Predictive Modeling Process

Austin Carango, Manuel Horta, Jared Wilber

Abstract

The objective of this project is to perform a predictive modeling process on the Credit dataset. This dataset reports four-hundred observations of twelve variables. The methods discussed are primarily based on chapter 6: Linear Model Selection and Regularization (from "*An Introduction to Statistical Learning*"" by James et al.). We discuss ridge regression, lasso regression, principal components regression, and partial least squares regression.

Introduction

Given a data set containing information pertaining to credit card balances and other factors such as income, race and gender, how could one develop a model to predict one's credit card balance?

To answer this question, the following paper will discuss several different regression methods used in the predictive modeling process. The specific methods discussed are: OLS, ridge, LASSO, PCA, and PLS. Each method will be discussed and applied separately to the Credit data. More information regarding these methods is available in sections 6.2, 6.3, 6.6, and 6.7 of chapter 6: Linear Model Selection and Regularization (from " *An Introduction to Statistical Learning*" by James et al).

We will attempt to model credit card balance as a function of 10 variables: `Income`, `Limit`, `Rating`, `Cards`, `Age`, `Education`, `Gender`, `Student`, `Married`, and `Ethnicity`. Because we are modelling this relationship via linear regression, we are inherently assuming that a linear relationship exists between our dependent variables and independent variable. We will run a total of five regressions: one for each regression method.

Data

The data set utilized in this paper, `Credit.csv`, contains 400 observations of 12 variables. Each observation corresponds to one person, and 11 of the variables are of general interest: `Income`, `Limit`, `Rating`, `Cards`, `Age`, `Education`, `Gender`, `Student`, `Married`, `Ethnicity`, and `Balance`. The variable `X` is an index.

This dataset is free to access online at the following url http://www-bcf.usc.edu/~gareth/ISL/Credit.csv. Alternatively, you can visit the author's GitHub repository My Github Account to access it as well.

For this paper, we will use the following variables: `Income`, `Limit`, `Rating`, `Cards`, `Age`, `Education`, `Gender`, `Student`, `Married`, `Ethnicity`, and `Balance`. `Income` refers to the person's income in thousands of dollars. `Limit` refers to the person's credit card limit in dollars. `Rating` refers to the person's credit rating. `Cards` refers to the number of cards the person has. `Education` refers to the number of years of education the person has completed. `Student` and `Married` are logical, answering the question of whether the person is currently a student and whether they are married. `Ethnicity` can take the values "Asian", "Caucasian", or "African American". `Balance` refers to the person's credit card balance in dollars. `Age` and `Gender` are self-explanatory.

In our analysis, we will treat `Balance` as the dependent variable and the other 10 variables of interest as the independent variables. When fitting the models, however, we factorized the categorical data, transforming `Gender`, `Student`, `Married`, and `Ethnicity` into `GenderFemale`, `StudentYes`, `MarriedYes`, `EthnicityCaucasian` and `EthnicityAsian`. Thus we have 11 independent variables in the models when fitting the models.

Methods

6.2 Shrinkage Methods: 1. Ridge Regression 2. Lasso 3. Ridge vs Lasso 4. Special case 5. Selecting parameter

6.3 Dimension Reduction Methods:
1. Principal Components Regression
1.1 PCA
1.2 Regression on PCA
2. Partial Least Squares

6.2

we can fit a model containing all p predictors using a technique that constrains or regularizes the coefficient estimates, or equivalently, that shrinks the coefficient estimates towards zero. It may not be immediately 6.2 Shrinkage Methods 215 obvious why such a constraint should improve the fit, but it turns out that shrinking the coefficient estimates can significantly reduce their variance. The two best-known techniques for shrinking th

The first class of techniques we will discuss fall into the domain of so-called "shrinkage" methods. These are techniques that allow us to fit a model with all p predictors via a regularization method that *shrinks* the coefficient estimates towards zero. We will discuss two regularization methods: *ridge regression* and the *lasso*.

Ridge Regression

Ridge regresion builds heavily off original least squares regression, If you'll recall, the RSS for original least squares is as follows:

$$RSS = \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2$$

In ridge regression, we add a *regularization* parameter to this equation, yielding the following quantity""

$$RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

This is the new equation that we minimize to find our coefficients. In effect, this quantity allows our regression model to predict observations that have more generalizability.

As with OLS, the above minimization equation hope to find the β coefficients that best fit the data. The additional parameter, the λ parameter, i known as the *shrinkage* parameter. It becomes small when the β coefficients are almost zero, hence the term *shrinkage*. To control the relative impact each of the two term in our new model has on our coefficient estimates, we use the parameter λ. When $\lambda = 0$, our equation simplifies to the original RSS. When $\lambda to inf$, the impact of the shrinkage parameter gets stronger, making our coefficient approach 0. Thus, our coefficient estimates vary as we tweak λ. Note, also, that this shrinkage parameter is not applied to our intercept term, as the goal of the minimization is to shrink the association between our predictors and the response.

