Template for Multiple Regression with Anova

James Long, Shyam Attiganal, Manoj Remela, Eric Young

September 26, 2019

Intro

This template shows how to create multiple regression models, remove high influencers, and validate assumptions.

- The example has two predictor variables.
- The example has a curved relationship between one predictor variable and the response variable.
- The example does **not** have interaction between the predictor variables.
- The example does **not** contain categorical variables.
- The example has irremovable collinearity.

File: 6304W Live Session 2 Data Sets.xlsx
Sheet: MPG

Pre-processing

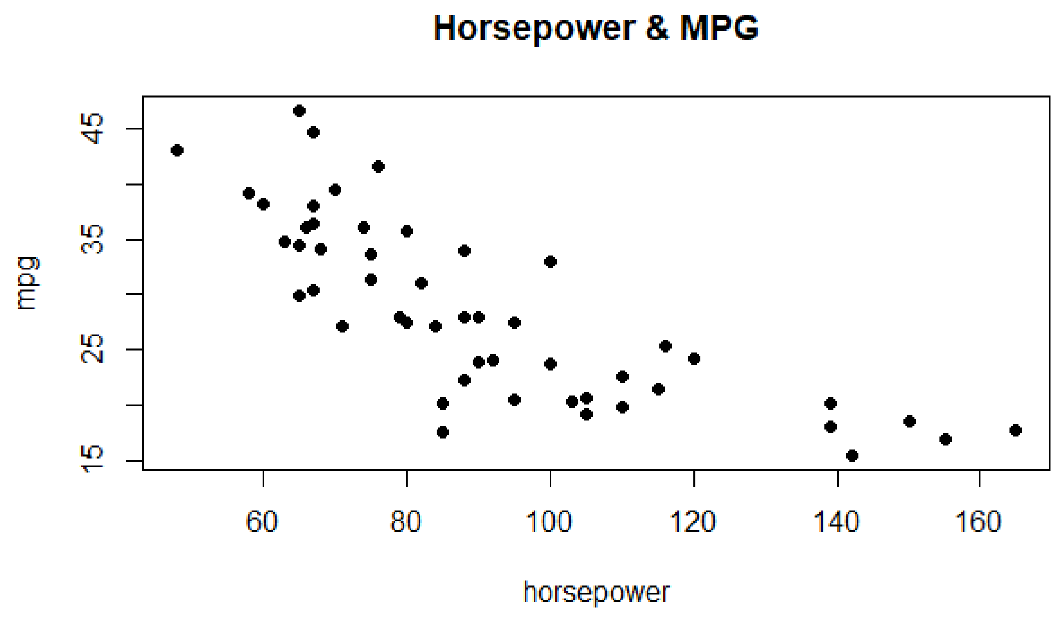
```
# setup environment and load data
setwd(params$wd)
library(rio)
# anova & Durbin-Watson autocorrelation module
library(car)

## Loading required package: carData

# partial correlation module
library(ppcor)

## Loading required package: MASS

cars=import("6304W Live Session 2 Data Sets.xlsx",sheet="MPG")
colnames(cars)=tolower(make.names(colnames(cars)))
attach(cars)
names(cars)
```

```
## [1] "mpg"        "horsepower" "weight"

# look for "chr" classes that should be coerced to "factor"
str(cars)

## 'data.frame':    50 obs. of  3 variables:
##  $ mpg       : num  43.1 19.9 19.2 17.7 18.1 20.3 21.5 16.9 15.5 18.5 ...
##  $ horsepower: num  48 110 105 165 139 103 115 155 142 150 ...
##  $ weight    : num  1985 3365 3535 3445 3205 ...

# use the following syntax to change classes
# dataframe$columnname=as.factor(dataframe$columnname)

# if changes are made, verify changes
# str(cars)
```

Explore to Find and Select the Best Model

- The step() function can be used to expedite our exploration of the various models as long as no transforms, no interaction terms, and no categorical variables are required (or categorical variables are transformed into binary representation).

