Tutorial R Part I- Experimentation with Linear Regression, Regularization and Model Comparison Intermediate Level

Hoofar Pourzand, Email: rcc@rcc.psu.edu
Research Computing and Cyberinfrastructure (RCC)
Computer Building

July 1, 2013

Abstract

This Tutorial in R is put in the form of an experimentation on a dataset to review a fuller capacity of R in performing statistical analysis. By the end of this Tutorial you'll be exposed to data preparation in R and working with libraries, how to access the outputs in R, reading T statistics in R, how to perform a linear regression , Lasso and Ridge regression and finally plotting results in R.

1 Introduction to the Data set

All following descriptions relate to the R script detailed in the appendix. For n=442 diabetes patients, 10 variables as the potential predictors were recorded as well as the response of interest (y) in the provided data set. Loading the data in R, partitioning the patients 75/25 for training/testing the first step is to do a linear regression analysis using ridge and lasso functions in R.

Listing 1: Loading Packages in R

```
#load the diabetes data
library(lars)
data(diabetes)
attach(diabetes)
```

Listing 2: Data Preparation in R

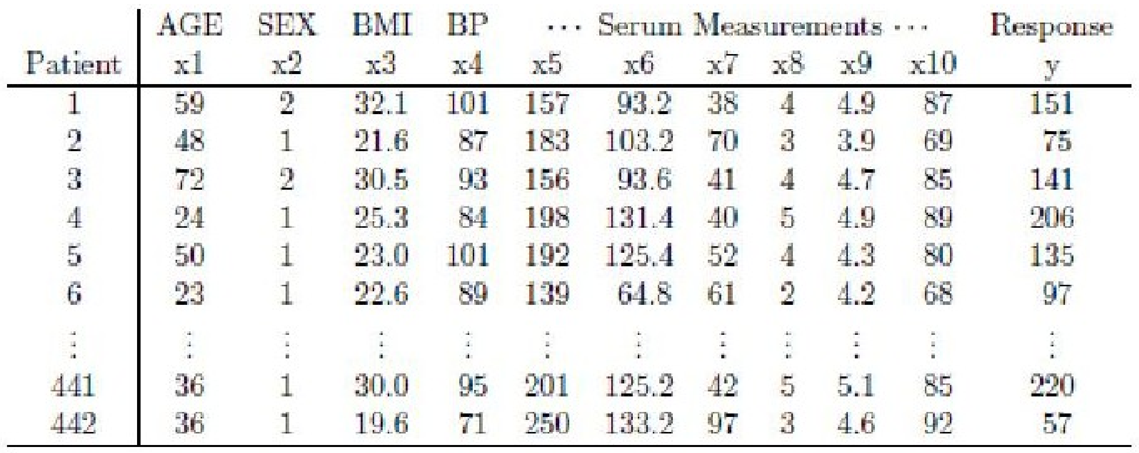
```
#Partition the data: training set and testing set

n= dim(diabetes$x)[1]
p= dim(diabetes$x)[2]
set.seed(2011)
test = sample(n, round(n/4))

#Training data
```

```
y = diabetes$y[-test]; X<- diabetes$x[-test,]

#Test data
y.test <- diabetes$y[test]; X.test <- diabetes$x[test,]
```



	AGE	SEX	BMI	BP	··· Serum		Measurements		···		Response
Patient	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
1	59	2	32.1	101	157	93.2	38	4	4.9	87	151
2	48	1	21.6	87	183	103.2	70	3	3.9	69	75
3	72	2	30.5	93	156	93.6	41	4	4.7	85	141
4	24	1	25.3	84	198	131.4	40	5	4.9	89	206
5	50	1	23.0	101	192	125.4	52	4	4.3	80	135
6	23	1	22.6	89	139	64.8	61	2	4.2	68	97
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
441	36	1	30.0	95	201	125.2	42	5	5.1	85	220
442	36	1	19.6	71	250	133.2	97	3	4.6	92	57

Figure 1: Diabetes study: 442 diabetes patients were measured on 10 baseline variables. A prediction model was desired for the response variable, a measure of disease progression one year after baseline.