So why is any of this important? Why don't we just continue employing original least squares for analysis if it works? The primary advantage of ridge regression over least squares is a result of the bias-variance trade-off, which plays into ridge regression as follows: increasing λ decreases the variance of our model but increases the bias. Decreasing λ increases the variance while decreasing the bias. Because original least squares uses $\lambda = 0$, it has a high variance with no bias. Therefore, at the expense of a slight increase in bias we can substantially reduce the the variance in the predictions. This shrinking of coefficient values disallows the opportunity for

one predictor to have too strong an effect on our model. This helps to curb overfitting and leads to a more accurate, generalizable model.

Additionally, least squares estimates is a very poor strategy for the case when the number of features is greater than the number of observations ($p > n$), whereas ridge will still perform well by trading variance for bias. Ridge regression is also more computationally efficient than subset selection methods, as only a single model fit is necessary.

LASSO

Least Absolute Shrinkage and Selector Operator, more commonly known as *LASSO*, is a regression method similar to ridge regression. The primary difference between LASSO and ridge is that LASSO will actually shrink some coefficients down to zero, while a ridge regression model fit with p predictors will maintain all p predictors. In this way, LASSO often results in a smaller model than ridge regression and can be looked at as a feature selection method. LASSO gets this property because of its choice of penalty:

$$RSS + \lambda \sum_{j=1}^{p} |\beta_j|$$

In fact, the only difference between ridge and LASSO is the norm choice in the regularization term. While ridge shrink the coefficients towards zero, no coefficients actually achieve a quantity of zero. On the other hand, coefficients in LASSO will beceome zero, resulting in a much more concise and interpretable model. Technically speaking, we say that LASSO yields a sparse model (a model with only a subset of variable).

Both ridge regression and LASSO rely on the parameter λ. Clearly, then, our choice of λ is very important for a robust analysis - so how do we choose it? For this analysis, we'll use cross-validation, a common re-sampling technique wherein which we determine a model performance based on multiple iterations of dividing our training data into folds of training and test sets. We'll pick the value of λ that results in the lowest error.

Principal Components regression (PCR)

The third method of our analysis is principal components regression, which is a popular approach to obtain a low dimensional feature set from a dataset with a high number of dimensions. Principal components regression works by first performing principal compnonents analysis (PCA) on the data, then performing a regression on the obtained principal component features.

Recall, PCA is a dimensionality reduction technique that creates a new, uncorrelated feature set where each variable is a linear combination of variables in the previous dimension. Furthermore, the principal components are ordered by the amount of variance captured, so choosing the number of features to use is as easy as setting a threshold for captured variance and picking the number of components corresponding to that much captured variance.

Principal components regression is a straightforward application from PCA: we construct a dataset with k chosen principal components and use them as feature/predictors in linear regression model. We then fit said model with least squares. The idea behind this analysis is that the principal components are often sufficient enough to explain the majority of the variability in the data while also capturing the relationhip between the predictors (principal components) and the response. It should be mentioned that it is not always the case that our components hold an associataion between the predictors and response, but it is often the case.

A benefit of principal components regression is that it often has better results than original least squares. Moreover, because we estimate a smaller number of coefficients, we are far less prone to overfitting. That said, principal components regression lacks in that its results lack interpretability; knowing how the response changes given some combination of all of our predictor is hardly intuitive.

Prior to performing PCR, predictors should be standardized, otherwise high-variance variables will play a larger role than warranted.

Partial Least Squares (PLS)

Partial least squares regression is a regression technique that attempts to overcome the shortcomings of PCA regression. It is very similar to PCA regression but addresses the problem inherent to PCA regression of using prinicipal components that are possibly not related to the response value. In this way, PLS regresion can be viewed as the supervised alternative to the unsupervised PCA regression.

As with PCA regression, PLS regression first reduces the dimensions of the data by creating linear combinations of the dataset's features. However, PLS regression identifies the features in a supervised manner wherein which the response value is utilized in such a way that our newly created features are related to the response; i.e. the PLS method finds directions that explain both the response and the predictors.

In PLS, we directly utilize the simple regression coefficient for each variable, as this coefficient is proportionally correlated to the response value. For this reason, PLS will place the highest weight on variables that are most strongly related to the response value.

PLSR generates the first directional vector by computing correlation between the feature space and the calculated regression coefficient of Y regressed onto its predictors. This has the effect of placing the highest weight on the most correlated predictors. Following this, second direction is created by regressing each predictor onto the previously calculated first direction and taking the residuals, (intuitively this represent the informatino not explaine dby the first component). The second direction is thus computed from these redsiduals.We continue selecting directions by iteratviely carrying out the following procedure: regress each variable on the former direction, then use the residuals to find the next direction.

Analysis

In this section we will show the processes used to perform each of the five regression methods: Ordinary Least Squares, Ridge regression, LASSO, PCR, and PLS. We will also display plots for each regression. Each regression model was fitted on the training data and we identified the best model. Using this best model we fit it again on the testing data and calculated the MSE value. Finally, we fit the best model again on the entire dataset for comparison. Before running these regressions, we factorized categorical variables and then standardized the data.