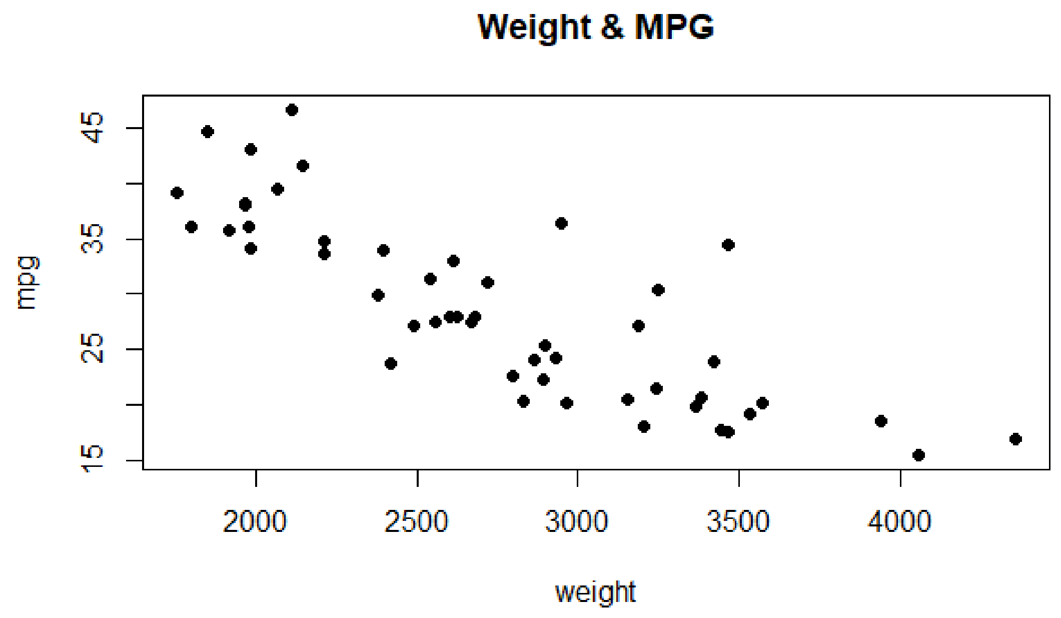
```
# inspect each predictor variable for a non-linear relationship
plot(horsepower,mpg,pch=19,main="Horsepower & MPG")
```



```
# use cor.test() to explore data, use corrplot() to present findings
cor.test(horsepower,mpg)

##
##  Pearson's product-moment correlation
##
## data:  horsepower and mpg
## t = -8.8726, df = 48, p-value = 1.093e-11
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.8746402 -0.6531149
## sample estimates:
##        cor
## -0.7881775
```

- A curve is present in the HP - MPG relationship, so we will include a transform for HP.

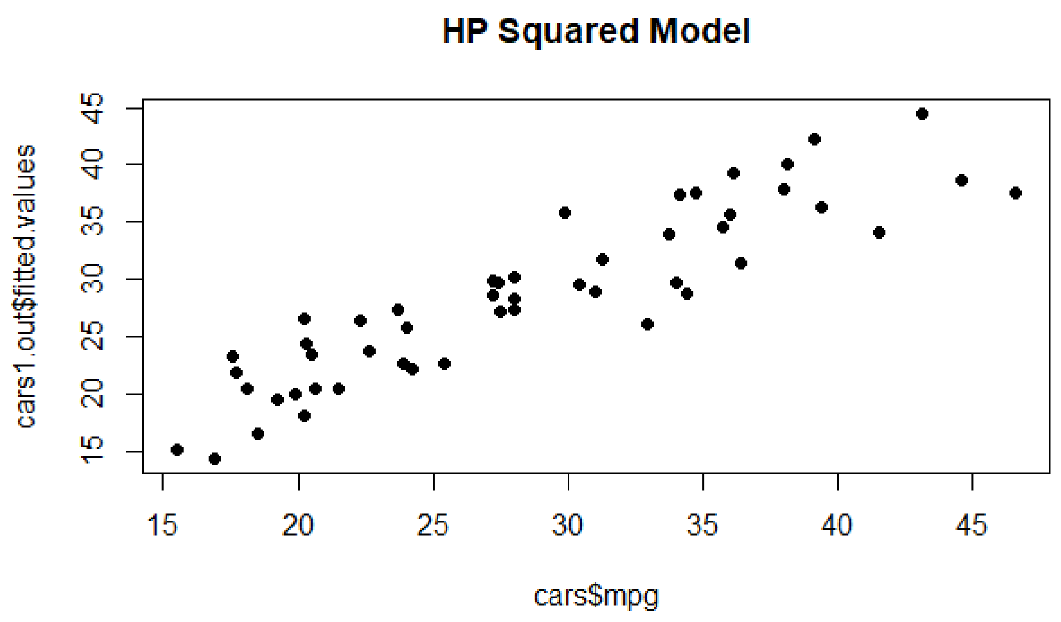
```
plot(weight,mpg,pch=19,main="Weight & MPG")
```



```
cor.test(weight,mpg)

##
##  Pearson's product-moment correlation
##
## data:  weight and mpg
## t = -10.107, df = 48, p-value = 1.787e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.8971788 -0.7093059
## sample estimates:
##        cor
## -0.8248086
```

- A curve might be present in the weight - MPG relationship, so we will conduct further testing using a transform for weight.

```
# create a model with a transform for HP
cars1.out=lm(mpg~horsepower+weight+I(horsepower^2),data=cars)
cars1.sum=summary(cars1.out)
cars1.ci=data.frame(signif(confint(cars1.out),digits=2))
cars1.sum

##
## Call:
## lm(formula = mpg ~ horsepower + weight + I(horsepower^2), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3658 -2.5898 -0.1787  2.0400  9.0705
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     82.5770060  6.3148979  13.077  < 2e-16 ***
## horsepower      -0.6435213  0.1298327  -4.957 1.02e-05 ***
## weight          -0.0065175  0.0012111  -5.382 2.42e-06 ***
## I(horsepower^2)  0.0024931  0.0006007   4.150 0.000142 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.601 on 46 degrees of freedom
## Multiple R-squared:  0.8177, Adjusted R-squared:  0.8058
## F-statistic: 68.77 on 3 and 46 DF,  p-value: < 2.2e-16

cars1.ci

##                  X2.5.. X97.5..
## (Intercept)     70.0000 95.0000
## horsepower      -0.9000 -0.3800
## weight          -0.0090 -0.0041
## I(horsepower^2)  0.0013  0.0037

# verify no curves
plot(cars$mpg,cars1.out$fitted.values,pch=19, main="HP Squared Model")
```