2 Step One- Data preparation

Let's apply a linear regression model to the data and after which we'll perform a stepwise variable selection to the diabetes data;
AIC will be used as the model selection criterion.
Both forward selection and backward elimination.
If they result in different models, we'll choose the one that gives the minimum AIC.
Writing the linear regression model as [1][6]:

$$E(Y \mid X) = \mathbf{X}\beta = X_1\beta_1 + X_2\beta_2 + .. + X_p\beta_p \tag{1}$$

Let's also perform a transformation of X_j to the standard distribution with y as the response :

$$\begin{aligned} X_j &= \frac{X_{new,j} - \mu(X_{train,j})}{\sigma(X_{train,j})} \\ y &= y_{new} - \mu(y_{train}) \end{aligned} \tag{2}$$

Then for the lasso and the ridge regression we'll be working with the X_{new} and y_{new}:

$$\hat{y}_{new} = \mu(y_{train}) + \sum_{j=1}^{p} \hat{\beta}_j \frac{X_{new,j} - \mu(X_{train,j})}{\sigma(X_{train,j})} \tag{3}$$

Listing 3: Standardizing the Data in R

```
for (j in 1:p)
  X.stand[,j] = (X[,j] - mean(X[,j])) / sd(X[,j])

standardizing the testing data

X.test.stand <- X.test
for (j in 1:p)
  X.test.stand[,j] = (X.test.stand[,j] - mean(X.test[,j])) / sd(X.test[,j])
```

This Experimentation with R is formatted such that the random variable is written in uppercase and the observed values in lowercase for easier reading. In linear regression, the expectation of a random variable is basically its "average" value over its distribution, so we'll also do a scatter plot of the response to get a better feeling for the data under study:

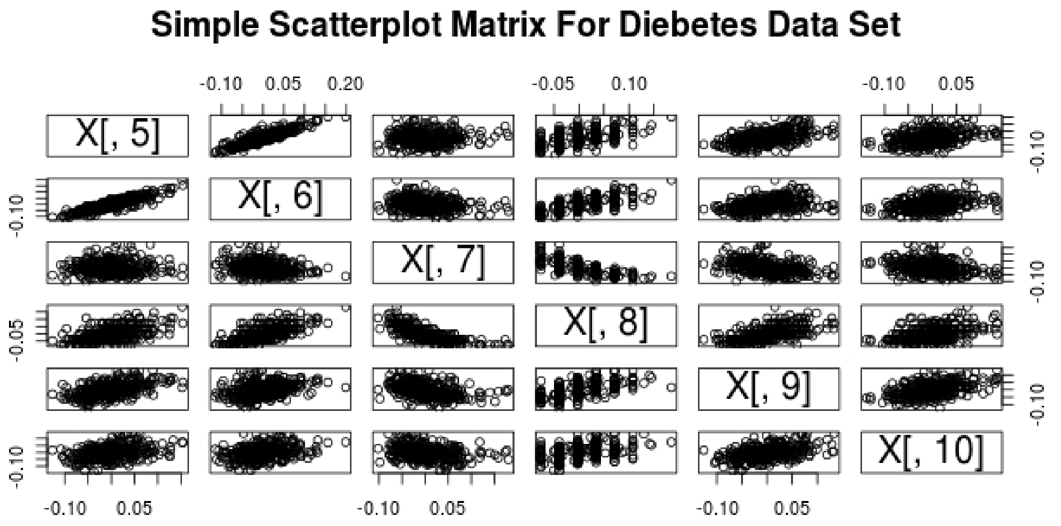


Figure 2: Scatter plots for the last five predictors

As we see from the scatter plot, there is some correlation between predictors; we could also check for any interactions separately. We have assumed that the data $x_1, x_2, .., x_p$ is composed of independent and identically distributed elements, however judging by the scatter plots they may not be completely so. A more advanced statistical analysis might employ a step wise selection method in order to pick the most interesting predictors, containing useful information on the response. A complete model and a T test should be performed for all the parameters in the interests of completeness and in order to appreciate the power of stepwise parameter selection techniques[4]. Another important feature in statistical analysis is the normality; the normality of the backward fitted model predictors are plotted using a Q-Q plot, below:

