# Lab 07: Multiple Linear Regression: Variable Selection

**OBJECTIVES**

1.Use PROC REG to fit multiple regression models.

2.Learn how to find the best reduced model.

3.Variable diagnostics and influential statistics

In multiple regression, several variables can be involved and regressed on one another (model: $Y = \beta_0 + \beta_1X_1+\beta_2X_2+ \cdots + \beta_pX_P + \varepsilon$). The overall test of hypothesis of multiple linear regression is:

H0: $\beta1=\beta2 = \cdots = \beta p =0$ v.s.

H1: at least one $\beta \neq 0$.

Rejection of $H_0$ implies that at least one of the regressors, $X_1, X_2, \ldots, X_p$, contributes significantly to the model. In lab 4s and 5, we have used several statistics such as F-test, t-test of regression coefficient, standardized regression coefficients and partial R2 to measure the relative importance of independent variables, which tell us which independent variables are more important than the others in predicating the values of the dependent variable.

Then the question is how to choose the 'best' model of multiple regression for the current data, i.e. which variables should remain in the model, to guide its application and future studies. Theoretically, the ideal model provides the best possible fit while using the fewest possible parameters, that is, a good model is easier both to fit and to interpret. In this lab, we will introduce common variable selection methods based on F-statistics or t-test of parameter estimates (the best criteria to measure the relative importance of independent variables) including Forward selection, Backward elimination, Stepwise selection that are widely used for multiple liner regression by different statistical analysis software like SAS.

**Forward selection** fits all possible simple linear models and chooses the best one (largest F-statistics for type II SS or t-value for test of parameter estimate). Then all possible 2-varible models that include the first chosen variable are compared, and so on. The process continues until no remaining variable generates a significant F-statistics or t-test of parameter estimate. With this process, once a variable enters the model it remains in the model. The significant level of $\alpha$, is called "alpha to enter".

**Backward elimination** starts with the full model including all the independent variables and removes one variable at a time based on a user-defined selection criterion. The default in SAS is to remove the variable with the least significant F-test for type II SS or t-test for parameter estimate. Then the model is refitted, and the process is repeated. When all statistical tests are significant (i.e. none of the parameter estimates are zero), the reduced model has been chosen. With this method, once a variable is dropped from the model it does not reenter. The preset significant level is called the "alpha to drop".

**Stepwise selection** works in much the same way as forward selection, with the exception that the significance of each variable is rechecked at each step along the process and removed if it falls below the significant threshold. Virtually this method combines forward selection and backward elimination. In this

method, a variable may enter and leave the model several times during the procedure. The procedure depends on two preset significant levels, “alpha to enter” and “alpha to drop”.

**LABORATORY INSTRUCTIONS**

**Part I.**

**Housekeeping Statements**

Before we dive into the main part of the code it is good to create a pre-amble in which we will load all the necessary packages for R to execute the following tasks. If you have them installed already great. If not, you can install it using the “packages” tab on the bottom right panel. Click install, and put the name of the package you want on the “install packages” window that pops up. The default setting is installing the packages from the CRAN repository where most “mainstream” packages can be found.

To activate the packages, use the following commands:

```
library(olsrr)
library(caret)
library(leaps)
library(MASS)
library(tidyverse)
```

**Dataset**
The data is from your textbook, chapter 7, problem 6 and you can attain it through the link:
http://www.stat.lsu.edu/exstweb/statlab/datasets/fwdata97/FW07P06.txt.

**Data set**

The data set is from Chapter 8, Problem 13 in “Statistical Methods” by Freund, Wilson and Mohr @ 2010 Elsevier Inc. This data set came from a study from an apartment owner to investigate what improvements or changes in her complex may bring in more rental income. From a sample of 34 complexes she obtains the monthly rent on single-bed room units and the following characteristics:

AGE: the age of the property,

SQFT: square footage of unit,

SD: amount of security deposit,

UNTS: number of units in complex,

GAR: present of a garage (0-no, 1-yes),

CP: presence of a carpet (0-no, 1-yes),

SS: Security system (0-no, 1-yes),

FIT: fitness facilities (0-no, 1-yes),

RENT: monthly rental.