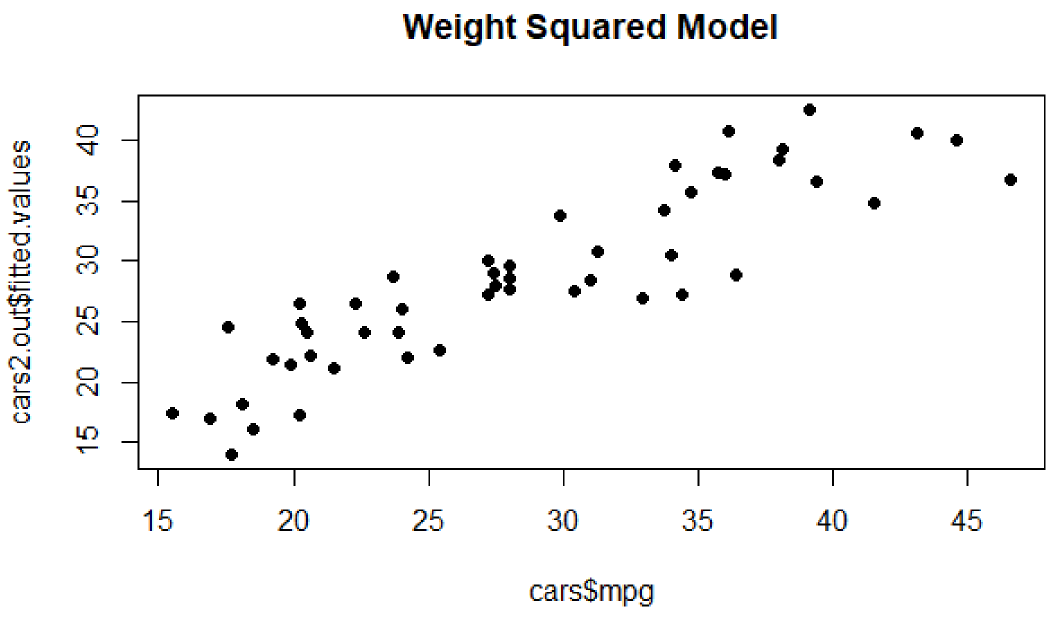
- All predictor variables are significant, and the curve was removed.

```
# create a model with a transform for weight
# look for significance
cars2.out=lm(mpg~horsepower+weight+I(weight^2),data=cars)
cars2.sum=summary(cars2.out)
cars2.ci=data.frame(signif(confint(cars2.out),digits=2))
cars2.sum

##
## Call:
## lm(formula = mpg ~ horsepower + weight + I(weight^2), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.0103 -2.0454 -0.5139  2.5498  9.8437
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.912e+01  9.549e+00   9.334 3.48e-12 ***
## horsepower   -1.332e-01  2.995e-02  -4.447 5.48e-05 ***
## weight       -2.907e-02  6.744e-03  -4.310 8.52e-05 ***
## I(weight^2)   3.959e-06  1.181e-06   3.351  0.00162 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.785 on 46 degrees of freedom
## Multiple R-squared:  0.7986, Adjusted R-squared:  0.7855
## F-statistic:  60.8 on 3 and 46 DF,  p-value: 4.869e-16
```

```
cars2.ci
```

```
##                 X2.5..  X97.5..
## (Intercept)  7.0e+01  1.1e+02
## horsepower  -1.9e-01 -7.3e-02
## weight      -4.3e-02 -1.5e-02
## I(weight^2)  1.6e-06  6.3e-06
```

```
# verify no curves
plot(cars$mpg,cars2.out$fitted.values,pch=19, main="Weight Squared Model")
```



- All predictor variables are significant, and no curve is easily detectable.
- Since both squared terms appear to be significant, we combine them into single model to verify.

```
# create full model
cars3.out=lm(mpg~horsepower+weight+I(horsepower^2)+I(weight^2),data=cars)
cars3.sum=summary(cars3.out)
cars3.ci=data.frame(signif(confint(cars3.out),digits=3))
cars3.sum
```

```
##
## Call:
## lm(formula = mpg ~ horsepower + weight + I(horsepower^2) + I(weight^2),
##     data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0658 -2.1520 -0.3217  1.5284  9.0875
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      9.096e+01  9.050e+00  10.051 4.43e-13 ***
## horsepower      -5.282e-01  1.571e-01  -3.363  0.00158 **
## weight          -1.669e-02  8.003e-03  -2.085  0.04277 *
## I(horsepower^2)  1.913e-03  7.482e-04   2.557  0.01402 *
## I(weight^2)      1.799e-06  1.400e-06   1.285  0.20526
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.576 on 45 degrees of freedom
## Multiple R-squared:  0.8241, Adjusted R-squared:  0.8085
## F-statistic: 52.72 on 4 and 45 DF,  p-value: < 2.2e-16
```

```
cars3.ci
```

```
##                    X2.5..   X97.5..
## (Intercept)      7.27e+01  1.09e+02
## horsepower      -8.45e-01 -2.12e-01
## weight          -3.28e-02 -5.68e-04
## I(horsepower^2)  4.06e-04  3.42e-03
## I(weight^2)     -1.02e-06  4.62e-06
```

- The weight transform is not significant in the full model.
- We can confirm this with a partial F statistic.

```
# create partial F statistic
wgt.ano=anova(cars1.out,cars3.out)
wgt.ano
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ horsepower + weight + I(horsepower^2)
## Model 2: mpg ~ horsepower + weight + I(horsepower^2) + I(weight^2)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     46 596.53
## 2     45 575.40  1    21.124 1.6521 0.2053
```

- The high p-value of the F-statistic in the anova table indicates we cannot reject the null and that no significant relationship exists between weight^2 and MPG in the full model.