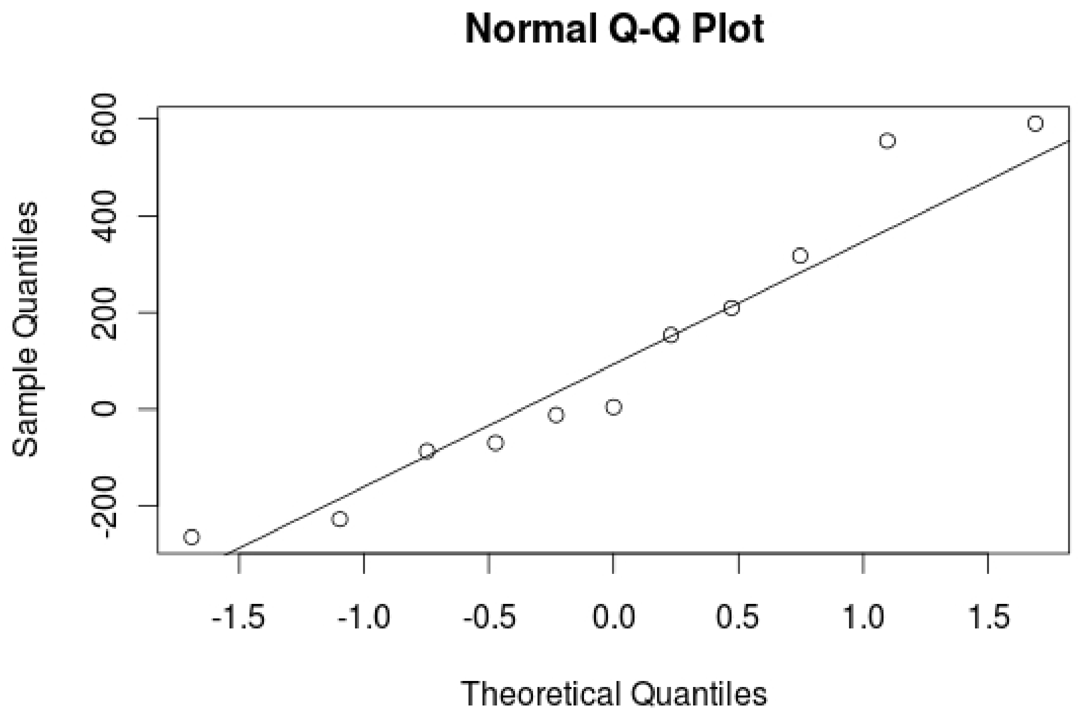


Figure 3: Normality Distribution: Significant departures from the line suggest violations of normality

3 Linear regression

Models are listed below; we wish to explain the data in the simplest way possible, ie., redundant predictors should be removed, ala Occam's Razor. The results of this process are recorded in the appendix.

Listing 4: Linear Regression in R

```
# Apply  linear regression models
fit.lm <- lm(y~X)
names(fit.lm)
```

```
fit.lm$coefficients #regression coefficients
fit.lm$residuals   #residuals
intercept <- lm(y~1)
```

```
 [1] "coefficients"  "residuals"     "effects"       "rank"          "fitted.values"
 [6] "assign"        "qr"            "df.residual"   "xlevels"       "call"
[11] "terms"         "model"
```