We will perform a multiple linear regression using RENT as dependent variable and the others as independent variables. The data is available at:

Make sure you download and save it in the same directory as your R script, using the name **data_lab7.txt**

First, we need some codes to clean up the dataset and make sure that R views each column appropriately. The following commands will do the trick:

```
RENTS=read.table("data_lab7.txt", header = TRUE, sep = "", dec = ".") #Reads the dataset and save it as the dataframe asphalt

RENTS=RENTS[,-1] # Removes the column with the observation number.

View(RENTS) # Views the dataset

RENTS[,c(1:4,9)] = lapply(RENTS[,c(1:4,9)], as.numeric) # Makes sure that all numeric variables are seen as such by R

RENTS[,5:8] = lapply(RENTS[,5:8], as.factor) # Makes sure that all factor variables are seen as such by R
```

This way we created a new dataframe in R called RENTS, by reading it in and removing the dummy column of row numbers. We also made sure that all the variables that are numeric are viewed as numbers. Also, the variables which are factors are viewed as such by mass-defining them to be the appropriate categories using the lapply command.

**Part II**

**Multiple Linear Regression by using lm**

The following code will create and output a full model:

```
full.model <- lm(rent ~., data = RENTS)

summary(full.model)
```

In order to create one with backward selection, forward selection or stepwise selection you use the following three commands respectively:

```
stepback.model <- stepAIC(full.model, direction = "backward",
                          trace = FALSE)
summary(stepback.model)
```

```
stepfor.model <- stepAIC(full.model, direction = "forward",
                         trace = FALSE)
summary(stepfor.model)
```

```
stepboth.model <- stepAIC(full.model, direction = "both",
                  trace = FALSE)
summary(stepboth.model)
```

The models are saved as **stepback.model, stepfor.model** and **stepboth.model** respectively. There is a plethora of outputs saved om those R objects but most of the things we use can be found through the **summary** command that follows each computation.

You can also do an anova analysis on the three models using the following codes:

For the full model:

```
an1=anova(full.model) # Traditional Analysis of variance
SS1=full.model$`Sum Sq`# List of Sums of Squares Type I
View(SS1)
View(an1)
```

For the backward selection:

```
an2=anova(stepback.model) # Traditional Analysis of variance
SS2=stepback.model$`Sum Sq`# List of Sums of Squares Type I
View(SS2)
View(an2)
```

For the forward selection:

```
an3=anova(stepfor.model) # Traditional Analysis of variance
SS3=stepback.model$`Sum Sq`# List of Sums of Squares Type I
View(SS3)
View(an3)
```

And for the stepwise one:

```
an4=anova(stepboth.model) # Traditional Analysis of variance
SS4=stepback.model$`Sum Sq`# List of Sums of Squares Type I
View(SS4)
View(an4)
```

Assuming now you have figured out the explanatory variables you can use them in the following code to create a reduced model:

```
reduced.model <- lm(rent ~ ,data = RENTS ) # This is the reduced multilinear regression model. Add the exploratory variables of interest here
```

Simply add the variable names that "survived" the regression selection after the **~** using **+** between them.

Then you can run the same diagnostics tests that we learned in pervious classes for the reduced model using the codes below:

```
ols_coll_diag(reduced.model) # Collinearity diagnostics table

ols_vif_tol(reduced.model) # VIF computation

ols_plot_obs_fit(reduced.model) # Observed vs Predicted Plot

ols_plot_diagnostics(reduced.model) # A panel of plots for regression diagnostics
```

Finally, if you want to run a normality test on the residuals of the model you can use the following code:

```
ols_plot_resid_qq(reduced.model) "Creates a qq plor for the residuals"

ols_test_normality(reduced.model) "Tests the residuals for nomarlity using various goodness of fit tests"
```

**LAB ASSIGNMENT**

Your assignment is to perform necessary analysis using R and answer the following questions (Please do not print all the output. Only print the graphs and tables that you think are relevant to your answers).

1. Run all three of the selection methods discussed above. Report the result of each method.

2. Do you get the same reduced model from three methods? Make brief comments.

3. Which model do you think is the "best" reduced model? Discuss why you choose this model.

4. Use lm to fit the best reduced model. Report the usual results (Hints: hypothesis test results, parameter estimates, validity of assumptions, multicollinearity, outliers, and influential statistics).

*Remember to attach your R code with your lab report.