- The best model is "cars1.out" because it has the highest Adjusted R-squared value (0.806) of all the models that have only significant predictor variables.
- The coefficient of the intercept is 82.6 with a CI of 70 to 95 and a p-value of 4.1910^{-17}. Even though the p-value indicates significance, this coefficient is meaningless because it is impossible to have a car with zero horsepower and zero weight. Since our predictor variables do not include observations at zero, we are extrapolating to get an intercept value. The farther we predict outside the range of observed predictor variables, the less reliable our predictions are.
- The coefficient of horsepower is -0.644 with a CI of -0.9 to -0.38 and a p-value of 1.0210^{-5}. The low p-value indicates we can reject the null hypothesis that no relationship exists between horsepower and MPG. Our model predicts MPG will drop by 0.644 for each additional horsepower. Scaling this conclusion makes it more useful. Our model predicts MPG will drop by 6.44 for each additional 10 horsepower.
- The coefficient of weight is -0.00652 with a CI of -0.009 to -0.0041 and a p-value of 2.4210^{-6}. The low p-value indicates we can reject the null hypothesis that no relationship exists between weight and MPG. Our model predicts MPG will drop by 0.00652 for each additional pound of vehicle or cargo weight. Scaling this conclusion makes it more useful. Our model predicts MPG will drop by 1.3 for each additional 200 pounds of vehicle or cargo weight.
- The coefficient of horsepower^2 is 0.00249 with a CI of 0.0013 to 0.0037 and a p-value of 1.4210^{-4}. The low p-value indicates horsepower^2 is significant in this model. We can verify that by looking for linearity in the plot of observed vs. fitted MPG values (HP Squared Model). However, there is no meaningful interpretation for the coefficient.

Validate the Best Model

```
# test for interaction between predictor variables
cars4.out=lm(mpg~horsepower+weight+I(horsepower^2)+horsepower*weight,data=car
s)
# could also use the following syntax, shorter but not self-documenting
# cars4.out=lm(mpg~horsepower*weight+I(horsepower^2),data=cars)
cars4.sum=summary(cars4.out)
cars4.ci=data.frame(signif(confint(cars4.out),digits=0))
cars4.sum

##
## Call:
## lm(formula = mpg ~ horsepower + weight + I(horsepower^2) + horsepower *
##     weight, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3427 -2.5579 -0.2697  1.6790  8.9739
```

```
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        8.529e+01  7.984e+00  10.683 6.30e-14 ***
## horsepower        -6.406e-01  1.309e-01  -4.893 1.31e-05 ***
## weight            -8.662e-03  3.999e-03  -2.166   0.0356 *
## I(horsepower^2)    2.085e-03  9.439e-04   2.209   0.0323 *
## horsepower:weight  2.505e-05  4.449e-05   0.563   0.5762
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.628 on 45 degrees of freedom
## Multiple R-squared:  0.819,  Adjusted R-squared:  0.8029
## F-statistic: 50.89 on 4 and 45 DF,  p-value: 3.878e-16
```

```
cars4.ci
```

```
##                   X2.5.. X97.5..
## (Intercept)        7e+01   1e+02
## horsepower        -9e-01  -4e-01
## weight            -2e-02  -6e-04
## I(horsepower^2)    2e-04   4e-03
## horsepower:weight -6e-05   1e-04
```

- No interaction is found.
- While we are looking at interaction, we can also look at partial correlation to better understand the true influence of each predictor variable on the response variable when the predictors are modeled together.

```
# remove categorical variables (if any)
# cars.nocat=cars[,-colnum]

# determine partial correlation
parcor=pcor(cars)
parcor$estimate["horsepower","mpg"]
```

```
## [1] -0.4649413
```

```
parcor$estimate["weight","mpg"]
```

