Diagnostics Examples in R

```
> #lessR full Regression function has a number of diagnostics by default[1]
> #Simple regression output slightly different-includes a plot with conf bands
> #adding res_rows="all" will print the diagnostics for all cases, which I recommend, but omit to save space
> Regression(salary ~ pubs)


BASIC ANALYSIS

Estimated Model

             Estimate    Std Err  t-value  p-value   Lower 95%   Upper 95%
(Intercept) 46357.449   3072.730   15.087    0.000   39719.219   52995.679
       pubs   335.526    128.067    2.620    0.021      58.853     612.199


Model Fit

Standard deviation of residuals:  6623.620 for 13 degrees of freedom

R-squared:  0.346    Adjusted R-squared:  0.295    PRESS R-squared:  0.157

Null hypothesis that all population slope coefficients are 0:
  F-statistic: 6.864     df: 1 and 13     p-value:  0.021


Analysis of Variance

            df         Sum Sq        Mean Sq   F-value   p-value
Model        1  301137778.671  301137778.671     6.864     0.021
Residuals   13  570340400.929   43872338.533
salary      14  871478179.600   62248441.400

  RELATIONS AMONG THE VARIABLES

Correlation Matrix

         salary pubs
  salary   1.00 0.59
    pubs   0.59 1.00

  RESIDUALS AND INFLUENCE

Data, Fitted, Residual, Studentized Residual, Dffits, Cook's Distance
   [sorted by Cook's Distance]
   [res.rows = 15, out of 15 ]
-------------------------------------------------------------
       pubs    salary    fitted      resid rstdnt dffits cooks
   7 38.000 66432.000 59107.436   7324.564  1.255  0.605 0.175
  13 10.000 39115.000 49712.709 -10597.709 -1.838 -0.625 0.165
   2  3.000 54511.000 47364.027   7146.973  1.208  0.554 0.148
   3  2.000 53425.000 47028.501   6396.499  1.078  0.517 0.132
   9  9.000 41934.000 49377.183  -7443.183 -1.214 -0.430 0.089
   4 17.000 61863.000 52061.390   9801.610  1.629  0.447 0.088
  11 30.000 49832.000 56423.228  -6591.228 -1.056 -0.361 0.065
  12 21.000 47047.000 53403.494  -6356.494 -0.993 -0.266 0.035
  10 22.000 47454.000 53739.020  -6285.020 -0.982 -0.266 0.035
  15 37.000 61458.000 58771.910   2686.090  0.432  0.200 0.021
  14 27.000 59677.000 55416.650   4260.350  0.658  0.201 0.021
   8 48.000 61100.000 62462.696  -1362.696 -0.248 -0.186 0.019
   5 11.000 52926.000 50048.235   2877.765  0.443  0.145 0.011
   6  6.000 47034.000 48370.605  -1336.605 -0.209 -0.084 0.004
   1 18.000 51876.000 52396.916   -520.916 -0.078 -0.021 0.000

  FORECASTING ERROR

Data, Predicted, Standard Error of Forecast, 95% Prediction Intervals
```

[1] It is possible to obtain some casewise diagnostic values from the Regression function and then use them in further analyses or graphing, but it is more complicated than in SPSS. First name the results, e.g., model1.

```
model1 <- Regression(dmanx ~ perfect + sa, res_rows="all")
```

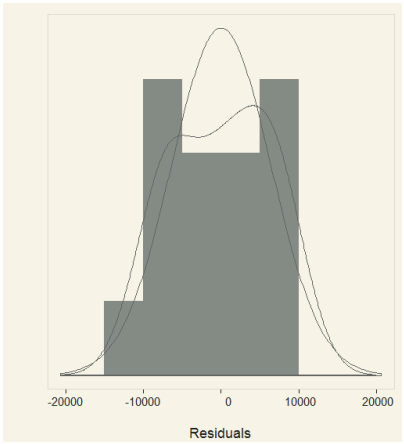
Then put the diagnostic value you want (here cooks.distance) into the data frame. Note that using View(model1) would show all of the casewise diagnostic names.

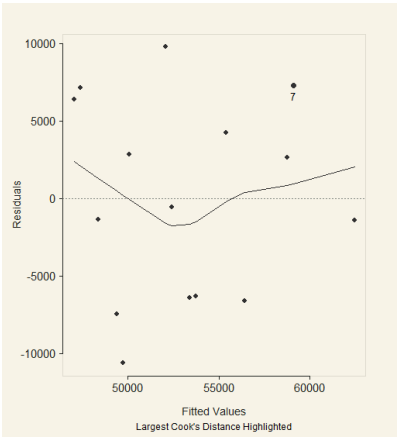
```
d$cooks <- model1$cooks.distance
```

