpha);
ones= [];
sj=[1:r];
u=1;
X_alpha=[];
while u<=n
    q=1;
    while q<=r
        if u==alpha(q) && c_hat(u)>=0
            sj(q)= 1;
            X_alpha(:,q)= 1*X(:,u);
            ones=[1 ones];
        elseif u==alpha(q)
            sj(q)= -1;
            X_alpha(:,q)= -1*X(:,u);
            ones=[1 ones];
        end
        q=q+1;
    end
    u=u+1;
end

X_alpha_trans=X_alpha.';
ones_trans=ones.';

g_alpha=  X_alpha_trans*X_alpha;
invert_g_alpha= inv(g_alpha);

A_alpha1= (ones*invert_g_alpha*ones_trans);
A_alpha= 1/(sqrt(A_alpha1));

w_alpha= A_alpha*invert_g_alpha*ones_trans;

    u_alpha= X_alpha*w_alpha;

a= X_trans*u_alpha;

gamma= [];
q=1;
test=[];
while q<=(n -1)
    u=1;
    in_alpha=0;
    while u<=r
        if q == alpha(u)
            in_alpha=in_alpha+1;
        end
        u=u+1;
    end

    if in_alpha==0

        derde= (C_hat-c_hat(q))/(A_alpha - a(q));
        test= [test derde];
        if derde>0
            gamma=[gamma derde];
        end
```

```
            zesde= (C_hat+c_hat(q))/(A_alpha +a(q));
            test= [test zesde];
            if zesde>0
                gamma=[gamma zesde];
            end
        end
        q=q+1;
    end

    gamma_estimated=min(gamma);
    if r==n
        gamma_estimated=C_hat/A_alpha;
    end

       q=1;
        while q<=n
            u=1;
            while u<=r
                if q == alpha(u)

                    beta(q)=beta(q) + gamma_estimated*sj(u)*w_alpha(u);
                end
                u=u+1;
            end
            q=q+1;
        end
        betas=[betas beta];

            mu= mu + gamma_estimated* u_alpha;
    r=length(alpha);
    size_alpha=[size_alpha r];
    step= [step i];
    mus=[mus mu];

     tussen= y-mu;                    %calculate the risk component for each step
    tussen2= sum(tussen.'*tussen);
     risky=(tussen2)/standard_y.^2 - k + 2*i;
    risk=[risk ;risky];

    i=i+1;
end

[~, index]= min(risk);               %output the coefficients for the minimal risk
beta=betas(:, index);
end
```

C.10 Run the LAR, EN simulation

Listing 10: Matlab code: Comparing the MMSE for EN and LAR

```
%this function works with either simulation one or two.
%for a certain amount of simulations (simulation one=10)
%(simulation two=3), the optimal model for LAR and EN are found and the MSE
%reported.
%over all simulations, the MMSE is calculated

mse_EN=[];
mse_LAR=[];
i=1;

while i<=amount_of_simulations

%Simulating a training set
[y, X]= simulation(1);
%training the LAR on the training set
[beta_lar]=LAR(y, X, max_steps);              %max_steps:set 500 for simulation two
%training EN on the training set
[alpha, lambda] = EN_training(X, y);
%optimizing EN on the training set
[beta_en]=EN_validation(X,y, alpha, lambda);

%validation
[y, X]= simulation(3);
[MSE_LAR] = validation(X, y, beta_lar);
mse_LAR=[mse_LAR MSE_LAR];
[MSE_EN] = validation(X, y,beta_en);
mse_EN=[mse_EN MSE_EN];

i=i+1;
end

MMSE_lar=median(mse_LAR);          %generating the MMSE for LAR and EN
MMSE_en=median(mse_EN);
```