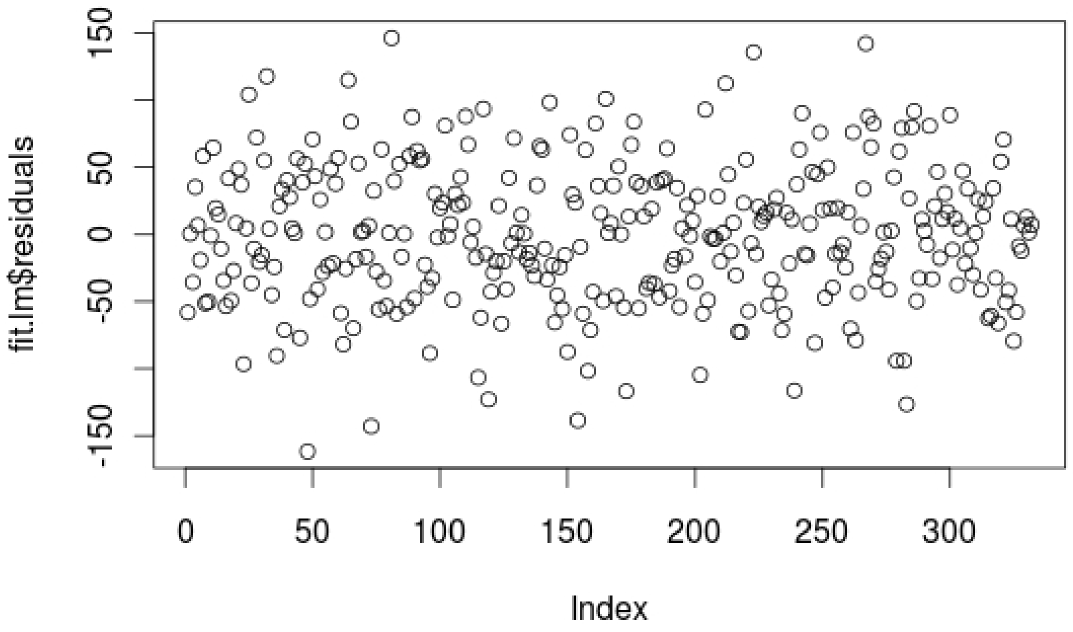


Figure 4: Cp Curve

Listing 5: Step-wise Variable Selection in R

```
#Apply step-wise variable selection
#Run forward selection
#Run backward elimination

bwfit.lm <- step(fit.lm, direction= "backward", trace = 0)
fwfit.lm <- step(intercept, direction= "forward",scope=list(upper=fit.lm,lower=~1),
    trace = 0)
intercM.lm <- step(intercept, direction= "backward", trace = 0)
```

Listing 6: Applying AIC for model selection in R

```
#using AIC for model selction criterion
aic.bwfit.lm <- AIC(bwfit.lm, k =2)
aic.fwfit.lm <- AIC(fwfit.lm, k =2)

#choose the one tht gives the min AIC
bwfit.lm$coefficients

#Mean Square Errors
mse.lmmodel = mean((bwfit.lm$fitted.values-y)^2)

y.perdicted = X.test[,c(2,3,4,5,8,9)] %*%
bwfit.lm$coefficients[-1] + bwfit.lm$coefficients[1]
mse.lmmodel.test = mean((y.perdicted-y.test)^2)
```

4 Ridge regression

Adding variables to the model will not always reduce the sum of squared residuals measured on the validation set, which we could easily see when comparing the MSE errors of the three linear models (lm.fit() , lasso.fit() and fit.rid models).

Listing 7: Ridge Regression in R

```
ridge.fit <- lm.ridge(y.center~X.stand,  lambda = exp(seq(-15, 15, 0.1)))
ridge.fit$GCV
select(ridge.fit)

ridge.fit <- lm.ridge(y.center~X.stand,  lambda = 24.53253) #ridge.fit$GCV)

ridge.fit.coef <- ridge.fit$coef

#lasso regression

lasso.fit <- lars(X.stand, y.center, type = c("lasso"))
summary(lasso.fit)
plot(lasso.fit, xvar = c("norm", "df", "arc.length", "step"), breaks = TRUE,
      plottype = c("Cp"), omit.zeros = TRUE)
lasso.fit.coef <- lasso.fit$beta[8,]
```

Applying ridge regression, the snippet above, to the diabetes data, we get:- λ is selected by the GCV method.

```
> ridge.fit$GCV
24.53253
8.619572
```

5 Lasso regression

Here we apply lasso regression to the diabetes data; the R function lars() can be used. λ can be selected by either the minimum C_p criterion in lars() or 5-fold cross-validation; below is a plot of a lars fit. The C_p curve, the C_p values for each step in forward step-wise models are more likely to provide an accurate measure in order to assess the fit of a regression model, versus plotting each coefficient as a function of the Degrees of Freedom (Df)[4,5]. The default is a complete coefficient path in R, shown in the Figure below[5].

Listing 8: Calculating MSE in R

```
y.train.lasso = X.stand%*% lasso.fit.coef + mean(y)
mse.lasso.fit = mean((y.train.lasso-y)^2)

#testing data
y.lasso.fit <-  X.test.stand %*% lasso.fit.coef + mean(y)
mse.lasso.fit = mean((y.lasso.fit - y.test)^2)
```

Listing 9: Working with Testing Data in R

```
#Part Two
  predict.lasso <- predict(lasso.fit,X.test)
  predict.rid   <- predict(ridge.fit,X.test)
  predict.lm    <- predict(fit.lm,X.test)

y.train.ridge.fit = X.stand %*% ridge.fit.coef + mean(y)
mse.ridge.fit = mean((y.train.ridge.fit - y)^2)

##########testing data for lasso regression#################
y.ridge.fit <- X.test.stand %*% ridge.fit.coef + mean(y)
mse.ridge <- mean((y.ridge.fit- y.test)^2)

mse_lasso <- sum((predict.lasso[[4]][,13] - y.test) ^ 2) / length(y.test
mse_ridge.fit <- sum((predict.rid - y.test) ^ 2) / length(y.test)
mse_ridge.fit <- sum((predict.lm - y.test) ^ 2) / length(y.test)
```

