Linear Regression in R

Anurag Nagar

CS 6301

Getting Started

Regression involves learning to predict real-valued or continuous output variable. R has some really good packages for regression. First of all, let's make sure we have the required packages:

```
require(MASS)
```

```
## Loading required package: MASS
```

```
require(ISLR)
```

```
## Loading required package: ISLR
```

If you don't have the above packages, be sure to install them before proceeding.

Boston Housing Dataset

We will work with the famous Boston Housing dataset. Let's examine it:

```
View(Boston)
names(Boston)
```

Visualizing and Plotting the Data

It is very useful to find correlation between variables and see how they are correlated to each other and to the output variable:

```
cor(Boston$crim, Boston$medv)
```

```
## [1] -0.3883046
```

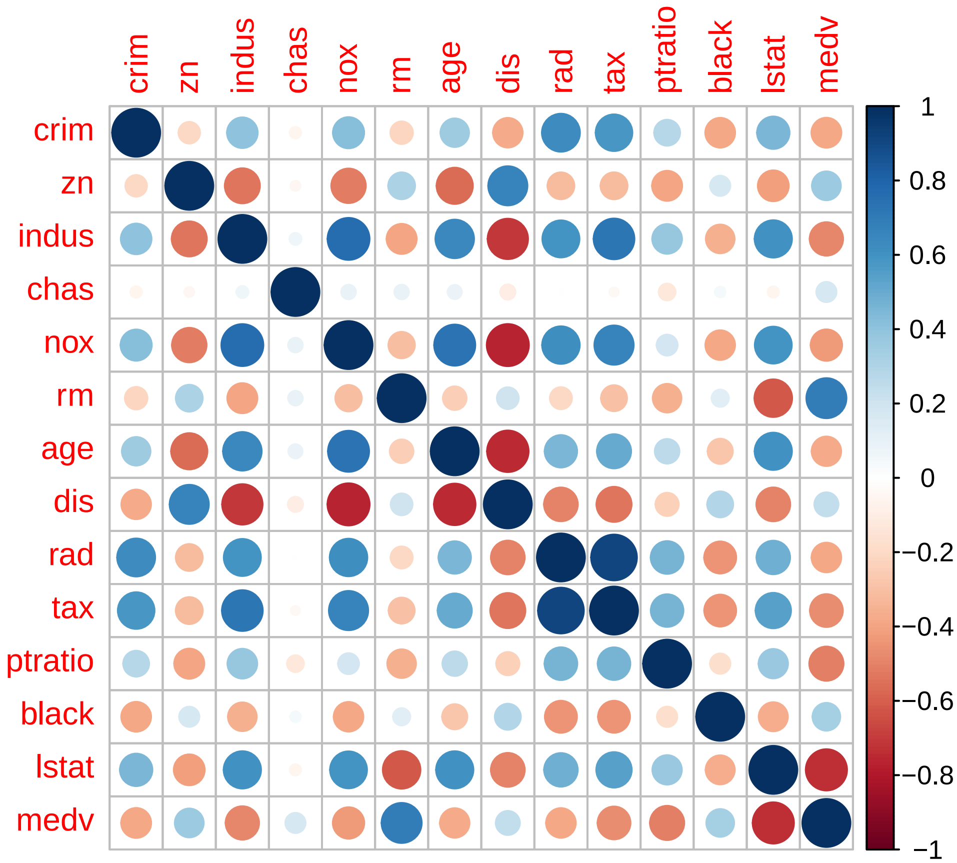
Let's find a better way of doing this:

```
require(corrplot)
```

```
## Loading required package: corrplot
```

