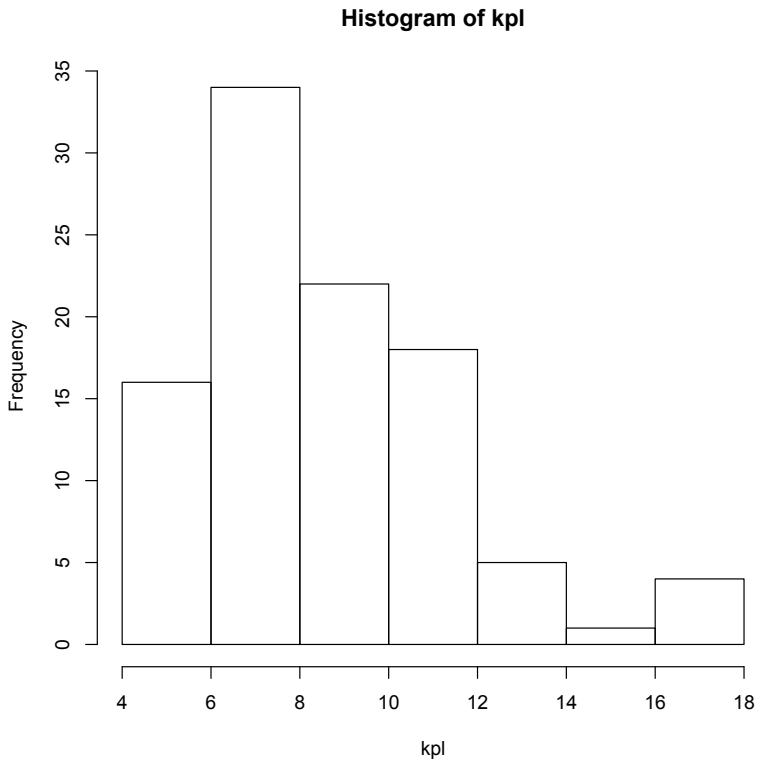
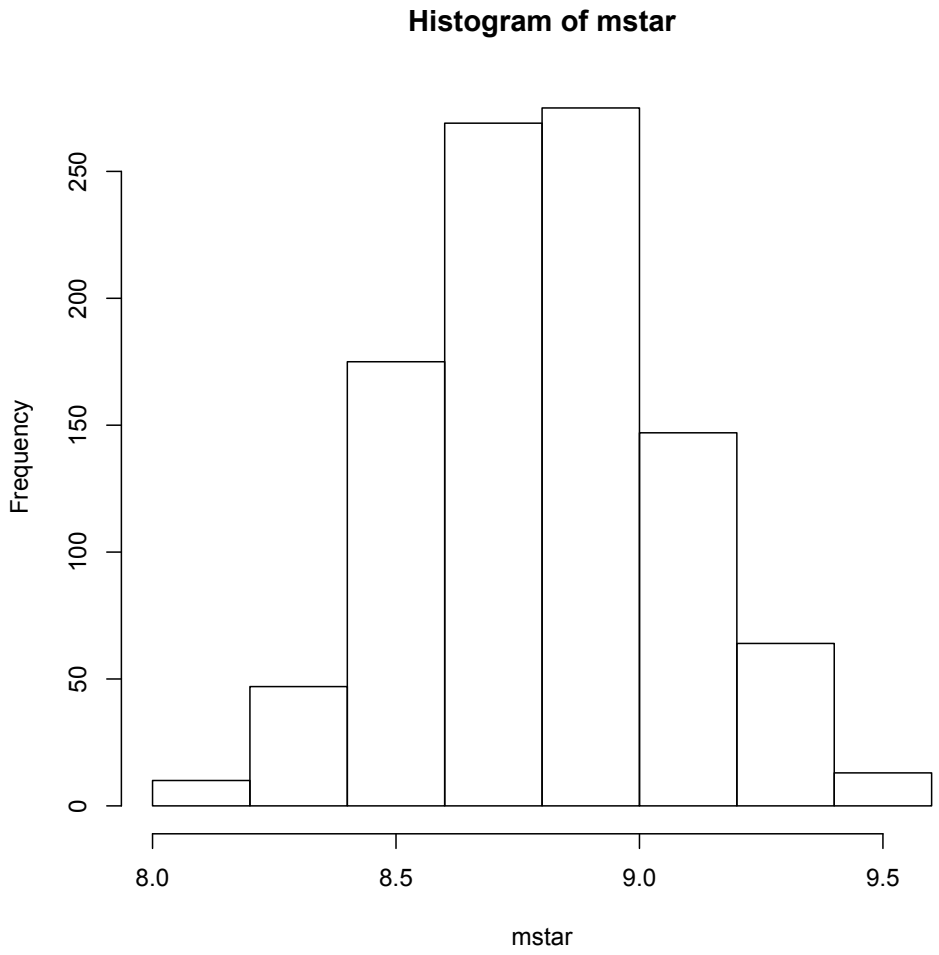
Bootstrap Regression with R

```
> # Bootstrap regression example
>
> kars =
read.table("http://www.utstat.toronto.edu/~brunner/appliedf12/data/mcars3.data",
+ header=T)
> kars[1:4,]
  Cntry   kpl weight length
1    US  5.04 2.1780 5.9182
2 Japan 10.08 1.0260 4.3180
3    US  9.24 1.1880 4.2672
4    US  7.98 1.4445 5.1054
> attach(kars) # Variables are now available by name
>
> # Before regression, a garden variety univariate bootstrap
> hist(kpl) # Right skewed
```



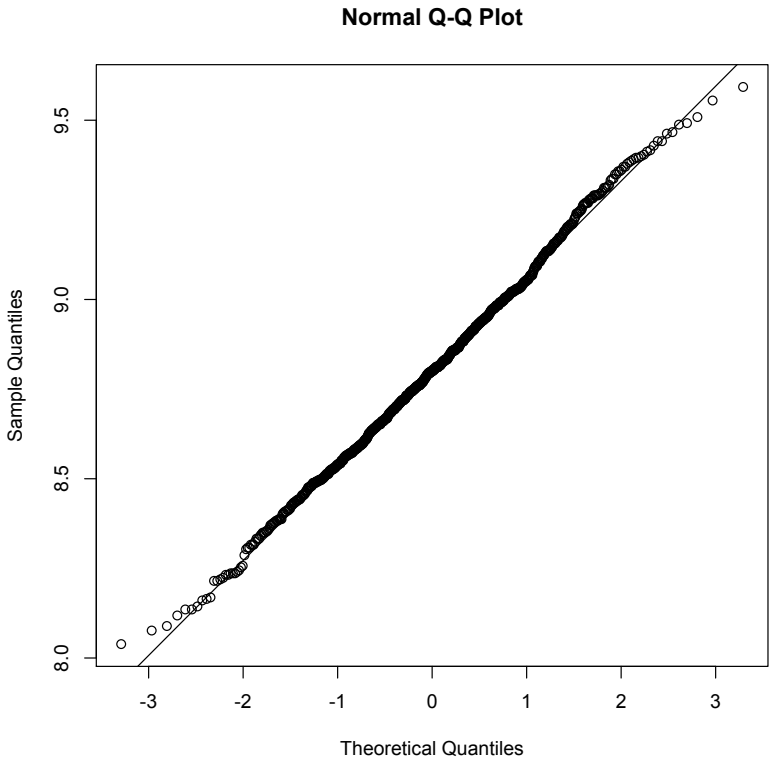

```
> # Small example for demonstration of R syntax
> set.seed(3244)
> x = kpl[1:10]; x
 [1]  5.04 10.08  9.24  7.98  7.98  7.98  9.66  7.56  5.88 10.92
> n = length(x)
> # Sample of size n from the numbers 1:n, with replacement.
> choices = sample(1:n,size=n,replace=T); choices
 [1]  2  7  5  1  4  6  9  8  4 10
> x[choices]
 [1] 10.08  9.66  7.98  5.04  7.98  7.98  5.88  7.56  7.98 10.92
```

```
> # Now bootstrap the mean of kpl
> n = length(kpl); B = 1000
> mstar = NULL # mstar will contain bootstrap mean values
>
> for(draw in 1:B) mstar = c(mstar,mean(kpl[sample(1:n,size=n,replace=T)]))
> hist(mstar)
```



```