OLS Regression

```
# Fit Model

ols<-lm(Balance~Income+Limit+Rating+Cards+Age+Education
        +GenderFemale+StudentYes+MarriedYes+EthnicityAsian
        +EthnicityCaucasian,
        data=as.data.frame(scaled_credit))

summary_ols <- summary(ols)
mse_ols <- mean(ols$residuals^2)

#sink(file = "data/model-results/ols-results.txt")
summary_ols
paste("MSE")
mse_ols
#sink()
```

```
#save(ols, summary_ols, mse_ols,
 #    file ='data/ols-saved-objects.Rdata')
```

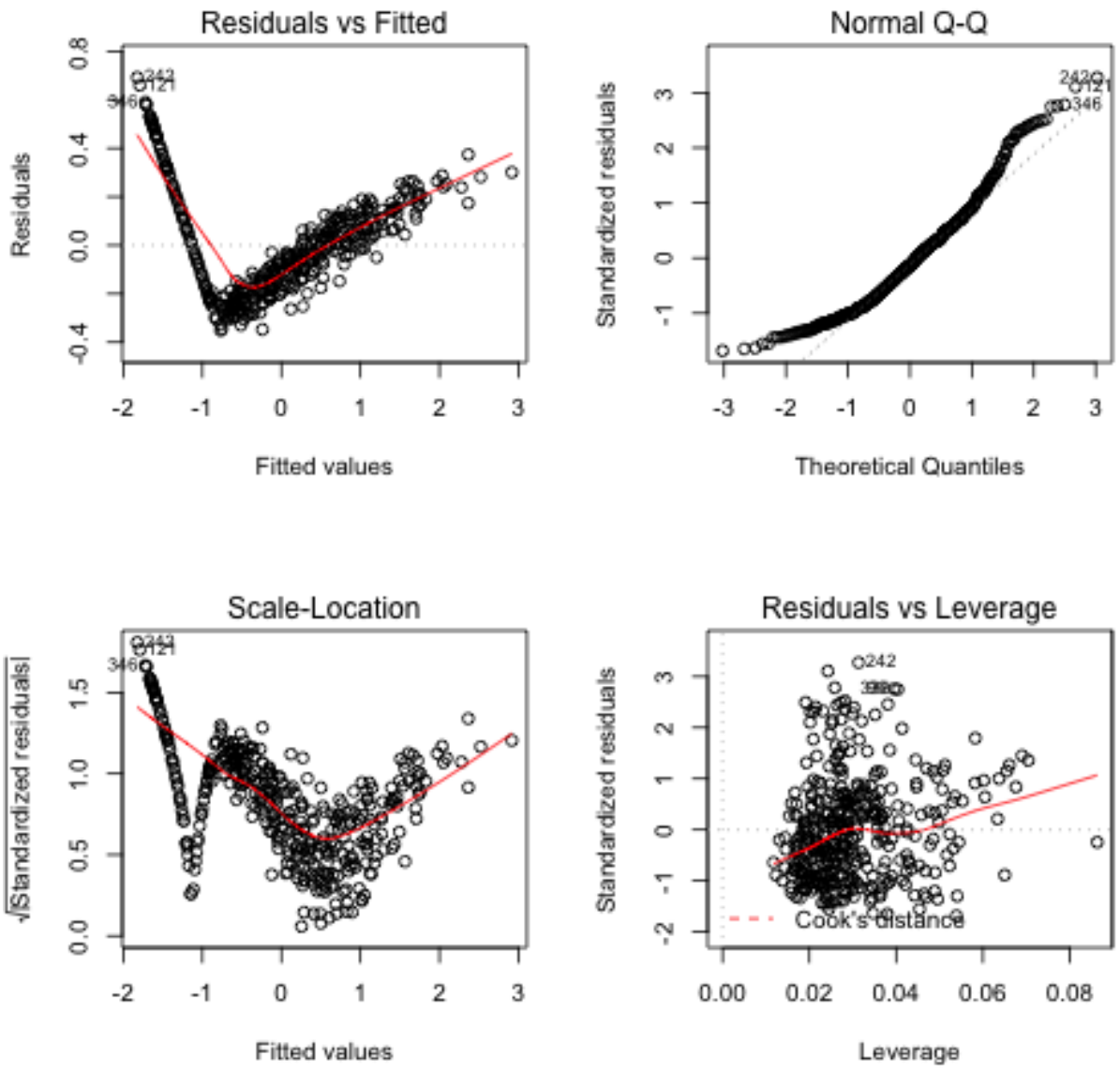


Figure 1:

Ridge Regression

```
library(glmnet)
set.seed(1)

# Fit Model
```

```r
ridge_mod <- cv.glmnet(x, y, intercept = FALSE, standardize = FALSE, lambda = grid, alpha = 0)

# Find best model

ridge_best_labmda <- ridge_mod$lambda.min

# Find MSE on test set

ridge_pred <- predict(ridge_mod,
                      s= ridge_best_labmda,
                      newx= model.matrix(Balance ~ ., data = data.frame(credit_test)))

ridge_mse <- mean((ridge_pred - credit_test[,12])^2)

# Refit on full dataset

ridge_fit_full <- cv.glmnet(model.matrix(Balance ~ .,data = data.frame(scaled_credit)),
                            data.frame(scaled_credit)$Balance,
                            intercept = FALSE,
                            standardize = FALSE,
                            alpha = 0)
```