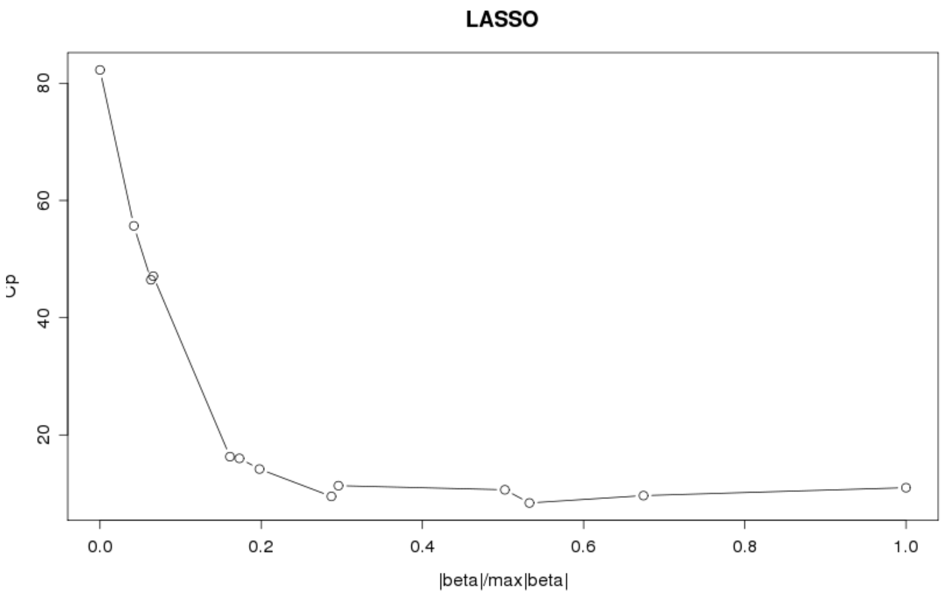


Figure 5: The Cp values plotted over Df.

```
> summary(lasso.fit)
LARS/LASSO
Call: lars(x = X.stand, y = y.center, type = c("lasso"))
   Df     Rss       Cp
0   1 2002285 382.7047
1   2 1763455 299.6944
2   3 1163515  88.1484
3   4 1125283  76.5396
4   5  982253  27.6289
5   6  952084  18.8903
6   7  925158  11.3059
7   8  910578   8.1162
8   9  906403   8.6302
9  10  902214   9.1392
10 11  901823  11.0000
```

Test data is used to compare the selected linear model, ridge regression and lasso regression through mean squared errors.

```
> mse.lmmodel
[1] 2716.334  %on the training data
> mse.lmmodel.test  %on the testing data:
   the whole 10 elements selected
```

```
[1] 3449.47

> mse.ridge.fit %on the training data
[1] 2727.146
> mse.ridge     %on the testing data
[1] 3558.877

> mse.lasso.fit   %on the training data
[1] 2742.704
> mse.lasso.fit  %on the testing data
[1] 3555.093
```

6 Discussion

Just as important as the correctness of a model for a particular data-set are the data analysis steps applied; paying attention to these can lead to lower fitting-errors. Understanding and treating the outliers in data is another way to improve the veracity of a model[3], while applying different measures, like Pearson's R, can prove valuable in giving insight into data and model suitability[7] as well.

7 Conclusion

In this example, a dataset of 442 cases of diabetes is partitioned into a training set of 332 cases and testing set of 110 cases. A linear regression model, after stepwise variable selection, has an AIC of 3591, the ridge regression and the lasso regression model, after centering the response and standardizing the predictors, gives, respectively, MSE of 2727.1 and 2742.7 on the training data and respectively 3558.8 and 3555.1 on the testing data. Also the MSE for the selected linear model was 2716.3 on the training data set and 2449.7 on the testing data set. This comparison would suggest that the lasso-regression model is the recommended model for this particular data set. The aforementioned predictive models are developed in R and here the details relating to their development and analysis is documented in this tutorial.

