Aarhus University

Decision Support Systems

PROJECT REPORT

Author:

Lasse Lildholdt
(201507170)
Stinus Skovgaard
(201507170)
Daniel Tøttrup
(201507170)
Johan Vasegaard
(201507170)
Frederik Madsen
(201504477)

Supervisor:

Christian Fischer Pedersen

9. maj 2020

Indhold

Figurer

Kapitel 1

INTRODUCTION

In this project report, the reader will be presented with problem solutions for the course Decision support systems. Throughout the solution the reader will achieve knowledge on several different subject within the main area. Each topic will be presented with the theory along with solutions for appropiate exercise to validate the presented theory.

The main topics which will be handled in this report will be as follows:

- Simple linear regression / Multiple linear regression
- Logisitic regression / Linear discriminant analysis
- Cross validation / Bootstrap
- Subset selection
- Shrinkage methods / DImension reduction methods

Kapitel 2

SIMPLE LINEAR REGRESSION / MULTIPLE LINEAR REGRESSION

The simple Linear Regression approach is a quick and simple method for fitting a line through a 2-dimensional dataset. It is assumed that there is a approximately linear relationship between the two dimensions. This can be written mathematically as:

$$Y \approx \beta_0 + \beta_1 * X \tag{2.1}$$

eq. (2.1) can also be seen as "Regressing Y onto X". As an example the dataset Advertising.csv contains sales og a certain product and advertisement money spent on certain media platforms. X represents TV advertising and Y represents sales. It is the possible to regress sales onto TV.

In order to do this, we need to calculate the constants β_0 and β_1 which represents the intercept and slope terms in the linear model.

Kapitel 3

LOGISITIC REGRESSION / LINEAR DISCRIMINANT ANALYSIS

Kapitel 4

CROSS VALIDATION / BOOTSTRAP

Kapitel 5

SUBSET SELECTION

This chapter will address the subject Subset selection. Subset selection concerns with selecting or shrinking the coefficients of features to make the model more interpretable and in some cases to predict better.

Linear models are simple and can be interpreted because it usually has a small number of coefficients. In cases where the number of predictors is bigger than the number of samples, we can't use the full least squares, because the solutions is not even defined. In such cases we must reduce the number of features to be able to obtain a solution. It is also very important to not fit your data too hard which regularize, or selection of features also helps with. Along the same lines, when we have a small number of features the model becomes more interpretable.

There exist many different methods to perform subset selection. One of the methods to select the most important features regarding a specific response is called best subset selection. This is where we identify a subset of the predictors that is most related to the response.

5.1 Best subset selection

Best subset selection algorithm is a simple algorithm used to understand which predictors are mostly linked to the responds. To perform best subset selection, we fit a separate least squares regression for each possible combination. It starts out by fitting all p models that contain exactly one predictor, then fitting all p models that contain exactly two predictors and so forth. The number of models to fit can be calculated like this:

$$(\frac{p}{k}) = \frac{p!}{k!(p-k)!} \tag{5.1}$$

Where p is the number of all predictors, and k is the number of predictors in the subset. After all possible combinations of p models has been fitted, we then look at the resulting model, with the goal of identifying the best one.
Best subset selection algorithm can be divided into three steps:

1. Let M_0 denote the *null model*, which contain no predictors. This model simply predicts the sample mean for each observation.

2. For k=1,2,...p:

    (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.

    (b) Pick the best among these $\binom{p}{k}$ models, and call it M_k. Here *best* is defined as having the smallest RSS, or equivalently largest R^2

3. Select a single best model from among M_0,...,M_p using crass-validated prediction error, AIC, BIC or adjusted R^2

The task of to selecting the best subset model, must be performed with care, because RSS decreases monotonically and the R^2 increases monotonically, as the number of features included in the models increases. So, if we use these statistics, we will always end up med the model involving all the variables. Another problem with a low RSS or a high R^2 is that it only indicates a model with a low training error. And a low training error does not equal a low test error. So, for those reasons we need to use cross-validation, AIC, BIC or adjusted R^2.

5.2 Stepwise Selection

If the number of predictors p gets too large, best subset selection algorithm cannot be applied for computational reasons. Another reason why best subset selection is not always the best algorithm to select at subset, is for statistical reasons if the p is too large. Then there is a higher chance of finding models that look good on the training data, even though they might not have any predictive power on future data. For these reasons, stepwise methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection.