LASSO Regression

```r
library(glmnet)
set.seed(1)

# Fit Model

grid = 10^seq(10, -2, length = 100)

set.seed(1)

x <- model.matrix(Balance ~ ., data = data.frame(credit_train))
y <- data.frame(credit_train)$Balance

lasso_mod <- cv.glmnet(x, y, intercept=FALSE, standardize=FALSE, lambda = grid, alpha=1)
summary(lasso_mod)


lasso_best_mod <- lasso_mod$lambda.min

# Preditct and MSE

lasso_pred <- predict(lasso_mod, s = lasso_best_mod,
                      newx= model.matrix(Balance ~ ., data = data.frame(credit_test)))
lasso_mse <- mean((lasso_pred - credit_test[,12])^2)

# Refit on full dataset

lasso_fit_full <- cv.glmnet(model.matrix(Balance ~ .,data = data.frame(scaled_credit)),
```

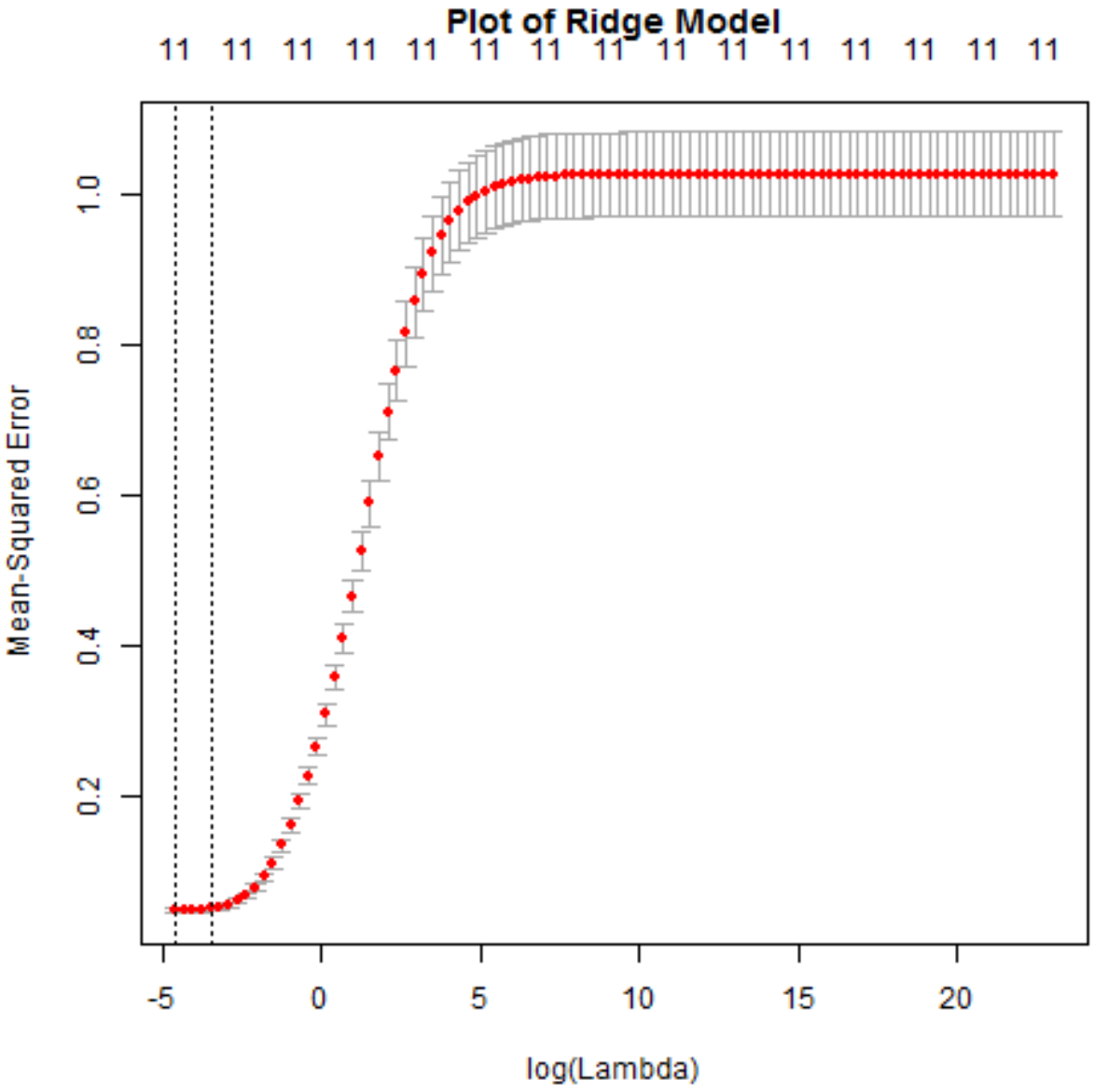

Plot of Ridge Model
11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
Mean-Squared Error
0.2
0.4
0.6
0.8
1.0
-5
0
5
10
15
20
log(Lambda)

