Abstract

We perform a predictive modeling process applied on the data set Credit in order to decipher how factors such as Income, Age, Education, and more influence the amount of credit card debt to a person. Our analysis is based off of Chapter 6 from *An Introduction to Statistical Learning*, "Linear Model Selection and Regularization".

Introduction

In this project, we applied model selection methods on different regression models introduced in *Chapter 6: Linear Model Selection and Regularization (from **"An Introduction to Statistical Learning"** by James et al).* The 5 regression models we practiced are *Ordinary Least Squares, Ridge Regression, Lasso Regression, Principle Components Regression and Partial Least Square Regression.* In order to find the best parameter (or coefficients) for those regression models, we splitted our datasets into train and test groups, performed 10-fold cross validation on each model respectively, and in the end chose the best model with the least cross-validation error.

Data

The Credit data set we will be using comes from *An Introduction to Statistical Learning*, contains the following variables: Income, Limit, Rating, Cards, Age, Education, Gender, Student, Married, Ethnicity and Balance. The Credit data set records Balance, the average credit card debt for an individual, according to quantitative predictors such as Income, Limit, etc. and qualitative predictors such as Student, Married, etc. Cards represents the number of credit cards a person has and Income is represented in the thousands of dollars. We investigate 400 individuals in this analysis.

Methods

We applied the following 3 methods which has a total of 5 regression models to our dataset.

Ordinary Least Square methods

In statistics, OLS is a method for estimating the coefficients in a linear regression model, with the goal of minimizing the sum of the squares of the differences between the observed responses in the given dataset and those predicted by the linear model.

1. **Ordinary Least Squares (OLS)**: This is a common liear regression model, and we use it as a benchmark to evaluate the performance of the other 4 regression models.

Shrinkage methods

Shrinkage methods invovles fitting a model with all P predictor. However, the estimated coefficients are shrunken towards 0 relative to the least squares estimates. Shrinkage has the effect of reducing variance. Since after the process of shrinkage, some of the coefficients might be exactly 0, it also performs variable selection.

1. **Ridge Regression (Ridge)**:
2. **Lasso Regression (Lasso)**

Dimension Reduction methods

Dimension reduction works by *projecting* the P predictors onto a M-dimensional space, where M < P. This is achieved by computing M different linear combinations, or projections, of the variables. Then these M projections are used as predictors to fit a linear regression model by least squares.

1. **Principle Components Regression (PCR)**
2. **Partial Least Square Regression (PLSR)**

Analysis

Data Processing

In order to conduct our analysis, we first began with some pre-modeling data processing. The two main actions we performed before applying our models were:

1. Dummying out categorical variable
2. Mean centering and standardizing all the variables

For step 1, we took our categorical, or qualitative variables, namely `Student`, `Married`, `Gender`, and `Ethnicity` and *dummied* them out. What this means is that the values for the variables were factored and then given binary indicators. The reason for this is you can not apply regression models, specifically the lasso and ridge regressions that we used from the `glmnet` package, to variables that are non numeric. We used the `model.matrix()` function to perform step 1.

For step 2, we standardized our data so that each variable would have a mean of 0 and standard deviation of 1. The purpose of this is to standardize across different scales, so that coefficients such as $\hat{\beta}_0$ do not vary based on whether they're calculated in pounds or ounces, etc. To do so, we used the `scale()` function.

Training and Testing Data Sets

Another step we took before building our model was to take a set of data for model building and another for testing model performance. The set of data used to build the model, our *training set*, was 300 events randomly selected using the `sample()` function. For our test set, we randomly selected 100 events, this time using the `set.seed()` function for reproducibility.

Model Building Process

Results

Conclusions