In this section we'll touch upon two different stepwise selection methods; the forward stepwise selection method and the backwards stepwise selection method.

5.2.1 Forward stepwise selection

Forward stepwise selection method is very similar to the best subset selection method. Like the best subset selection method, the forward stepwise selection also begins with a model containing no predictors, and adds one predictor at a time until all predictors are in the model. But unlike the best subset selection method the forward stepwise selection doesn't look at all possible models that contains k predictors at each step. Instead, we are just looking at the models that contain the k-1 predictors that already were chosen in the previous step, plus one more. This means that at the k-th step, we are looking at a much more restricted set of models compared to the best subset selection method.
Forward stepwise selection method can be divided into three steps:

1. Let M_0 denote the *null model*, which contain no predictors.

2. For k=0,...,p-1:

   (a) Consider all p-k models that augment the predictors in M_k with one additional predictor.

   (b) Chose the best among these p-k models, and call it $M_k + 1$. Here *best* is defined as having the smallest RSS, or highest R^2

3. Select a single best model from among M_0,...,M_p using crass-validated prediction error, AIC, BIC or adjusted R^2

Compared to best subset selection forward stepwise selection has a computational advantage, as we consider 2^p models in best subset selection and only p^2 models in forward stepwise selection method. This relates to the fact that forward stepwise selection is not guaranteed to find the best possible model out of all 2^p models containing subset of p predictors.

5.2.2 Backward stepwise selection

Backward stepwise selection is exactly the opposite of forward stepwise selection. In contrast backward stepwise selection begins with the full least squares model containing all p predictors, and then removes the least useful predictor, one at a time.
Forward stepwise selection method can be divided into three steps:

1. Let M_p denote the *full model*, which contain all p predictors.
2. For k=p,p-1,...,1:
   (a) Consider all k models that contain all but one of the predictors in M_k, for a total of k-1 predictors.
   (b) Chose the best among these k models, and call it $M_k - 11$. Here *best* is defined as having the smallest RSS, or highest R^2
3. Select a single best model from among M_0,...,M_p using crass-validated prediction error, AIC, BIC or adjusted R^2

Just like forward stepwise selection, backwards stepwise selection is not guaranteed to gives us the best model containing a particular subset of p predictors.

The major difference between forward and backwards stepwise selection is that the number of samples needs to be larger than the number of variables to perform the backward stepwise selection method (so its possible to fir a least squares model). This is not case for forward stepwise selection, it can be applied when n<p and p>n.

5.3 Choosing the optimal model

As already explained RSS and R^2 are not suitable for selecting the best model among a collection of models, because these quantities are related to the training error and not the test error. So, in order to select the best model with respect to the test-error there are two common approaches:

1. Indirectly estimate the test error by making an adjustment to the training error to account for the bias due to overfitting.
2. Directly estimate the test error by either using a validation set approach or a cross-validation approach.

In this section we'll only talk about AIC, BIC and adjusted R^2 which all are indirectly estimate of the test error.

AIC

AIC stand for Akaike information criterion and deals with the trade-off between goodness of fit of the model and simplicity of the model or it deals with the risk of overfitting and underfitting.

$$AIC = \frac{1}{n\hat{\sigma}^2}(RSS + 2d\hat{\sigma}^2) \tag{5.2}$$

The best model according to the AIC is where the AIC is smallest.

BIC

BIC stands for Bayesian information criterion. This is closely related to the AIC. Both the BIC and the AIC introduces a penalty term for number of parameters in the model to resolve the problem of overfitting. The penalty term is larger in the BIC than in the AIC.

$$BIC = \frac{1}{n}(RSS + log(n)d\hat{\sigma}^2) \tag{5.3}$$

The best model according to the BIC is where the BIC is smallest.

Adjusted R^2

Another very popular approach for selecting among a set of models that contain a different number of variables. R^2 is defined as 1-RSS/TSS, where TSS is the total sum of squares for the response. But as already mentioned R^2 keeps increasing as more variables are added to the model. The adjusted R^2 is calculated as:

$$Adjusted R^2 = 1 - \frac{RSS/(n-d-1)}{TSS/(n-1)} \tag{5.4}$$

Unlike the AIC and BIC, a large value for adjusted R^2 indicates a model with a small test error.

5.4 Lab 6.5.1

In Lab 6.5.1 we have implemented best subset selection and performed it on the "Hitters"-dataset, to predict a baseball players salary based on various statistics. This has been implemented in python and will be discussed briefly in this section, for a more detailed look into the implementation see the source-code in the appendix.