Figure 2:

```
                        data.frame(scaled_credit)$Balance,
                        intercept = FALSE,
                        standardize = FALSE,
                        alpha=1)


(lasso_pred_full <- predict (lasso_fit_full, type="coefficients", s = lasso_best_mod))
```

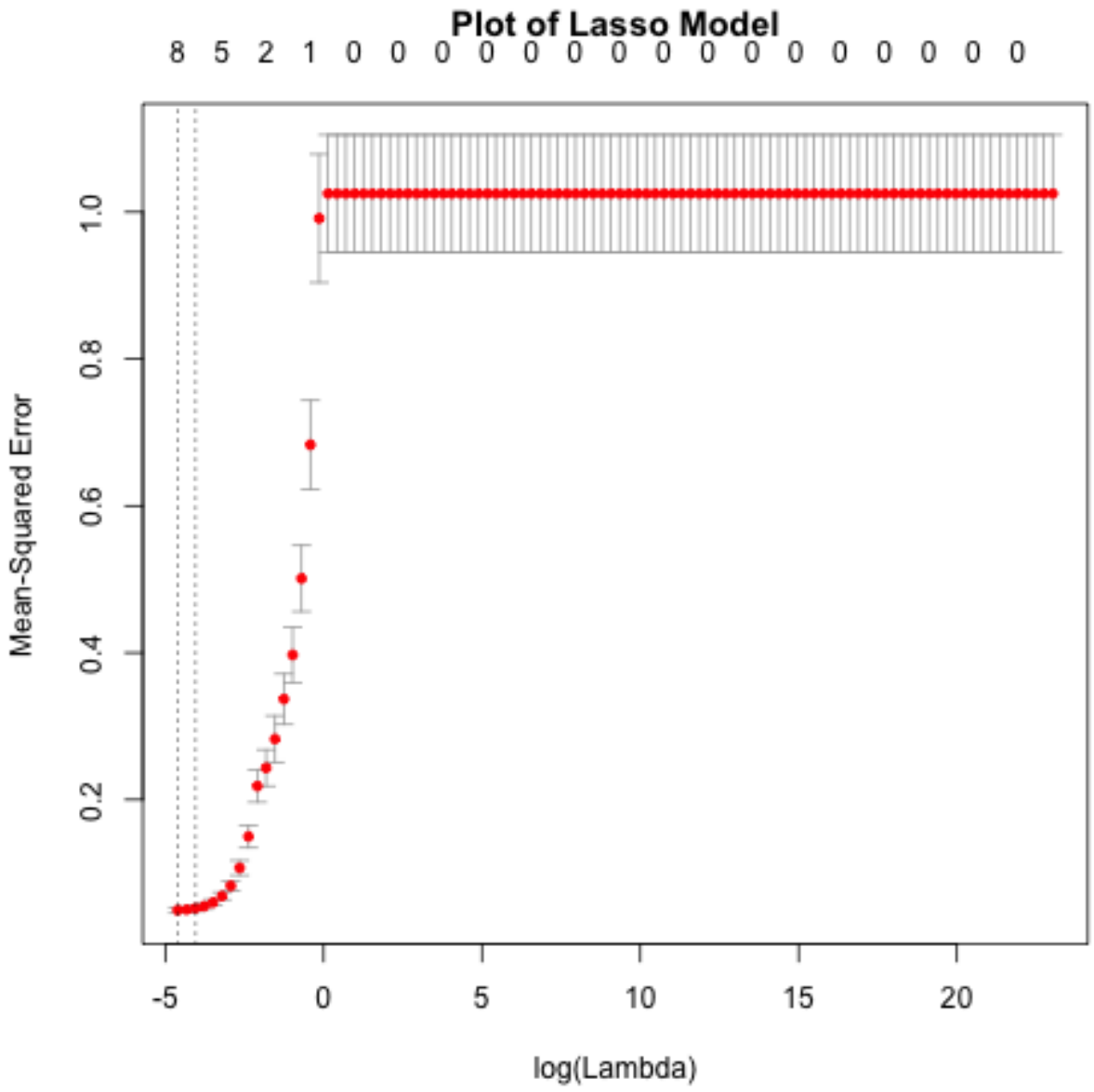


Figure 3:

Principal Components Regression

```
library(pls)
set.seed(1)

# Fit Model

pcr_fit=pcr(Balance ~ .,
            data = data.frame(credit_train),
            scale = FALSE,
            validation = "CV")
summary(pcr_fit)

# Find number of components with best fit

best_mod_comp_pcr <- which.min(pcr_fit$validation$PRESS) # 10
#save(best_mod_comp_pcr, file = "data/pcr_best_model_component.RData")

# Predict and MSE

pcr_pred = predict(pcr_fit,
                   credit_test[,-12],
                   ncomp = best_mod_comp_pcr)

pcr_mse <- mean((pcr_pred - credit_test[,12])^2)
#save(pcr_mse, file = "data/pcr_mse.RData")

# Refit on full dataset & get coefficients

pcr_fit_full <- pcr(Balance ~ .,
                    data = data.frame(scaled_credit),
                    scale = FALSE,
                    ncomp = best_mod_comp_pcr)

pcr_full_coefs <- coefficients(pcr_fit_full)
```