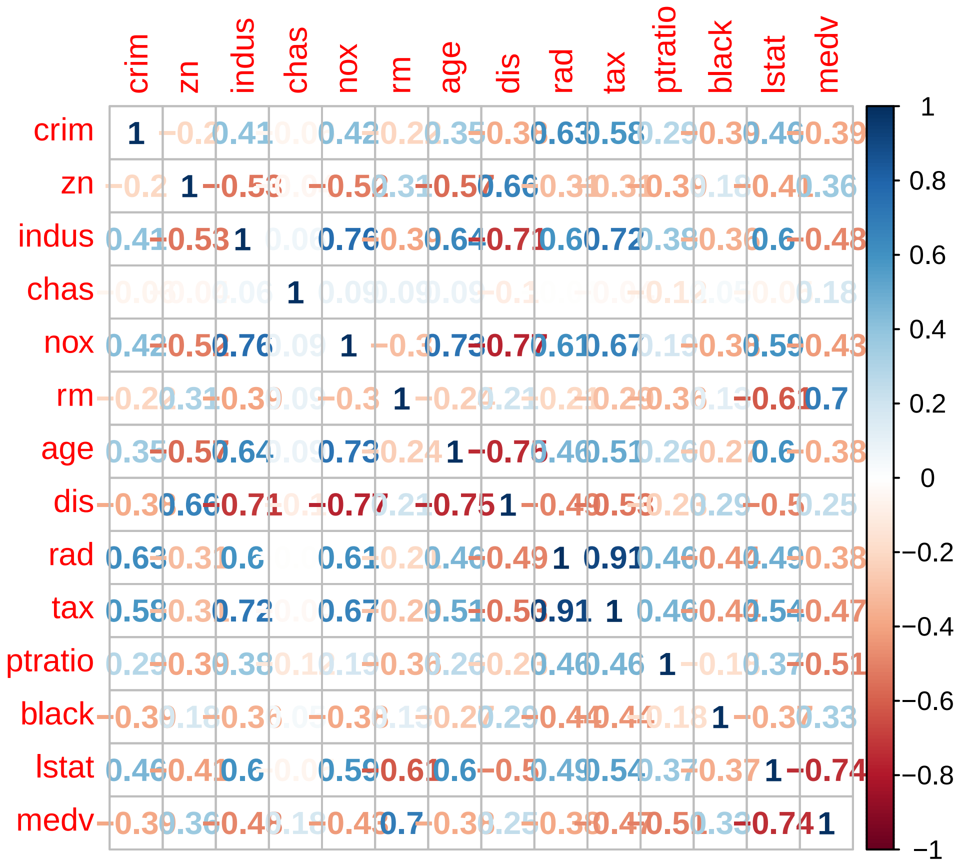
```
## corrplot 0.84 loaded
```

```
M <- cor(Boston)
corrplot(M, method = "circle")
```



Note how the color intensity indicates correlation

```
corMat <- as.data.frame(corrplot(M,method = "number"))
```




```r
names(corMat) <- names(Boston)
```

Find out which attributes have correlation of more than 50% with MEDV

```r
row.names(corMat)[abs(corMat$medv) > 0.50]
```

```
## [1] "rm"      "ptratio" "lstat"   "medv"
```

Predicting Home Prices

We will try to predict the median home value *medv* as a function of other variables. Let's just start with one variable:

```r
lm.fit=lm(medv~lstat,data=Boston)
attach(Boston)
lm.fit=lm(medv~lstat)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = medv ~ lstat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.168  -3.990  -1.318   2.034  24.500
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.55384    0.56263   61.41   <2e-16 ***
## lstat       -0.95005    0.03873  -24.53   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.216 on 504 degrees of freedom
## Multiple R-squared:  0.5441, Adjusted R-squared:  0.5432
## F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

Let's try to figure out what each of the above value means:

1. **Call** - gives the formula of the model.
2. **Residuals** - are defined as $\hat{y} - y$ where $\hat{y}$ is the predicted value and y is the actual value. The distribution of residuals is shown here. A large range implies the errors are high, and in all directions.
3. **Coefficients (β)** - gives the predicted coefficients for each variable and the constant (intercept) term.
4. **Standard Error ($se(\beta)$)** - measure of the variability in the estimate for the coefficient. A lower value, as compared with the coefficient, is a better indicator. The 95% confidence interval is defined as the region $\beta \pm 2se(\beta)$
5. **t-Value** - defined as the estimated coefficient divided by its standard error $\beta/se(\beta)$. A large value indicates a good precision, i.e. we are sure of the coefficient and its standard error is relatively small as compared to coefficient's value. It is used to test the hypothesis that the true value of the coefficient is non-zero, in order to confirm that the independent variable really belongs in the model.
6. **p-value** - indicates the probability of getting the obtained t-value if the null hypotesis were true. A smaller probability gives you more evidence that you can reject the null hypothesis and claim that the variable plays a significant role. The stars next to the row indicate how significant is the variable. 3 stars mean a very low p-value and we can safely reject the null hypothesis.
7. **Residual Standard Error(RSS)** - The Residual Std Error is just the standard deviation of your residuals. You'd like this number to be proportional to the quantiles of the residuals in part 2. For a normal distribution, the 1st and 3rd quantiles should be 1.5 +/- the std error.
8. **Degrees of Freedom** - The Degrees of Freedom is the difference between the number of observations included in your training sample and the number of variables used in your model (intercept counts as a variable). It's used in conjunction with the RSS estimate above.
9. **R Squared and Adjusted R Squared** - R squared isa statistical measure of how close the data are to the fitted regression line. It is also equal to the percent of the total variation in the dependent variable (y) that is explained by the independent variables (X), i.e., the model's overall "goodness of fit". It is possible to get higher R squared by simply adding more independent variables. Adjusted R Squared is a modified version of R-squared that has been adjusted for the number of predictors in the model. The adjusted R-squared increases only if the new term improves the model more than would be expected by chance. Suppose you have a 1-predictor and 5-predictor models. Does the five predictor model have a higher R-squared because it's better? Or is the R-squared higher because it has more predictors? Adjusted R squared provides the answer for this.

10 **F-statistic and its p-value** Performs an F-test on the model. This takes the parameters of our model and compares it to a model that has fewer parameters. In theory the model with more parameters should fit better. If the model with more parameters (your model) doesn't perform better than the model with fewer parameters, the F-test will have a high p-value (probability NOT significant boost). If the model with more parameters is better than the model with fewer parameters, you will have a lower p-value.