In this section only the heart of the implemented algorithm will be presented. The function getBestModel returns the model with the lowest RSS, based on all combinations of k number of features.

```
def getBestModel(k, X, y):
        results = []
        # Goes through all combinations of k number of features
        for combination in itertools.combinations(X.columns, k):
        results.append(subset_Process(combination, X, y))

        # Stores all combinations i a single dataframe
        models = pd.DataFrame(results)
        #Select the model with the lowest RSS
        best_model = models.loc[models['RSS'].argmin()]

        return best_model
```

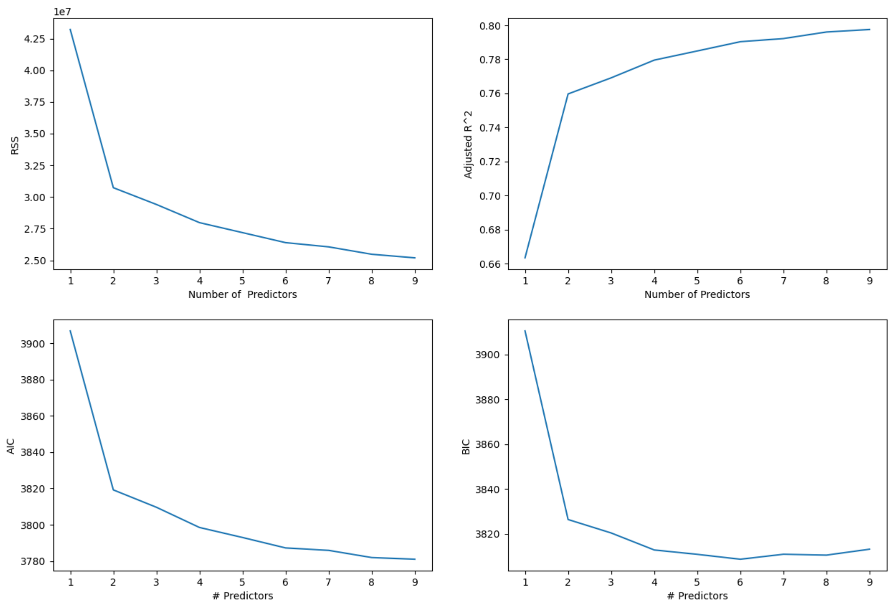
The function subset_Process is called by the getBestModel function described above. This function performs linear regression on a model and calculates the RSS the given model.

```
def subset_Process(predictor, X, y):
        # sm.OLS is an estimator that performs linear regression on a model
        temp_model = sm.OLS(y, X[list(predictor)])
        model = temp_model.fit()

        #Then calculate the RSS for the chosen model, and return the model and its RSS together
        RSS = ((model.predict(X[list(predictor)]) - y) ** 2).sum()
        return {"model": model, "RSS": RSS}
```

These two functions are repeated a lot of times to estimate all possible combinations of k predictors.

On fig. 5.1 are the results plotted. Because the number of combinations increases dramatically as k increases so does the computation time. For this reason, k is equal to nine. We can see that according to the BIC, the model with 6 variables performs the best. But according to Adjusted R^2 and AIC a model with more variables than six might be better. Again none of these measures gives us an entirely accurate picture, but they all agree that a model with fewer than five predictors is insufficient.



Figur 5.1: Lab 6.5.1 plotted results

Kapitel 6

SHRINKAGE AND DIMENSION REDUCTION METHODS

Contrary to subset selection methods, which uses least squares to fit a linear model containing a subset with n of all p predictors, shrinkage methods fits a model containing all p predictors. This is done by a penalty that regularizes the coefficient estimates and thereby shrinks them towards zero. By shrinking the coefficient estimates their variance can be significantly reduced. In this section the two shrinkage methods Ridge regression and The Lasso will be covered.
Both shrinking methods tries to control variance by either using a subset of the original variables or by shrinking their coefficients towards zero and uses all of the original predictors. Another class of approaches, dimension reduction, is one that transforms the predictors first and then fits a least squares model using the transformed predictors. The reduction comes from the fact that the methods reduces the problem of estimating $p + 1$ coefficients to estimating $M + 1$ where $M < p$. In this section the two dimension reduction methods Principal Component Regression and Partial Least Squares will be covered.