PLS Regression

```
library(pls)
set.seed(1)

# Fit Model

pls_fit=plsr(Balance ~ .,
            data = data.frame(credit_train),
            scale = FALSE,
            validation = "CV")
summary(pls_fit)

# Find number of components with best fit

best_mod_comp_pls <- which.min(pls_fit$validation$PRESS)
```

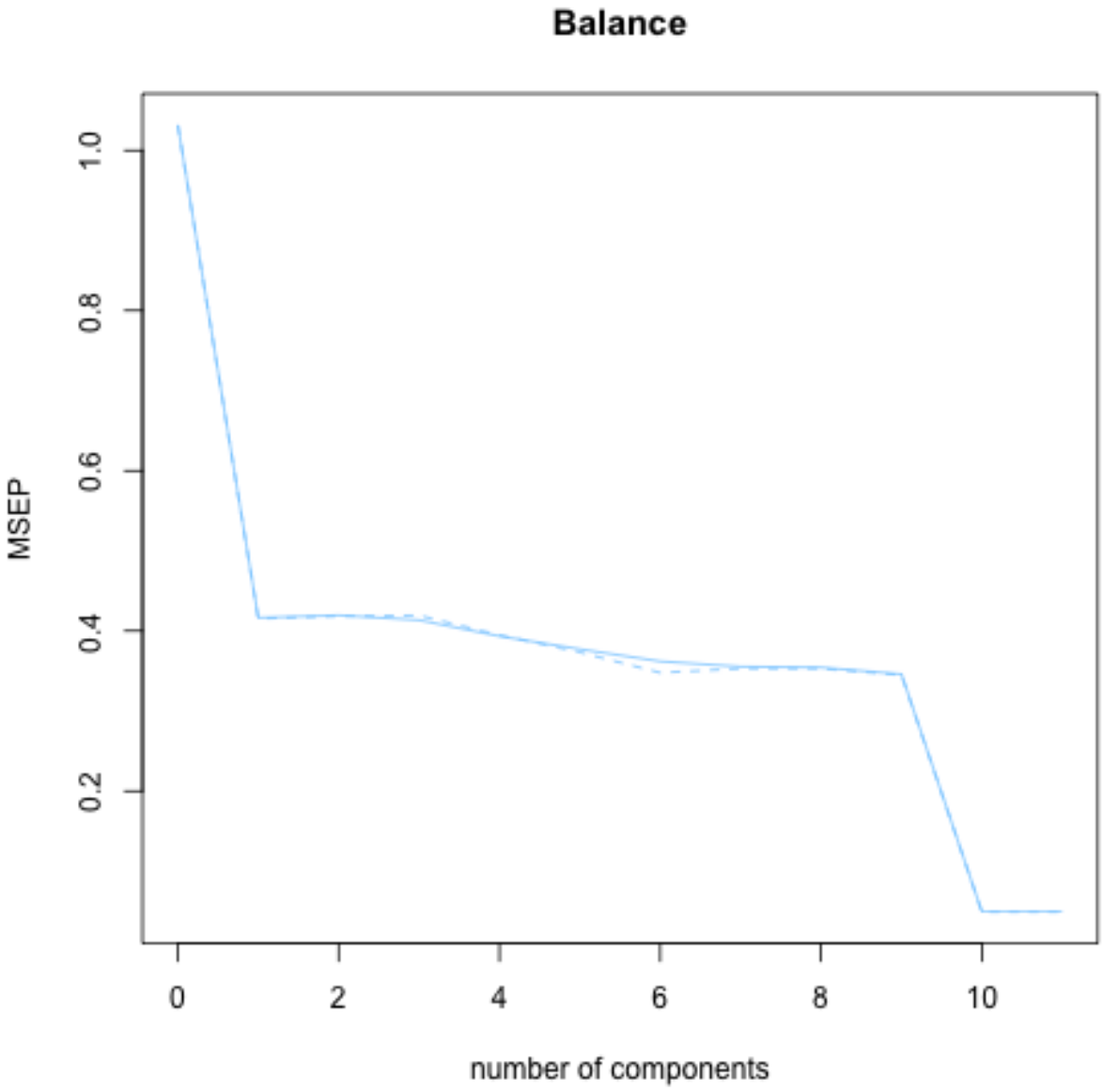


Figure 4:

```r
# Predict and MSE

pls_pred = predict(pls_fit,
                   credit_test[,-12],
                   ncomp = best_mod_comp_pls)


pls_mse <- mean((pls_pred - credit_test[,12])^2)

# Refit on full dataset

pls_fit_full <- plsr(Balance ~ .,
                     data = data.frame(scaled_credit),
                     scale = FALSE,
                     ncomp = 5)

pls_full_coefs <- coefficients(pls_fit_full)
```