> # Look at a normal qq plot. That's a plot of the order statistics against
> # the corresponding quantiles of the (standard) normal. Should be roughly linear
> # if the data are from a normal distribution.

> qqnorm(mstar); qqline(mstar)
```




```
> # Quantile bootstrap CI for mu. Use ONLY if the bootstrap distribution is symmetric.
> sort(mstar)[25]; sort(mstar)[975]
[1] 8.3034
[1] 9.3492
> # Compare the usual CI
> t.test(kpl)

      One Sample t-test

data:  kpl
t = 32.9363, df = 99, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 8.264966 9.324634
sample estimates:
mean of x
   8.7948
```

```
> # Now regression
> # Compute some polynomial terms
> wsq = weight^2; lsq = length^2; wl = weight*length
> # Bind it into a nice data frame
> datta = cbind(kpl,weight,length,wsq,lsq,wl)
> datta = as.data.frame(datta)
>
> model1 = lm(kpl ~ weight + length + wsq + lsq + wl, data=datta)
> summary(model1)

Call:
lm(formula = kpl ~ weight + length + wsq + lsq + wl, data = datta)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0861 -0.8702  0.0490  0.6898  4.4006

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   79.124     29.121   2.717  0.00784 **
weight        24.336     26.570   0.916  0.36204
length       -33.764     19.350  -1.745  0.08427 .