```r
names(lm.fit)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
```

```
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

```r
coef(lm.fit)
```

```
## (Intercept)       lstat
##  34.5538409  -0.9500494
```

```r
confint(lm.fit)
```

```
##                 2.5 %     97.5 %
## (Intercept) 33.448457 35.6592247
## lstat       -1.026148 -0.8739505
```

Let's do some prediction on test data:

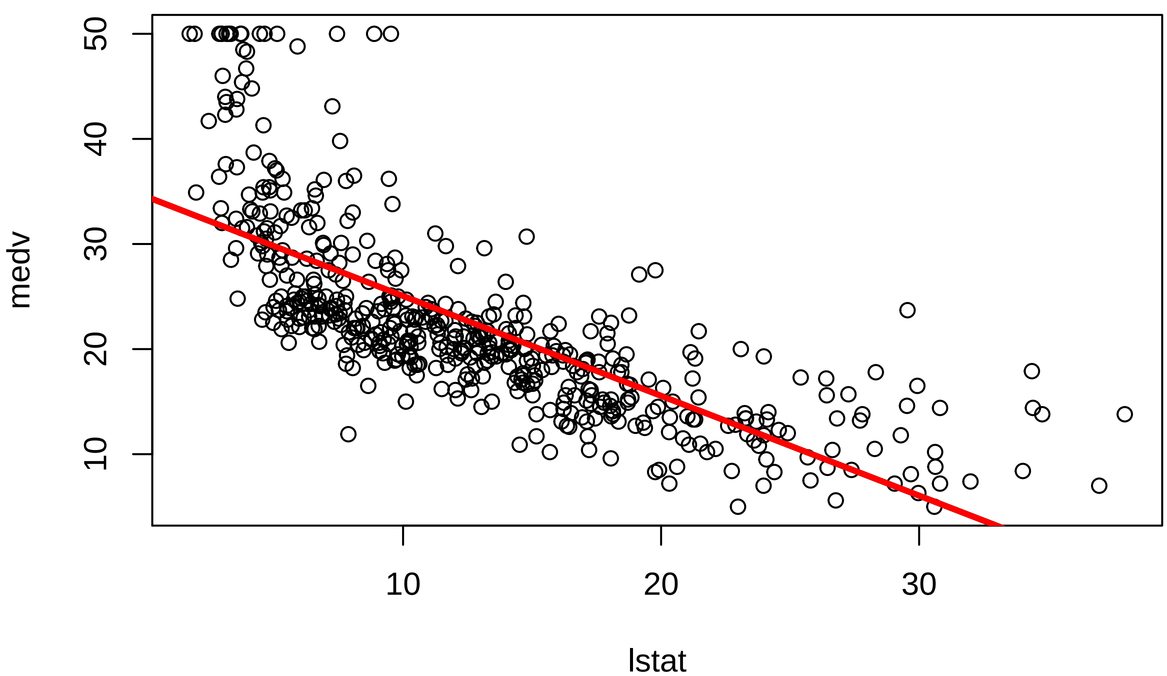
```r
predict(lm.fit,data.frame(lstat=(c(5,10,15))), interval="confidence")
```

```
##        fit      lwr      upr
## 1 29.80359 29.00741 30.59978
## 2 25.05335 24.47413 25.63256
## 3 20.30310 19.73159 20.87461
```