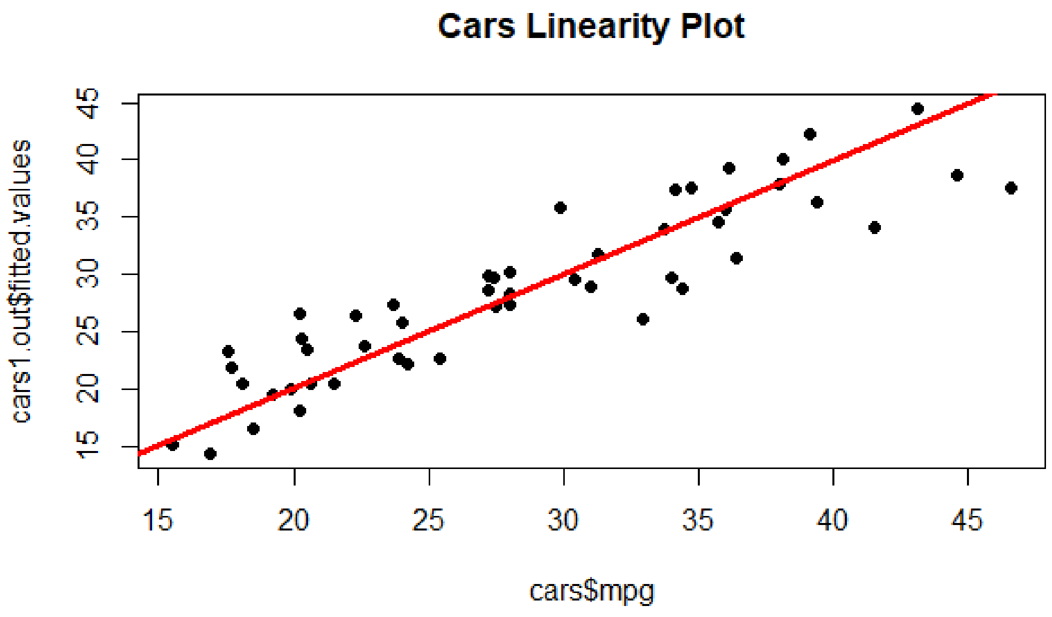
```
## [1] -0.5817564
```

- The partial correlation of each predictor variable is lower than the standard correlation of each predictor variable.
- Next we test for correlation and collinearity between predictor variables.

```
# test for correlation between predictor variables
cor.test(horsepower,weight)

##
##  Pearson's product-moment correlation
##
## data:  horsepower and weight
## t = 7.6652, df = 48, p-value = 7.084e-10
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5841526 0.8456048
## sample estimates:
##       cor
## 0.7418735

# check variance inflation factors to find collinearity in the model
# values should be 12 or less
vif(cars1.out)

##      horsepower          weight I(horsepower^2)
##       47.326313        2.235115       45.066617
```

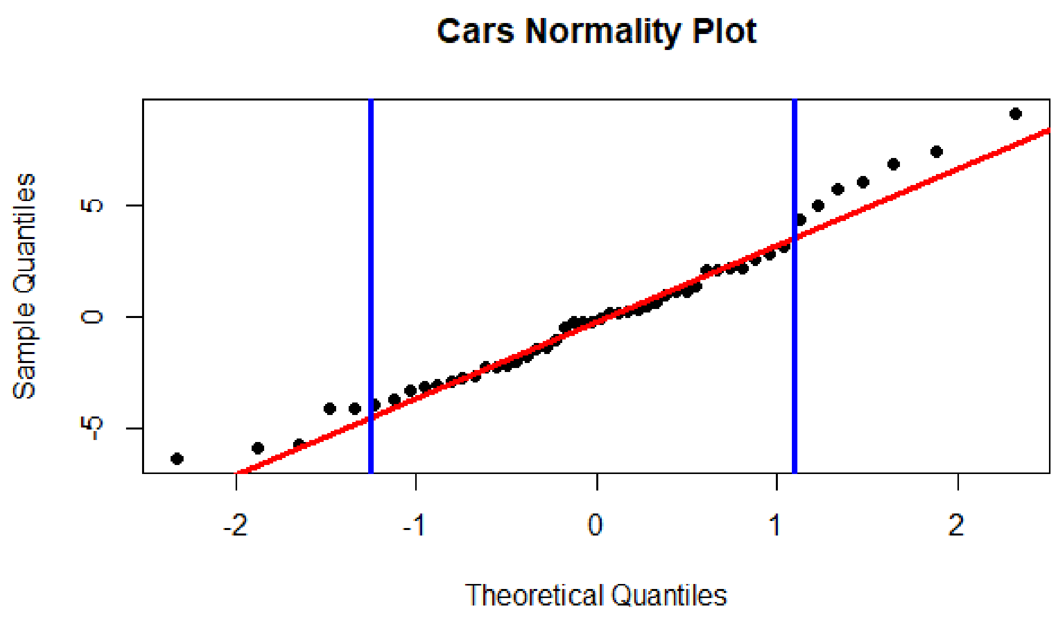
- Standard correlation has a low p-value, so we reject the null hypothesis and conclude a relationship does exist between horsepower and weight. Moreover, the correlation is high, which could indicate collinearity.
- If evidence of collinearity is present, eliminate one collinear variable from the model and check the results. Then check for collinearity again. You might need to iteratively replace and remove variables until you find the best model.
- No collinearity is found between horsepower and weight.
- HP and HP^2 are collinear as expected. We cannot remove HP without also removing HP^2. We cannot remove HP^2 without re-introducing the curved relationship. So, we move on to model assumptions.

```
# check linearity assumption
plot(cars$mpg,cars1.out$fitted.values,pch=19,main="Cars Linearity Plot")
abline(0,1,lwd=3,col="red")
```



- The linearity assumption appears valid.