wsq           11.377      8.531   1.334  0.18556
lsq            5.140      3.410   1.507  0.13508
wl           -12.442     10.174  -1.223  0.22442
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.577 on 94 degrees of freedom
Multiple R-squared: 0.6689,   Adjusted R-squared: 0.6513
F-statistic: 37.98 on 5 and 94 DF,  p-value: < 2.2e-16

> betahat = coef(model1); betahat
(Intercept)      weight      length         wsq         lsq          wl
  79.124214   24.336110  -33.763782   11.376646    5.139649  -12.442449
>
> set.seed(3244)
> bstar = NULL # Rows of bstar will be bootstrap vectors of regression coefficients.
> n = length(kpl); B = 1000
> for(draw in 1:B)
+     {
+     # Randomly sample from the rows of kars, with replacement
+     Dstar = datta[sample(1:n,size=n,replace=T),]
+     model = lm(kpl ~ weight + length + wsq + lsq + wl, data=Dstar)
+     bstar = rbind( bstar,coef(model) )
+     } # Next draw
>
> bstar[1:5,]
     (Intercept)     weight      length       wsq       lsq         wl
[1,]    64.73852  15.322187  -25.549389 14.378022  4.437351 -12.779138
[2,]   270.35328 158.868074 -149.690584 26.298031 21.834487 -47.728186
[3,]    30.97446  -0.156246   -7.492504 10.815623  1.853871  -7.490135
[4,]   102.07061  48.638339  -50.062648 15.305469  7.844481 -19.669466
[5,]    74.31620  -4.632140  -23.726771 -1.497523  2.162598   1.070870
>
```

```
> Vb = var(bstar) # Approximate asymptotic covariance matrix of betahat
> Vb
            (Intercept)     weight     length        wsq        lsq         wl
(Intercept)   4009.5755  2805.7433 -2432.9272  403.57428  359.95762 -795.78318
weight        2805.7433  2337.0043 -1816.6783  434.68245  288.94772 -724.67663
length       -2432.9272 -1816.6783  1511.4557 -292.33275 -229.91599  534.81663
wsq            403.5743   434.6825  -292.3327  117.92217   52.99175 -158.04873
lsq            359.9576   288.9477  -229.9160   52.99175   36.19566  -89.15841
wl            -795.7832  -724.6766   534.8166 -158.04873  -89.15841  239.35661
>
> # Test individual coefficients.  H0: betaj=0
> se = sqrt(diag(Vb)); Z = betahat/se
> rbind(betahat,se,Z)
        (Intercept)     weight      length       wsq       lsq          wl
betahat   79.124214 24.3361095 -33.7637822 11.376646 5.1396488 -12.4424491
se        63.321209 48.3425725  38.8774443 10.859198 6.0162826  15.4711541
Z          1.249569  0.5034095  -0.8684671  1.047651 0.8542898  -0.8042354

> # Now test the product terms all at once
>
> WaldTest = function(L,thetahat,Vn,h=0) # H0: L theta = h
+ # Note Vn is the asymptotic covariance matrix, so it's the
+ # Consistent estimator divided by n. For true Wald tests
+ # based on numerical MLEs, just use the inverse of the Hessian.
+      {
+      WaldTest = numeric(3)
+      names(WaldTest) = c("W","df","p-value")
+      r = dim(L)[1]
+      W = t(L%*%thetahat-h) %*% solve(L%*%Vn%*%t(L)) %*%
+           (L%*%thetahat-h)
+      W = as.numeric(W)
+      pval = 1-pchisq(W,r)
+      WaldTest[1] = W; WaldTest[2] = r; WaldTest[3] = pval
+      WaldTest
+      } # End function WaldTest
>
> Lprod  = rbind( c(0,0,0,1,0,0),
+                 c(0,0,0,0,1,0),
+                 c(0,0,0,0,0,1) )
> WaldTest(Lprod,betahat,Vb)
       W       df  p-value
9.463393 3.000000 0.023724
>
>
> # Normal test for comparison
> model0 = lm(kpl ~ weight + length) # No product terms
> anova(model0,model1) # p = 0.0133
Analysis of Variance Table

Model 1: kpl ~ weight + length
Model 2: kpl ~ weight + length + wsq + lsq + wl
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     97 261.81
2     94 233.72  3    28.095 3.7666 0.0133 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Final comment: This is not a typical bootstrap regression. It's more common to bootstrap the residuals. But that applies to a conditional model in which the values of the explanatory variables are fixed constants.