6.1 Ridge regression

Ridge regression is very similar to least square fitting in that it seeks to minimize RSS, but it adds a second term called the shrinkage penalty.
The equation 6.1 shows the full equation for ridge regression. Here the first term is the RSS where $\beta_0, ..., \beta_p$ is to be estimated such that it is minimized, but the second term introduces a penalty to β_j which effectively shrinks it towards zero. This penalty is scaled with λ such that as it moves towards zero, the penalty moves towards zero and the equation produces the least squares estimates. Moving the tuning parameter towards infinity will in turn drive the coefficient estimates towards zero.
Thus ridge regression will produce a different set of coefficient estimates, $\hat{\beta}^R_\lambda$, for each value of λ hence making it a tuning parameter. It is then critical to choose a good value for λ, which can be done e.g. by the cross-validation method.

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_i j)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 \tag{6.1}$$

The advantage of ridge regression over standard least squares comes from the bias-variance trade-off. By increasing the value of λ and thereby the effect of the penalty term, it is possible to decrease the variance of the predictor however the bias increases. As the test MSE (mean squared error) is a function of the variance plus the squared bias, finding a λ value which decreases the variance more than it increases the bias can result in a lower test MSE. Thus ridge regression will be superior whenever the least squares estimates have high variance.

6.2 The Lasso

The Lasso improves upon a disadvantage of ridge regression, namely that it includes all p predictors in the final model, because it only reduces the magnitudes of the coefficients but never actually excludes any of them. Where ridge regression uses the ℓ_2-norm in its regularization term (equation 6.1) , the Lasso uses ℓ_1-norm (equation 6.2). This has the effect of forcing some of the coefficients to be zero, depending on te value of λ, instead of only driving them towards zero. Therefore the Lasso acts like subset selection, as it effectively performs variable selection yielding a sparse model which exactly is its improvement over ridge regression.
Selecting a good value for λ is critical just as it is for ridge regression and can be done by cross-validation.

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_i j)^2 + \lambda \sum_{j=1}^{p}|\beta_j| \tag{6.2}$$

6.3 Principal Component Regression

As with all dimension reduction methods Principal Component Regression (PCR) works in two steps: first transformed predictors are obtained and then a model is fit using the transformed predictors.
The underlying method for obtaining the transformed predictors is in this case PCA (Principal Component Analysis). PCA derives a low-dimensional set of features from a large set of variables by acquiring basis vectors, *principal components*, that forms an orthogonal basis. This is done by finding vectors with values that minimizes the sum of squared perpendicular distances between each point and the vector. Finding the next principal component is done as a linear combination of the variables that is uncorrelated with the principal component before it and has the largest variance. Up to $M <= p$ principal components can be constructed this way, with p being the number of predictors. By this construction the first principal component will contain the most information of the data-set with each following principal component containing less and less information.
Fitting a least square model to $Z_1, ..., Z_M$ principal components instead of $X_1, ..., X_p$ data points with $p << M$, can in theory lead to better results as it can mitigate overfitting. This stems from the notion that most or all of the information in the data relating to the response if contained in the $Z_1, ..., Z_M$ principal component. If however $p == M$ PCR will perform the same as doing least square fitting on all of the original predictors.
In PCR the number of principal components M used for least square fitting becomes a hyper parameter that needs to be chosen carefully. This is typically done using the cross-validation method.

6.4 Partial Least Squares

In PCR the M principal components is guaranteed to best explain the predictors, however they are not guaranteed to best explain the response. This is due to the unsupervised nature of PCR, where the response does not supervise the identification of the transformed features (principal components).

Unlike PCR, Partial Least Squares (PLS) is supervised in the sense that the response is used to identify the transformed features that not only approximates the original predictors but also relates to the response.
PLS first identifies a new set of features $Z_1, ..., Z_M$ that represents $M < p$ linear combinations of the original p predictors. This is done by first standardizing the p predictors. Then the first translated feature Z_1 is calculated as in equation 6.3, with the difference that each θ_{j1} is set equal to the coefficient from the simple linear regression of the response Y onto X_j. In doing this tweaked version PLS places a higher weight on variables that are strongly related to the response. To compute Z_2 the residuals from regressing each variable on Z_1 are then used following the same calculations as for determining Z_1. Lastly least squares as in equation 6.4 is used to fit a linear model to predict Y using $Z_1, ..., Z_M$ translated feature instead of the original p features just like in PCR.

$$Z_m = \sum_{j=1}^{p} \phi_{jm} X_j \tag{6.3}$$

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m Z_{im} + \epsilon_i \tag{6.4}$$