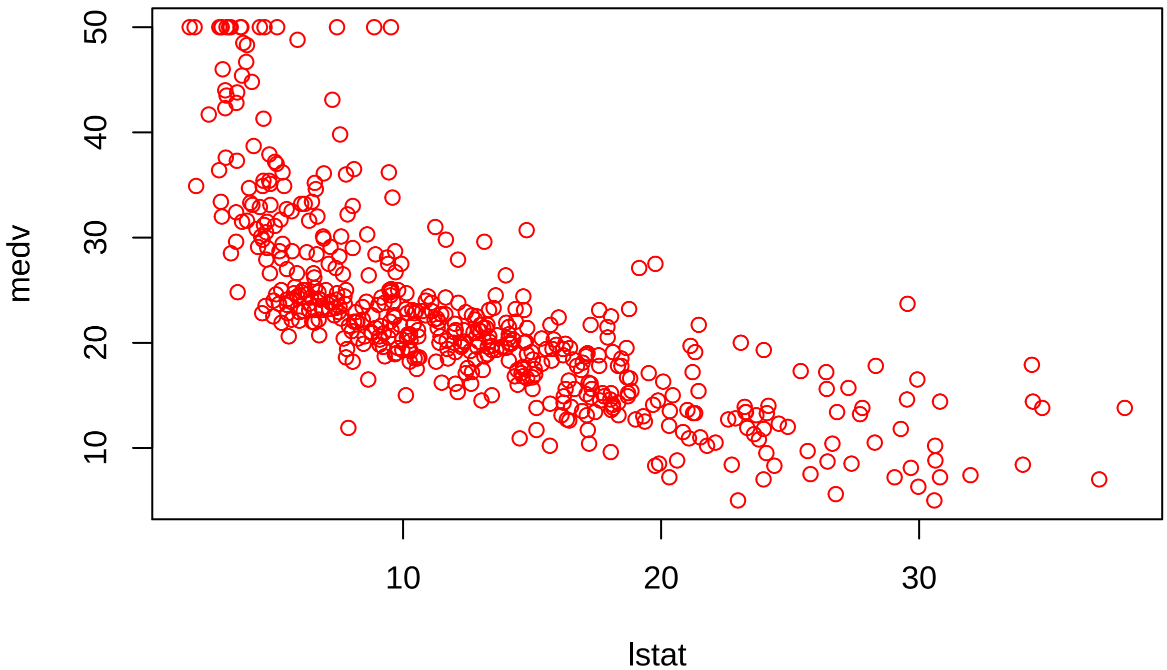
```r
predict(lm.fit,data.frame(lstat=(c(5,10,15))), interval="prediction")
```

```
##        fit       lwr      upr
## 1 29.80359 17.565675 42.04151
## 2 25.05335 12.827626 37.27907
## 3 20.30310  8.077742 32.52846
```