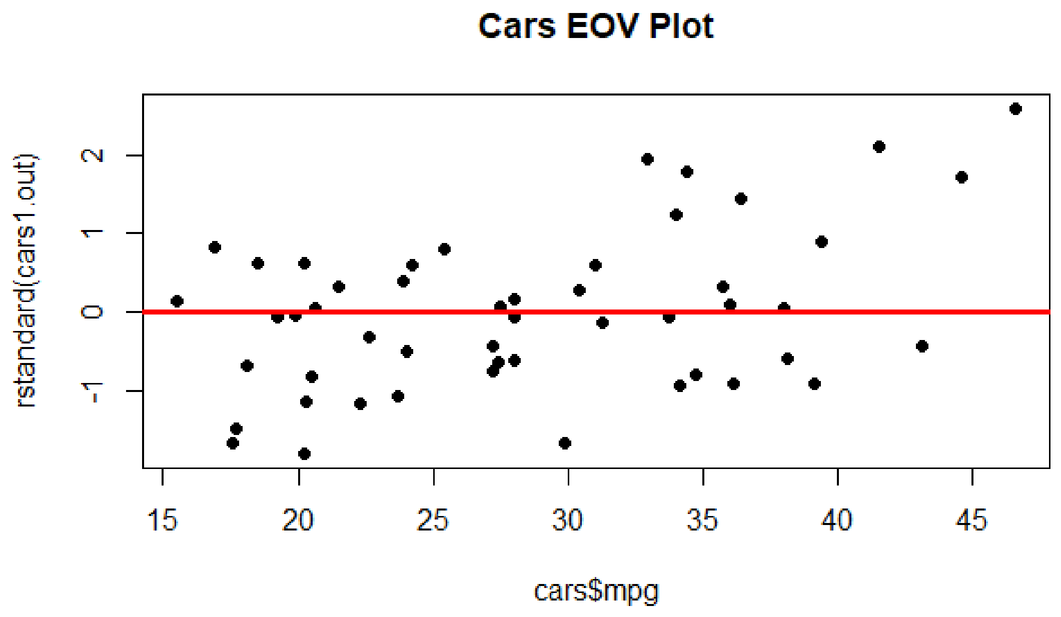
```r
# check normality assumption
qqnorm(cars1.out$residuals,pch=19,main="Cars Normality Plot")
qqline(cars1.out$residuals,lwd=3,col="red")

# optionally draw vertical lines to demarcate valid region(s)
abline(v=-1.25,lwd=3,col="blue")
abline(v=1.1,lwd=3,col="blue")
```



```
# calculate prob. dist. in the valid range
round(1-pnorm(-1.25)-pnorm(1.1,lower.tail=F),digits=3)

## [1] 0.759
```

- The normality assumption appears valid from -1.25 SD to 1.1 SD but is questionable outside that range.
- This range represents approximately 75.9% of the residuals.
- We could use ggplot2() to plot the CI. That would tell us how many points at the extremes are still acceptable (i.e. within the CI).

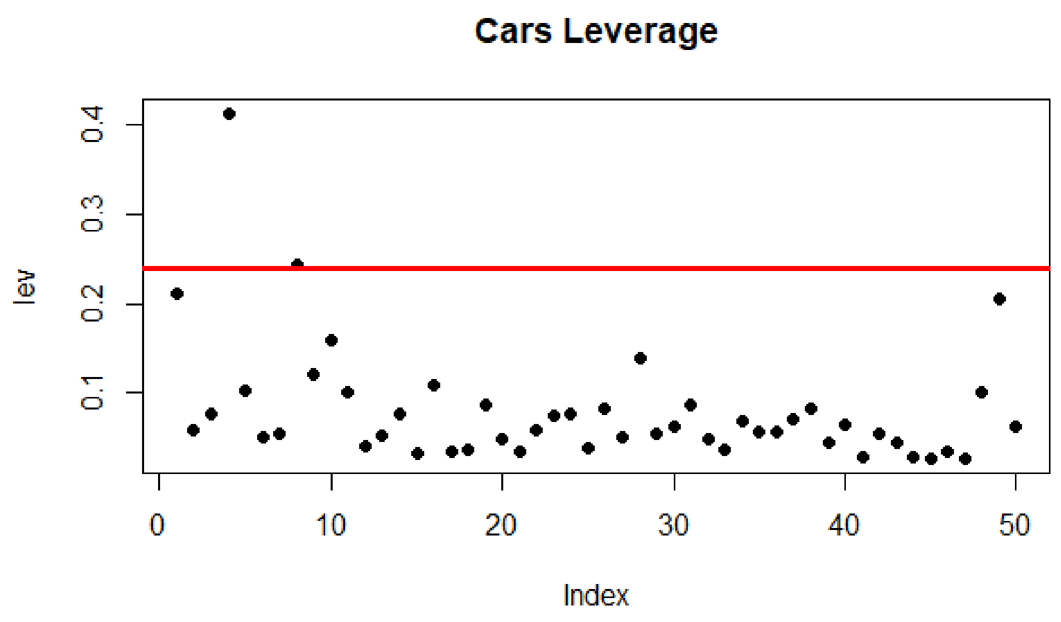
```
# check EOV assumption
plot(cars$mpg,rstandard(cars1.out),pch=19,main="Cars EOV Plot")
abline(0,0,col="red",lwd=3)
```



- The EOV assumption is questionable.
- After settling on a valid model, look for high influencers to improve model accuracy.