In the first part, a linear regression model was developed using the training data set. While performing the parameter selection, we notice that both the forward and backward elimination methods are fast methods for subset selection in linear regression, however, only step-wise selection is guaranteed to find the best subset. In this case the results of the three models were identical (producing the same AIC).

References

[1] Julian Faraway (2006) “Linear Models in R”. CRC Press.

[2] Schapire, R.E.(1990) “The Strength of Weak Learnability”. Machine Learning 5 (2): 197-227.

[3] Rousseeuw, P., Leroy (1987) “A. Robust Regression and Outlier Detection”. New York: John Wiley & Sons.

[4] Trisha Greenhalgh (1997) “How to read a paper: Statistics for the non-statistician. II: Significant relations and their pitfalls”, BMJ.

[5] Efron, Hastie, Johnstone and Tibshirani (2003) “Least Angle Regression” (with discussion) Annals of Statistics.

[6] Hastie, Tibshirani and Friedman (2002)“Elements of Statistical Learning”, Springer, NY.

[7] Joseph Lee Rodgers; W. Alan Nicewander (Feb., 1988) “Thirteen Ways to Look at the Correlation Coefficient”. The American Statistician, Vol. 42, No. 1. , pp. 59-66.

[8] Paul Teetor. “R Cookbook”. O’Reilly, first edition, 2011. ISBN 978-0-596-80915-7

A T statistics for parameter selection

```
Predictor       Coef  SE Coef      T      P
Constant     152.133    2.532  60.09  0.000
x.age          50.72    65.51   0.77  0.439
x.sex        -267.34    65.27  -4.10  0.000
x.bmi         460.72    84.60   5.45  0.000
x.map         342.93    72.45   4.73  0.000
x.tc           -3600    60575  -0.06  0.953
x.ldl           3028    53239   0.06  0.955
x.hdl           1103    22636   0.05  0.961
x.tch           74.9    275.8   0.27  0.786
x.ltg           1828    19915   0.09  0.927
x.glu          62.75    70.40   0.89  0.373
x2.age^2       67.69    69.47   0.97  0.330
x2.bmi^2       45.85    83.29   0.55  0.582
x2.map^2       -8.46    71.65  -0.12  0.906
x2.tc^2         6668     7059   0.94  0.345
x2.ldl^2        3583     5326   0.67  0.502
x2.hdl^2        1732     1591   1.09  0.277
x2.tch^2       773.4    607.0   1.27  0.203
x2.ltg^2        1452     1730   0.84  0.402
x2.glu^2      114.15    94.12   1.21  0.226
x2.age:sex    148.68    73.41   2.03  0.044
x2.age:bmi    -18.05    79.62  -0.23  0.821
x2.age:map     18.53    76.30   0.24  0.808
x2.age:tc     -158.9    617.1  -0.26  0.797
x2.age:ldl     -67.3    494.5  -0.14  0.892
x2.age:hdl     209.2    280.6   0.75  0.456
x2.age:tch     185.0    210.3   0.88  0.380
x2.age:ltg     124.7    223.8   0.56  0.578
x2.age:glu     62.58    80.38   0.78  0.437
x2.sex:bmi     64.61    77.90   0.83  0.407
x2.sex:map     88.47    74.74   1.18  0.237
x2.sex:tc      433.6    590.7   0.73  0.463
x2.sex:ldl    -352.8    469.0  -0.75  0.452
x2.sex:hdl    -124.7    273.9  -0.46  0.649
x2.sex:tch    -131.2    199.7  -0.66  0.512
x2.sex:ltg    -119.0    226.5  -0.53  0.600
x2.sex:glu     45.76    73.65   0.62  0.535
x2.bmi:map    154.72    86.34   1.79  0.074
x2.bmi:tc     -302.0    667.9  -0.45  0.651
x2.bmi:ldl     241.5    561.0   0.43  0.667
x2.bmi:hdl     121.9    329.9   0.37  0.712
x2.bmi:tch     -33.4    230.8  -0.14  0.885
x2.bmi:ltg     114.7    256.0   0.45  0.654
```

```
x2.bmi:glu    23.38    91.04   0.26  0.797
x2.map:tc     478.3    682.3   0.70  0.484
x2.map:ldl   -326.7    574.3  -0.57  0.570
x2.map:hdl   -187.3    309.6  -0.61  0.546
x2.map:tch    -58.3    198.6  -0.29  0.769
x2.map:ltg   -154.8    272.0  -0.57  0.570
x2.map:glu  -133.48    91.31  -1.46  0.145
x2.tc:ldl     -9314    11771  -0.79  0.429
x2.tc:hdl     -3932     3817  -1.03  0.304
x2.tc:tch     -2206     1762  -1.25  0.211
x2.tc:ltg     -3801    13166  -0.29  0.773
x2.tc:glu    -176.3    595.5  -0.30  0.767
x2.ldl:hdl     2643     3166   0.83  0.404
x2.ldl:tch     1207     1471   0.82  0.412
x2.ldl:ltg     2774    10960   0.25  0.800
x2.ldl:glu     85.6    505.1   0.17  0.865
x2.hdl:tch     1188     1002   1.19  0.236
x2.hdl:ltg     1468     4610   0.32  0.750
x2.hdl:glu    217.5    296.7   0.73  0.464
x2.tch:ltg    389.8    624.7   0.62  0.533
x2.tch:glu    235.7    235.1   1.00  0.317
x2.ltg:glu     83.5    264.7   0.32  0.753
```

