INTERMEDIATE ANALYTICS

ALY6015, FALL 2019

MODULE 3 ASSIGNMENT

REGULARIZATION

SUBMITTED BY: SHIVANI ADSAR

NUID: 001399374

CRN: 71933

SUBMITTED TO: PROF. LI, TENGLONG

DATE: 19/11/2019

Introduction

The assignment aims at using the practical knowledge into technique and skills with their applicational value. We have implemented LASSO regression function using glmnet () function in oreder to build linear and logistic models over the regularization parameter.

Analysis

Using the built-in "Boston" dataset for predicting the housing prices which consists of per capita crime rate, proportion of residential land, proportion of non-retail business, age, full value property tax, pupil-teacher ratio by town and other parameters.

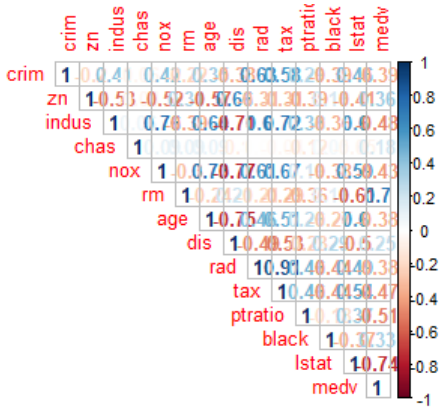
Loading the data

We have loaded and installed al the required packages like MASS, GGally, glmnet and dplyr for performing the LASSO regularization.

Data Exploration

- This involves understanding the dataset using the dim(), cormat() and corrplot() functions. The corrplot package provides a graphical representation of correlation matrix and confidence interval.
- Using the below functions for plotting:

```
> cormat=cor(Boston)
> corrplot(cormat, type = "upper", method="number")
```



It can be observed that corrplot package contains some methods to perform the matrix re-ordering. It shows some positive correlations in blue color and negative correlations in red color. The intensity of color and the circle size is proportional to the correlation coefficients.

Fig.1: Correlation matrix

Splitting the Test and Train Data

In order to avoid biasing in the Test using the Train dataset, the train-test split should be performed. Hence, we would be splitting the dataset randomly into 80% and 20% of the values. The set.seed() function is used as a random number generator which allows us to generate numbers for performing statistics on the given data.

```
> set.seed(123)
> index <- sample(nrow(Boston),nrow(Boston)*0.80) #80-20 split
> Boston.train <- Boston[index,]
> Boston.test <- Boston[-index,]
```

Fig.: 2 Code for Train and Test data

Building the Linear Regression Model

- We have used the linear regression which assumes that there exists a linear relationship between the response variable and the explanatory variable.
- We have performed models for the Boston dataset using the indus and age variables using the train dataset. As can be seen in the below code, the "medv" is our target variable and the remaining variables are the feature variables which will help in predicting the housing prices.

```
model0<- lm(medv~lstat, data = Boston.train)
model1<- lm(medv~., data=Boston.train)
model2<- lm(medv~. -indus -age, data=Boston.train)
```

Fig.3 : Linear Regression Code

```
Residual standard error: 4.723 on 392 degrees of freedom
Multiple R-squared:  0.7436,    Adjusted R-squared:  0.7364
F-statistic: 103.4 on 11 and 392 DF,  p-value: < 2.2e-16
```

Fig.4: Output of the Linear Regression

We have used the lm() function to perform linear regression. Also, anova() has been used for the two models. It can be observed that the the RSE is 4.723, Adjusted R-square is 0.7436 and the F-statistic is 103.4. We can note that, as the p-value for anova is around 0.6, it is seen that indus and age do not contribute in predicting the housing price.

Lasso Regression

Lasso regression performs L1 Regularization. It adds a penalty equivalent to the regression coefficients and minimizes them.