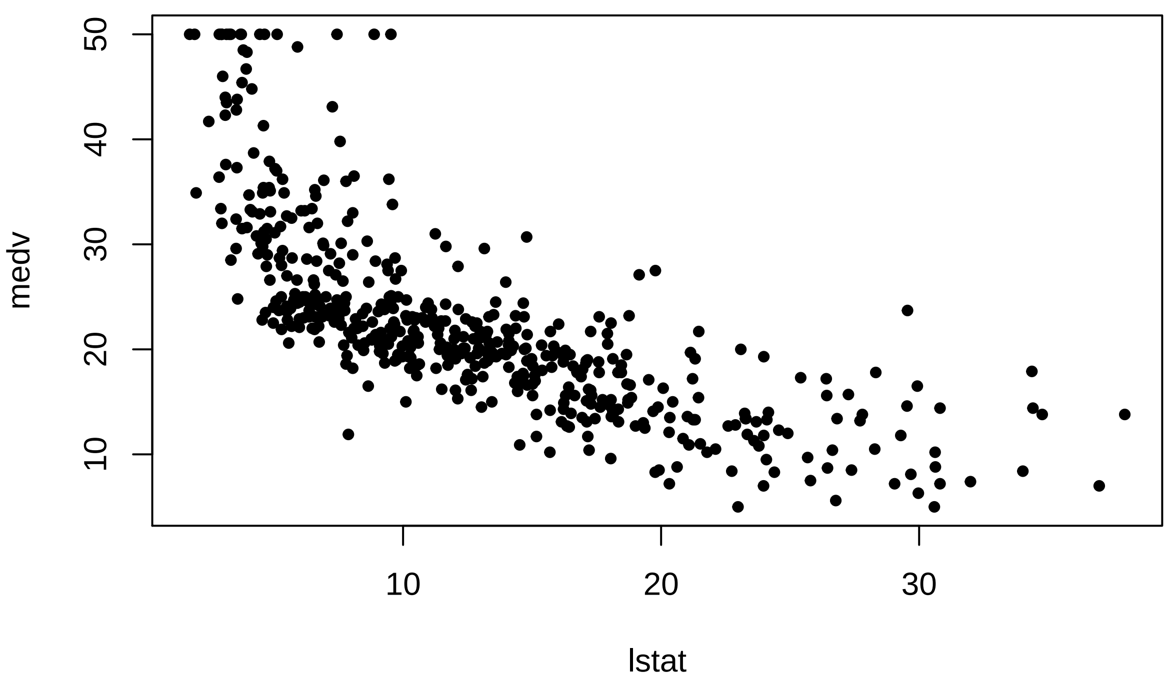
```r
plot(lstat,medv)
abline(lm.fit)
abline(lm.fit,lwd=3)
abline(lm.fit,lwd=3,col="red")
```