High Influencers - Manual Process

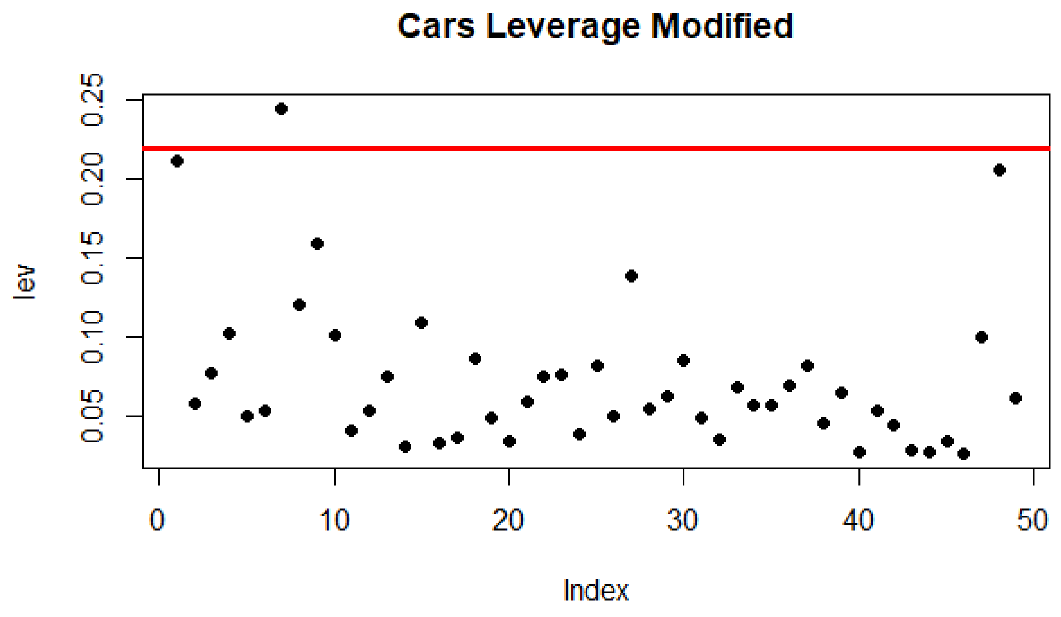
```
# create a vector of influence values
cars1.lev=hat(model.matrix(cars1.out))

# visually identify highest influencers
plot(cars1.lev,pch=19,main="Cars Leverage",ylab="lev")
abline(h=mean(cars1.lev)*3,lwd=3,col="red")
```



```
# store the index of the highest influencer
high.inf=which.max(cars1.lev)

# inspect the highest influencer
removed.df=cars[high.inf,]
removed.df[1,]
```

```
##    mpg horsepower weight
## 4 17.7        165   3445
```

```
# if appropriate, remove the highest influencer from the leverage vector
cars1.lev=cars1.lev[-high.inf]

# visually verify the removed observation was the highest influencer
plot(cars1.lev,pch=19,main="Cars Leverage Modified",ylab="lev")
abline(h=mean(cars1.lev)*3,lwd=3,col="red")
```



- The highest influencer was identified, inspected, and confirmed.
- We can now create a reduced data set and a new model.

Reduce Data Set and Create New Model

```
# copy the original data set
cars.red1=cars

# remove the highest influencer if appropriate, always justify removal
cars.red1=cars.red1[-high.inf,]
length(cars.red1$mpg)

## [1] 49

length(cars$mpg)

## [1] 50

# change names
detach(cars)
attach(cars.red1)

# create the new model based on the reduced data set
```

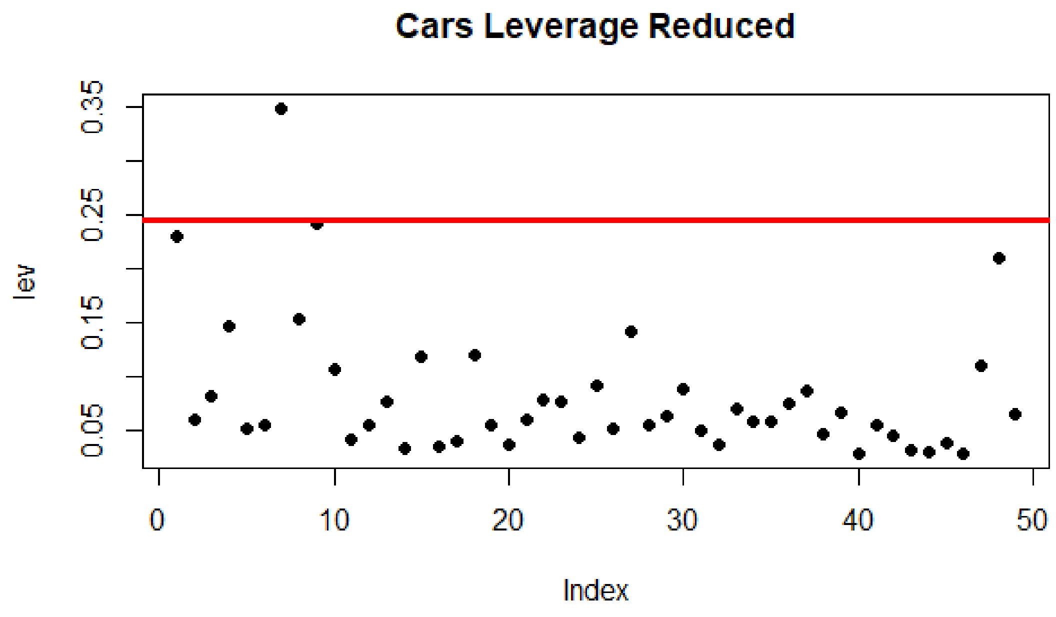
```r
cars.red1.out=lm(mpg~horsepower+weight+I(horsepower^2),data=cars.red1)
summary(cars.red1.out)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower + weight + I(horsepower^2), data =
cars.red1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0181 -2.5181 -0.0475  1.2335  8.8874
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     87.129846   6.916961  12.597 2.35e-16 ***
## horsepower      -0.732170   0.140817  -5.199 4.72e-06 ***
## weight          -0.006911   0.001223  -5.653 1.02e-06 ***
## I(horsepower^2)  0.003019   0.000687   4.395 6.69e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.552 on 45 degrees of freedom
## Multiple R-squared:  0.8199, Adjusted R-squared:  0.8079
## F-statistic: 68.29 on 3 and 45 DF,  p-value: < 2.2e-16
```