Results

This table reports the coefficients we get after fitting the models:

```
##
##                     OLS       Ridge     LASSO      PCR       PLS
##  Income             -0.598171 -0.396532 -5.514e-01 -0.598867 -0.59880
##  Limit               0.958439  0.574679  7.816e-01  0.671407  0.67551
##  Rating              0.382479  0.558804  5.110e-01  0.670638  0.66648
##  Cards               0.052865  0.047147  3.884e-02  0.040431  0.04055
##  Age                -0.023033 -0.035899 -1.676e-02 -0.023274 -0.02295
##  Education          -0.007469 -0.003220  0.000e+00 -0.005999 -0.00582
##  GenderFemale       -0.011593 -0.005278 -9.817e-05 -0.011647 -0.01176
##  StudentYes          0.278155  0.249341  2.661e-01  0.276355  0.27652
##  MarriedYes         -0.009054 -0.012836  0.000e+00 -0.011161 -0.01122
##  EthnicityAsian      0.015951  0.012458  0.000e+00  0.017409  0.01872
##  EthnicityCaucasian  0.011005  0.009923  0.000e+00  0.011187  0.01257
```

This is a graphical representation of the coefficients:

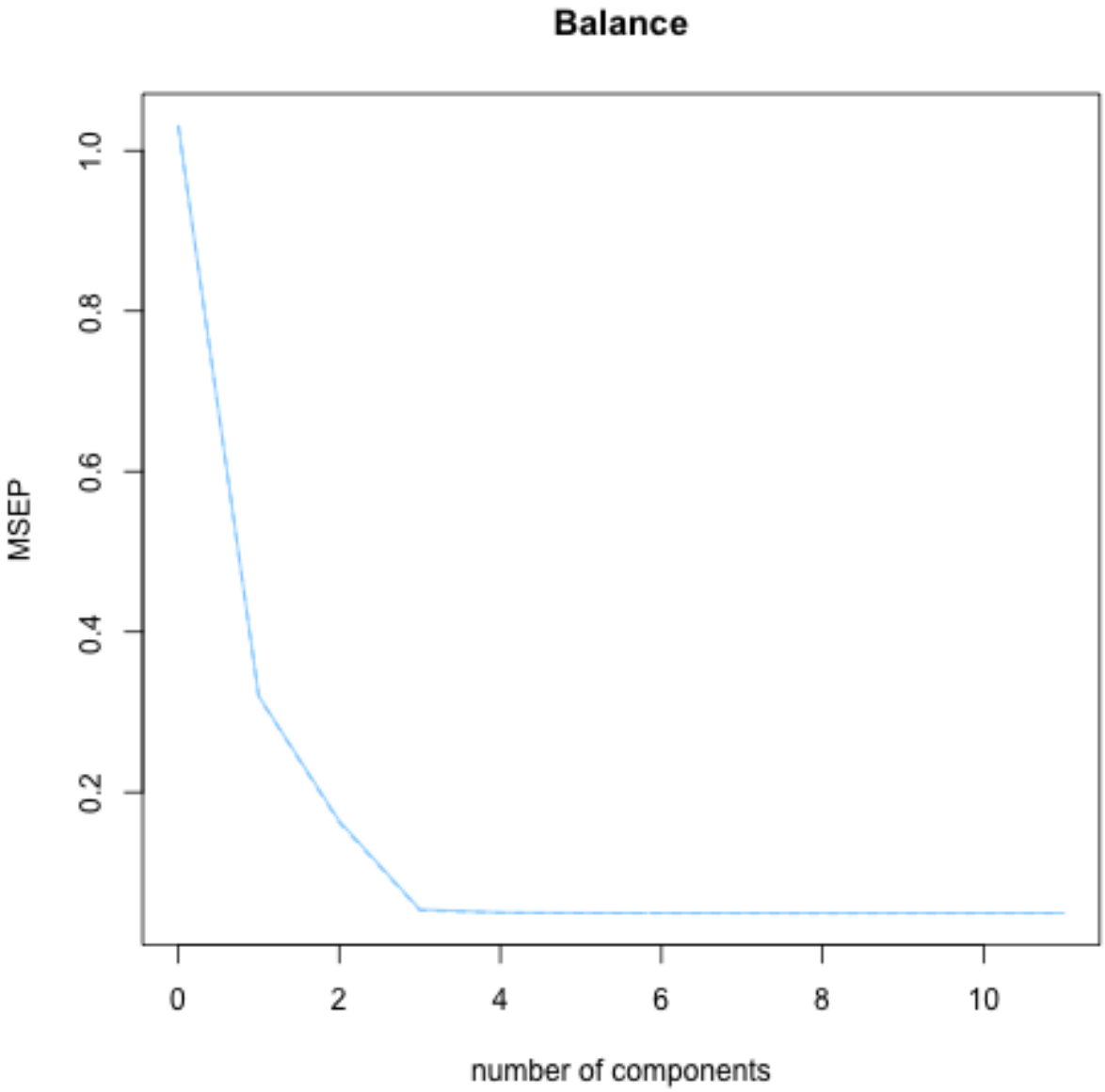
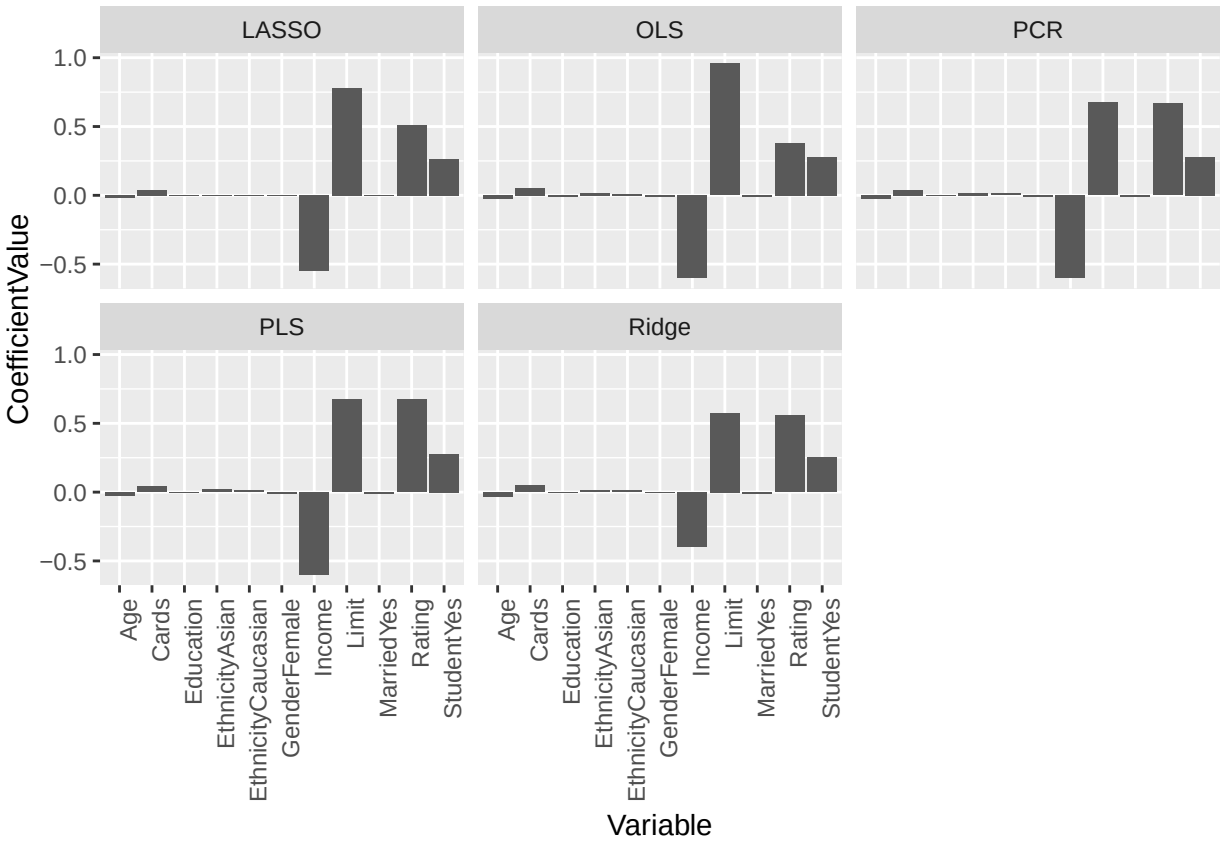