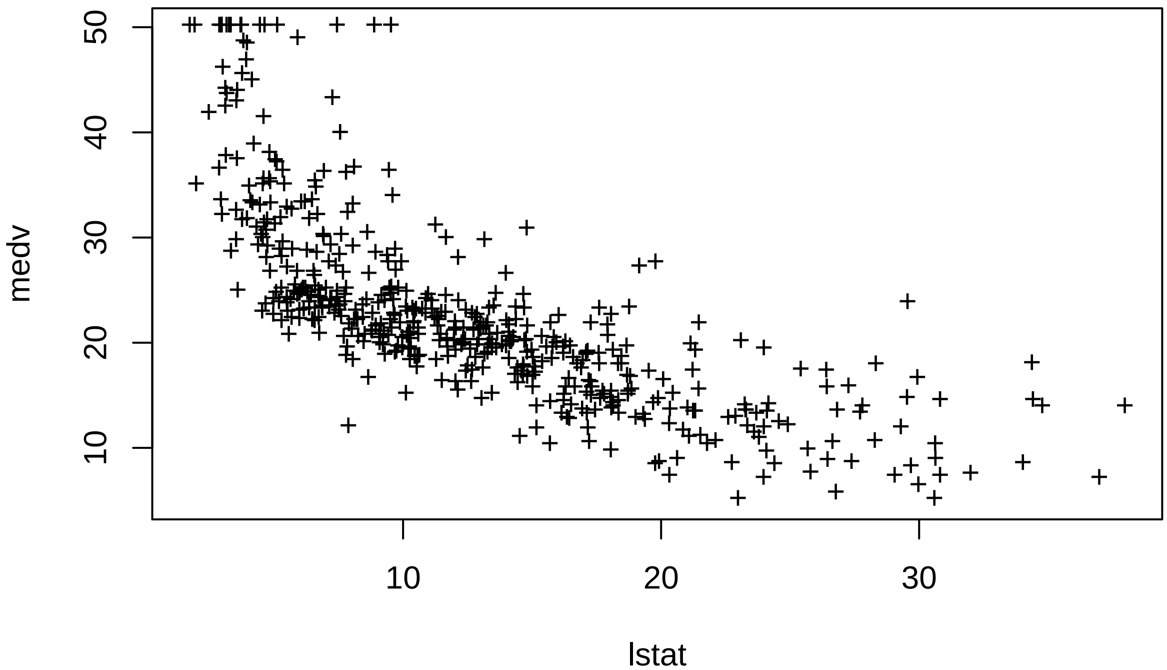

```r
plot(lstat,medv,col="red")
```



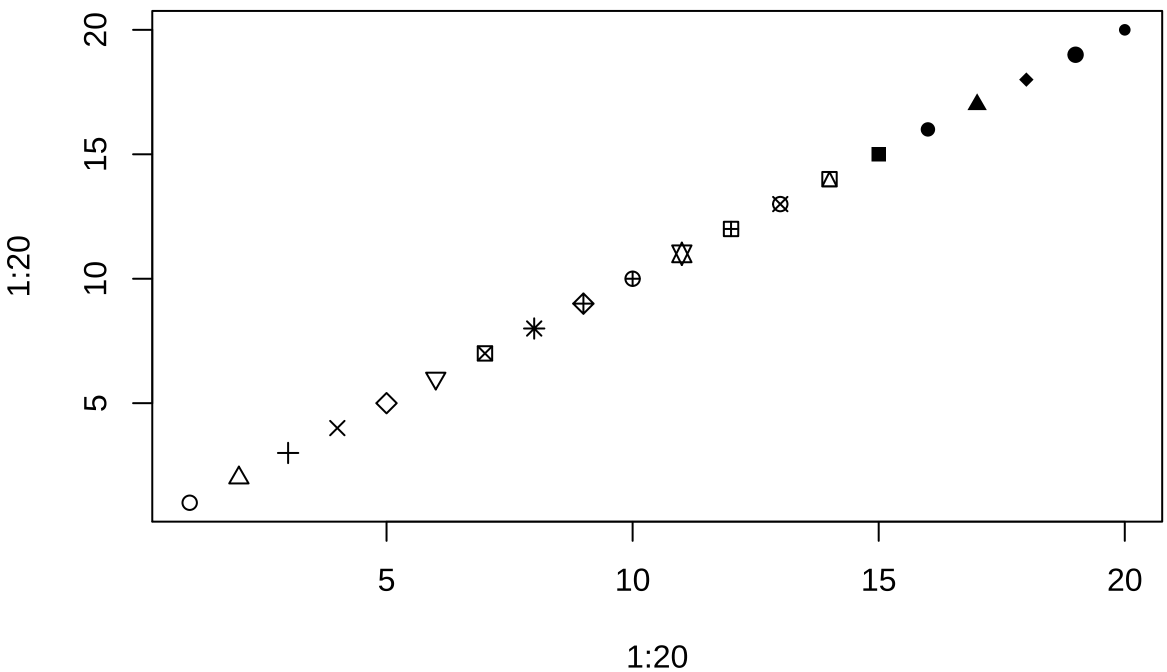

```r
plot(lstat,medv,pch=20)
```




```r
plot(lstat,medv,pch="+")
```




```r
plot(1:20,1:20,pch=1:20)
```



Multiple Linear Regression

Let's work with more than one attributes.

```r
lm.fit=lm(medv~lstat+age,data=Boston)
summary(lm.fit)
```

```
##
## Call:
```

```
## lm(formula = medv ~ lstat + age, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.981  -3.978  -1.283   1.968  23.158
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 33.22276    0.73085  45.458  < 2e-16 ***
## lstat       -1.03207    0.04819 -21.416  < 2e-16 ***
## age          0.03454    0.01223   2.826  0.00491 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.173 on 503 degrees of freedom
## Multiple R-squared:  0.5513, Adjusted R-squared:  0.5495
## F-statistic:   309 on 2 and 503 DF,  p-value: < 2.2e-16
```

```
lm.fit=lm(medv~.,data=Boston)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = medv ~ ., data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas         2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## black        9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

```
lm.fit1=lm(medv~.-age,data=Boston)
summary(lm.fit1)
```

```
##
## Call:
## lm(formula = medv ~ . - age, data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.6054  -2.7313  -0.5188   1.7601  26.2243
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.436927   5.080119   7.172 2.72e-12 ***
## crim         -0.108006   0.032832  -3.290 0.001075 **
## zn            0.046334   0.013613   3.404 0.000719 ***
## indus         0.020562   0.061433   0.335 0.737989
## chas          2.689026   0.859598   3.128 0.001863 **
## nox         -17.713540   3.679308  -4.814 1.97e-06 ***
## rm            3.814394   0.408480   9.338  < 2e-16 ***
## dis          -1.478612   0.190611  -7.757 5.03e-14 ***
## rad           0.305786   0.066089   4.627 4.75e-06 ***
## tax          -0.012329   0.003755  -3.283 0.001099 **
## ptratio      -0.952211   0.130294  -7.308 1.10e-12 ***
## black         0.009321   0.002678   3.481 0.000544 ***
## lstat        -0.523852   0.047625 -10.999  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.74 on 493 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7343
## F-statistic: 117.3 on 12 and 493 DF,  p-value: < 2.2e-16
```