One baffling question regarding R is the format of its output. R is very efficient in the output it produces on the screen but there are always much more results from calling a function in R than are just printed out by calling the resulting variable. For example, the t.test function returns a list, so to apply a function to every elements of this list we use lapply(), also we can use sapply to extract elements from the t.test results, such as the bounds of the confidence interval.

B Linear Regression- biggest model with no selection

```
The regression equation is
y = 152 + 50.7 x.age - 267 x.sex + 461 x.bmi + 343 x.map - 3600 x.tc
    + 3028 x.ldl + 1103 x.hdl + 75 x.tch + 1828 x.ltg + 62.8 x.glu
    + 67.7 x2.age^2 + 45.8 x2.bmi^2 - 8.5 x2.map^2 + 6668 x2.tc^2
    + 3583 x2.ldl^2 + 1732 x2.hdl^2 + 773 x2.tch^2 + 1452 x2.ltg^2
    + 114 x2.glu^2 + 149 x2.age:sex - 18.1 x2.age:bmi + 18.5 x2.age:map
    - 159 x2.age:tc - 67 x2.age:ldl + 209 x2.age:hdl + 185 x2.age:tch
    + 125 x2.age:ltg + 62.6 x2.age:glu + 64.6 x2.sex:bmi + 88.5 x2.sex:map
    + 434 x2.sex:tc - 353 x2.sex:ldl - 125 x2.sex:hdl - 131 x2.sex:tch
    - 119 x2.sex:ltg + 45.8 x2.sex:glu + 155 x2.bmi:map - 302 x2.bmi:tc
    + 242 x2.bmi:ldl + 122 x2.bmi:hdl - 33 x2.bmi:tch + 115 x2.bmi:ltg
    + 23.4 x2.bmi:glu + 478 x2.map:tc - 327 x2.map:ldl - 187 x2.map:hdl
    - 58 x2.map:tch - 155 x2.map:ltg - 133 x2.map:glu - 9314 x2.tc:ldl
    - 3932 x2.tc:hdl - 2206 x2.tc:tch - 3801 x2.tc:ltg - 176 x2.tc:glu
    + 2643 x2.ldl:hdl + 1207 x2.ldl:tch + 2774 x2.ldl:ltg + 86 x2.ldl:glu
    + 1188 x2.hdl:tch + 1468 x2.hdl:ltg + 218 x2.hdl:glu + 390 x2.tch:ltg
    + 236 x2.tch:glu + 84 x2.ltg:glu
```

```
> fit.lm$coefficients #regression coefficients
(Intercept)        Xage        Xsex        Xbmi        Xmap         Xtc        Xldl
 153.530371  -69.625820 -227.237883  590.798840  317.550724 -263.885885  -11.655818
       Xhdl        Xtch        Xltg        Xglu
 -86.473205  209.598954  555.123020    4.785505
```

Note: Getting regression statistics in R requires some further "asking-for-it". The important information is available from the model object return of the lm() by the summary() . However there are specialized extractor functions for other important information like coef() for Model coefficients, confint() for confidence intervals for model coefficient, anova() for the ANOVA table and so forth. For a complete list, look up ?lm() in the R package.