- The default value of regularization parameter in lasso regression is 1.

```
lasso.fit<- glmnet(x=X.train, y=Y.train, family = "gaussian", alpha = 1)
plot(lasso.fit, xvar = "lambda", label=TRUE)
```

Fig.5: Lasso Regression Code

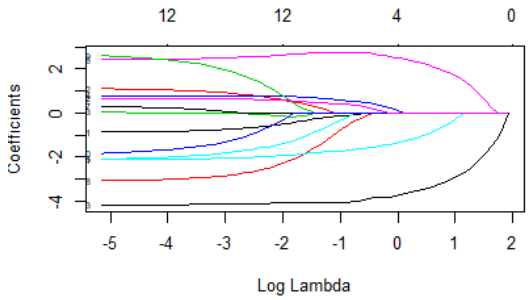


Fig.6: L1 vs. Coefficients Plot

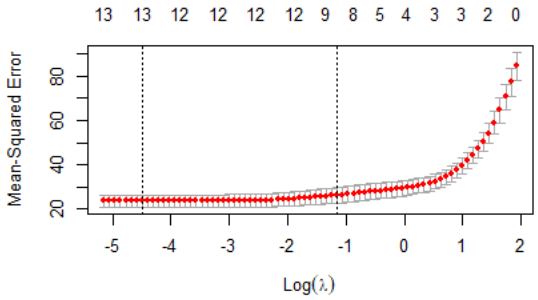
It can be observed that, when lambda =0, the lasso gives the least squares while when the lambda becomes large, the lasso gives the output as a null model having all coefficients as 0. Moving from left to right, it can be observed that the lasso results in a model that contains the predictor.

- We are using the five important variables as r, chas, nox, dis and ptratio.

- Further, we select some potential values, and use the cross validation to estimate the error rate on test data and will prefer the values with minimum error.
- Cross Validation in LASSO

```
cv.lasso<- cv.glmnet(x=X.train, y=Y.train, family = "gaussian", alpha = 1, nfolds = 10)
plot(cv.lasso)
```

Fig.7: Cross Validation Code in LASSO



It can be observed from the plot, that the error is high for lambda and the coefficients are restricted to be minimal. This indicates that the model is working fine. Amongst the two vertical lines, one is at minimum and the other line is at standard error of minimum.

Fig.8: MSE vs Lambda Plot For LASSO

As can be observed, LASSO Regression performs variable shrinkage as the coefficient of age turns 0.

Big lasso

The big lasso is used for large datasets that cannot be loaded in to the memory. We are using the “Colon” dataset which contains the gene data of 62 samples from colon-cancer patients. We have used the biglasso() function which involves inclusion of an intercept during the model fitting process and helps in making the data more normalized. The cvfit() function helps in calculating the cross validation.

```
cvfit <- cv.biglasso(X.bm, y, family = 'binomial', seed = 1234, ncores = 2)
fit <- biglasso(X.bm, y, screen = "SSR-BEDPP")
```

Fig.9: Cross Validation Function Code

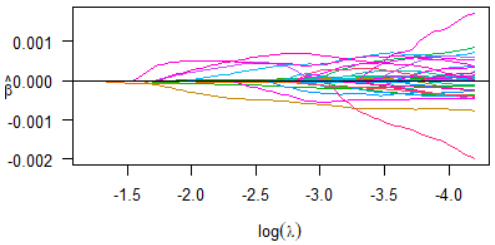


Fig.10: Lasso Regression

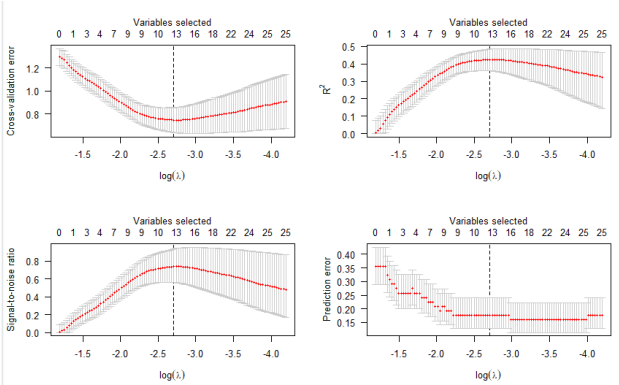


Fig.11: Cross Validation Plots

The entire dataset is converted into a matrix using the big.matrix() function as shown in the code snippet below:

```
X.bm <- as.big.matrix(X)
```

We have used the cross validation plots to understand and analyse the Cross-validation error, Signal to noise ratio, R^2 and the prediction error.

References

- Rpubs.com. (2019). *RPubs - Linear Model Selection and Regularization Using Boston Housing Data*. [online] Available at: https://rpubs.com/Swidle/368819 [Accessed 19 Nov. 2019].
- https://cran.r-project.org/web/packages/biglasso/vignettes/bigla