```r
lm.fit1=update(lm.fit, ~.-age)
```

Is linear and independent assumption always valid?

Let's create a variable that is product of *lstat* and *age* variables. Does the summary tell you something?

```r
summary(lm(medv~lstat*age,data=Boston))
```

```
##
## Call:
## lm(formula = medv ~ lstat * age, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.806  -4.045  -1.333   2.085  27.552
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 36.0885359  1.4698355  24.553  < 2e-16 ***
## lstat       -1.3921168  0.1674555  -8.313 8.78e-16 ***
## age         -0.0007209  0.0198792  -0.036   0.9711
## lstat:age    0.0041560  0.0018518   2.244   0.0252 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 6.149 on 502 degrees of freedom
## Multiple R-squared:  0.5557, Adjusted R-squared:  0.5531
## F-statistic: 209.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

Let's try some other possibilities - like polynomial, log, etc. We will also check if there is significan difference between linear and non-linear model.

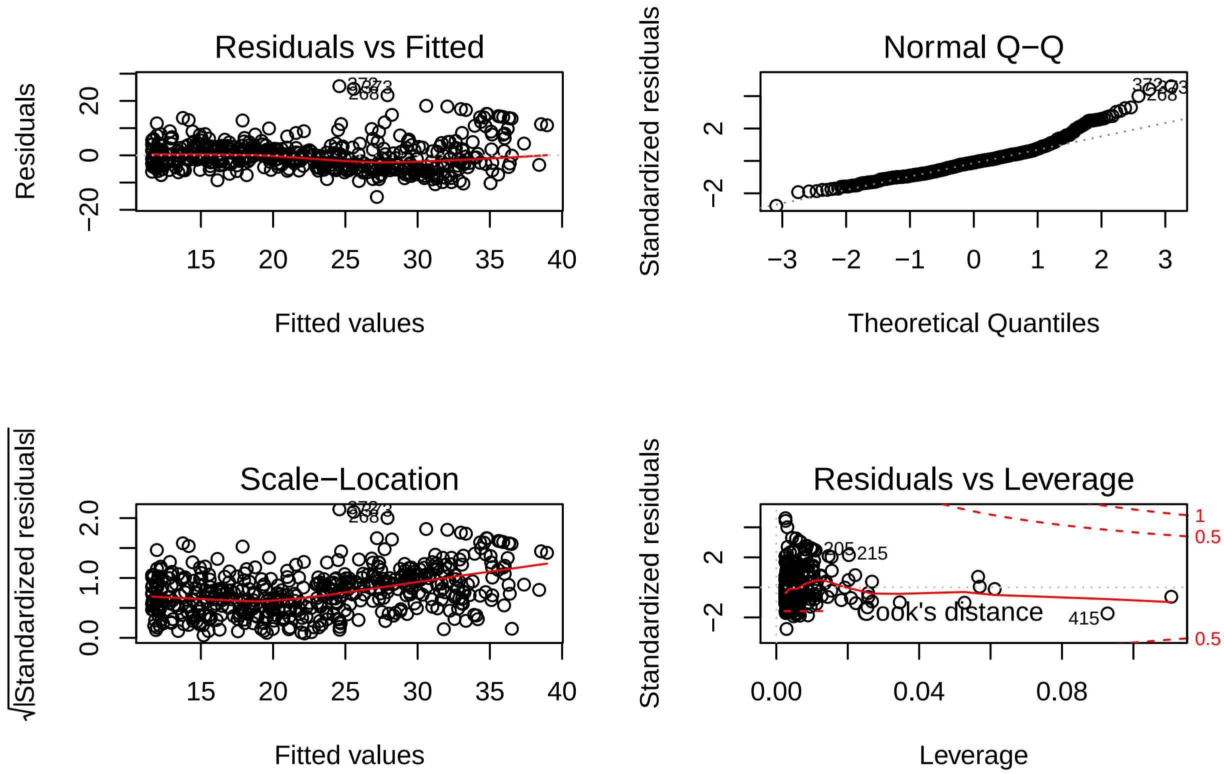
```r
lm.fit2=lm(medv~lstat+I(lstat^2))
summary(lm.fit2)
```

```
##
## Call:
## lm(formula = medv ~ lstat + I(lstat^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2834  -3.8313  -0.5295   2.3095  25.4148
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.862007   0.872084   49.15   <2e-16 ***
## lstat       -2.332821   0.123803  -18.84   <2e-16 ***
## I(lstat^2)   0.043547   0.003745   11.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.524 on 503 degrees of freedom
## Multiple R-squared:  0.6407, Adjusted R-squared:  0.6393
## F-statistic: 448.5 on 2 and 503 DF,  p-value: < 2.2e-16
```

```r
lm.fit=lm(medv~lstat)
anova(lm.fit,lm.fit2)
```

```
## Analysis of Variance Table
##
## Model 1: medv ~ lstat
## Model 2: medv ~ lstat + I(lstat^2)
##   Res.Df   RSS Df Sum of Sq     F    Pr(>F)
## 1    504 19472
## 2    503 15347  1    4125.1 135.2 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
par(mfrow=c(2,2))
plot(lm.fit2)
```




```r
lm.fit5=lm(medv~poly(lstat,5))
summary(lm.fit5)
```

```
##
## Call:
## lm(formula = medv ~ poly(lstat, 5))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.5433  -3.1039  -0.7052   2.0844  27.1153
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       22.5328     0.2318  97.197  < 2e-16 ***