Figure 5:



Here are the MSE's of each regression:

```
##
##       Ridge   LASSO   PCR     PLS
##  MSE 0.04544 0.04676 0.04601 0.04658
```

For our analysis,

Ridge Regression, with 11 components, resulted in an MSE of 0.0454419

LASSO performed best with 7 components. The full model fit with this number of components resulted in an MSE of 0.0467553

Principal Components Regression performed best with 11 components. The full model fit with this number of components resulted in an MSE of 0.0460076

Partial Least Squares Regression performe best with 11 components. The model fit with this number of components resulted in an MSE of 0.0465753

Thus, our best performing model, with an MSE of 0.0454419, is Ridge.

Conclusion

Given a data set containing information pertaining to credit card balances and other factors such as income, race and gender, how could one develop a model to predict one's credit card balance?

The above analysis answered this question via a predictive modeling process that discussed several different regression methods. Specifically, the analysis utilized OLS, ridge, LASSO, PCA, and PLS regressions. Each method was discussed and applied separately to the Credit data, where credict card `Balance` was modeled as

a function of 10 variables: `Income`, `Limit`, `Rating`, `Cards`, `Age`, `Education`, `Gender`, `Student`, `Married`, and `Ethnicity`.

All of our models performed well. Thus, regression techniques work well when you have multiple predictors, a continuous response type, and assume a linear relationship between the response and predictors. Different regression techniques exist with different strengths and weaknesses.