- We do not need to re-validate the model, so now we visually indentify high influencers in the new model.
- Repeat the removal procedure if needed and justified. Be very cautious about continually removing observations.

High Influencers Again - Manual Process

```r
# visually identify highest influencers
cars.red1.lev=hat(model.matrix(cars.red1.out))
plot(cars.red1.lev,pch=19,main="Cars Leverage Reduced",ylab="lev")
abline(h=mean(cars.red1.lev)*3,lwd=3,col="red")
```



- In the case, we choose not to remove any more influencers even though one exists above the red line.
- Alternatively, the next code chunk can be used to remove all high influencers iteratively and automatically. Be very cautious with such methods. Model results might not be reliable due to excessive skew from reality.
- Choose either the manual process or the automated process (not both).

High Influencers - Automated Process

```
# # copy the original data set
# cars.red=cars
#
# # change names
# detach(cars)
# attach(cars.red)
#
# # initialize needed objects
# cars5.out = NULL
# cars.prev.out = NULL
# prevAdjRSq = 0.0
#
# # document the removed observations
# count=0
# removed.df=data.frame(mpg=NA, horsepower=NA, weight=NA)
#
# # remove high influencers
# repeat
# {
#   cars5.out = lm(mpg~horsepower+weight+I(horsepower^2),data=cars.red)
#   adjustedRSq = summary(cars5.out)$adj.r.squared
#   cars5.lev = hat(model.matrix(cars5.out))
#   hilev = length(cars5.lev[cars5.lev>(3*mean(cars5.lev))])
#   if(prevAdjRSq > adjustedRSq)
#   {
#   print("Previous model had better coefficient of determination ...
retaining previous model.")
#   summary(cars.prev.out)
#   break;
#   }
#   else if(hilev == 0)
#   {
#     print("No more high leverage points.",quote=F)
#     break;
#   }
#   else
#   {
#   maxlev=which.max(cars5.lev)
#   count=count+1
#   removed.df[count,]=cars.red[maxlev,]
#   print(paste("There are ", hilev, " high leverage points. Removing highest
at index ", maxlev, ".", sep=""),quote=F)
#   cars.red = cars.red[-maxlev,]
#   cars.prev.out <- cars5.out
#   prevAdjRSq <- summary(cars.prev.out)$adj.r.squared
#   }
# }
#
# # display the number of removed observations
# print(paste(length(removed.df$mpg)," high leverage points were
removed.",sep=""),quote=F)
#
# # display the removed observations
# # row numbers do not match original row numbers
# removed.df
```

- After honing the model’s accuracy, we can make predictions.

Make Predictions

```
# determine a prediction value

# calculate the midpoint of the observed horsepower values
hpval=((range(cars.red1$horsepower)[2]-
range(cars.red1$horsepower)[1])/2)+range(cars.red1$horsepower)[1]

# calculate the midpoint of the observed weight values
wgtval=((range(cars.red1$weight)[2]-
range(cars.red1$weight)[1])/2)+range(cars.red1$weight)[1]

# store values in a data frame
predval=data.frame(horsepower=hpval,weight=wgtval,hpsq=hpval^2)

# make predictions
pred1=round(predict(cars.red1.out,predval,interval="prediction"),digits=1)
pred2=round(predict(cars.red1.out,predval,interval="confidence"),digits=1)
pred1

##    fit  lwr  upr
## 1 22.8 15.5 30.1

pred2

##    fit  lwr  upr
## 1 22.8 21.2 24.4
```

- Our model predicts a car weighing 3057.5 pounds with 101.5 horsepower will get 22.8 MPG. Our 95% CI is 15.5 to 30.1.
- Our model predicts cars weighing 3057.5 pounds with 101.5 horsepower will get 22.8 MPG on average. Our 95% CI is 21.2 to 24.4.