## poly(lstat, 5)1 -152.4595     5.2148 -29.236  < 2e-16 ***
## poly(lstat, 5)2   64.2272     5.2148  12.316  < 2e-16 ***
## poly(lstat, 5)3  -27.0511     5.2148  -5.187 3.10e-07 ***
## poly(lstat, 5)4   25.4517     5.2148   4.881 1.42e-06 ***
## poly(lstat, 5)5  -19.2524     5.2148  -3.692 0.000247 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.215 on 500 degrees of freedom
## Multiple R-squared:  0.6817, Adjusted R-squared:  0.6785
## F-statistic: 214.2 on 5 and 500 DF,  p-value: < 2.2e-16
```

```r
summary(lm(medv~log(rm),data=Boston))
```

```
##
## Call:
```

```
## lm(formula = medv ~ log(rm), data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.487  -2.875  -0.104   2.837  39.816
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -76.488      5.028  -15.21   <2e-16 ***
## log(rm)       54.055      2.739   19.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.915 on 504 degrees of freedom
## Multiple R-squared:  0.4358, Adjusted R-squared:  0.4347
## F-statistic: 389.3 on 1 and 504 DF,  p-value: < 2.2e-16
```

What if data is not linear

If your dataset is not linear, you can use the *glm* function, which has many more option.

See this link: https://www.rdocumentation.org/packages/stats/versions/3.5.2/topics/glm

Exercise

One of the most popular challenges in Kaggle is house price prediction. In the class assignment, you can any of the datasets below and perform regression analysis. Include a report of your results, with plots and summary data. Also, indicate what interesting information did you obtain

1. California Housing Dataset https://www.kaggle.com/camnugent/california-housing-prices
2. Boston Housing dataset https://www.kaggle.com/vikrishnan/boston-house-prices
3. Advanced Regression Techniques competition https://www.kaggle.com/c/house-prices-advanced-regression-techniques/overview
4. Zillow Home Price Dataset https://www.kaggle.com/c/zillow-prize-1
5. Russian Housing Dataset https://www.kaggle.com/c/sberbank-russian-housing-market
6. Melbourne Housing Dataset https://www.kaggle.com/anthonypino/melbourne-housing-market
7. Ames Housing Dataset https://www.kaggle.com/c/ames-housing-data

Submission Details

Following are submission details

1. You are allowed to work in groups of 1 - 4 students.
2. Treat this as a data science project. Show as many statistics and plots as you can.
3. Submit your R code file and report file. Please do not hard code any paths in your code. You can put the data in your UTD web account and read from that link.
4. Be sure to detail any interesting findings.