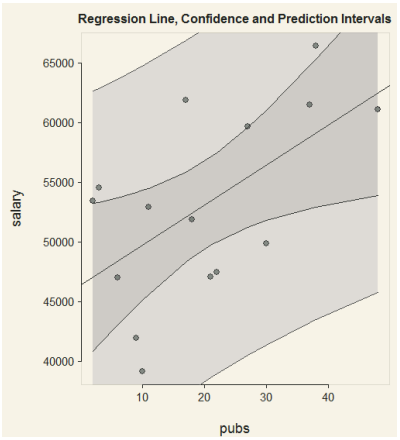
You could then select values based on a recommended cutoff or other desired value

```
library(dplyr)
d <-filter(d,cooks > 1)
```

```
   [sorted by lower bound of prediction interval]
-----------------------------------------------------------------------
       pubs    salary      pred       sf    pi:lwr    pi:upr     width
   3  2.000 53425.000 47028.501 7216.085 31439.097 62617.905 31178.809
   2  3.000 54511.000 47364.027 7176.352 31860.461 62867.593 31007.132
   6  6.000 47034.000 48370.605 7069.743 33097.353 63643.857 30546.503
   9  9.000 41934.000 49377.183 6982.674 34292.033 64462.332 30170.299
  13 10.000 39115.000 49712.709 6958.124 34680.595 64744.822 30064.227
   5 11.000 52926.000 50048.235 6935.853 35064.235 65032.234 29967.999
   4 17.000 61863.000 52061.390 6851.152 37260.376 66862.405 29602.028
   1 18.000 51876.000 52396.916 6845.324 37608.492 67185.341 29576.848
  12 21.000 47047.000 53403.494 6842.209 38621.801 68185.188 29563.387
  10 22.000 47454.000 53739.020 6845.963 38949.216 68528.825 29579.609
  14 27.000 59677.000 55416.650 6900.450 40509.135 70324.165 29815.030
  11 30.000 49832.000 56423.228 6961.266 41384.326 71462.130 30077.804
  15 37.000 61458.000 58771.910 7181.531 43257.156 74286.663 31029.507
   7 38.000 66432.000 59107.436 7221.538 43506.251 74708.620 31202.369
   8 48.000 61100.000 62462.696 7727.682 45768.053 79157.338 33389.285
-----------------------------------------------------------------------
Plot 1: Distribution of Residuals
Plot 2: Residuals vs Fitted Values
Plot 3: Regression Line, Confidence and Prediction Intervals
-----------------------------------------------------------------------
```



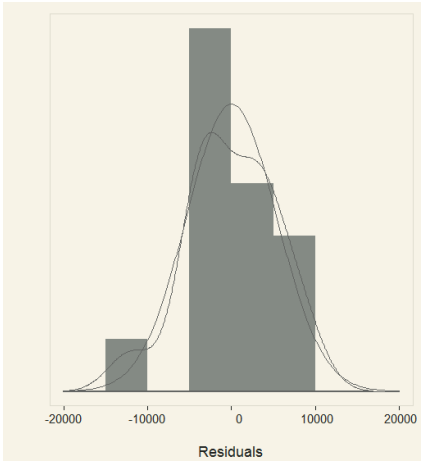


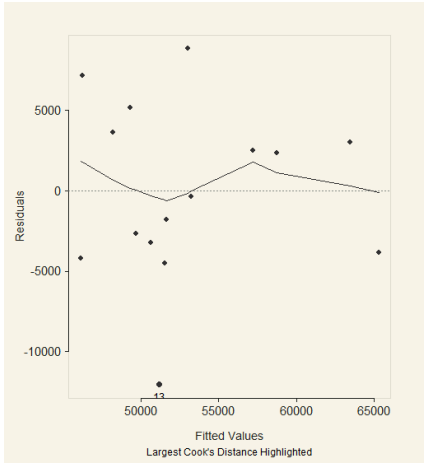


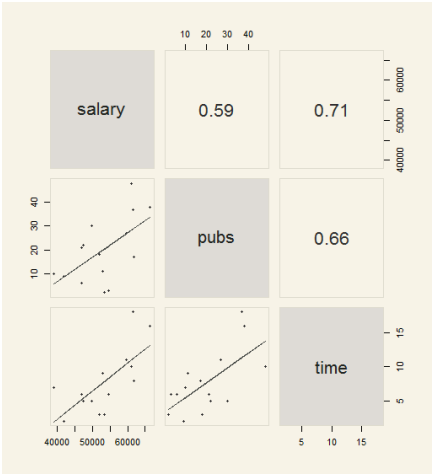

```
> #some diagnostics with car package
> library(car)
>
> model <- lm(salary ~ pubs + time, data = d)
> summary(model)

Call:
lm(formula = salary ~ pubs + time, data = d)

Residuals:
   Min     1Q Median     3Q    Max
-12066  -3522   -342   3324   8847

Coefficients:
            Estimate Std. Error t value      Pr(>|t|)
(Intercept)  43082.4     3099.5  13.900 0.00000000926
pubs           121.8      149.7   0.814        0.4317
time           982.9      452.1   2.174        0.0504

Residual standard error: 5839 on 12 degrees of freedom
Multiple R-squared:  0.5305,  Adjusted R-squared:  0.4522
F-statistic:  6.78 on 2 and 12 DF,  p-value: 0.01071
```

To remove cases in R

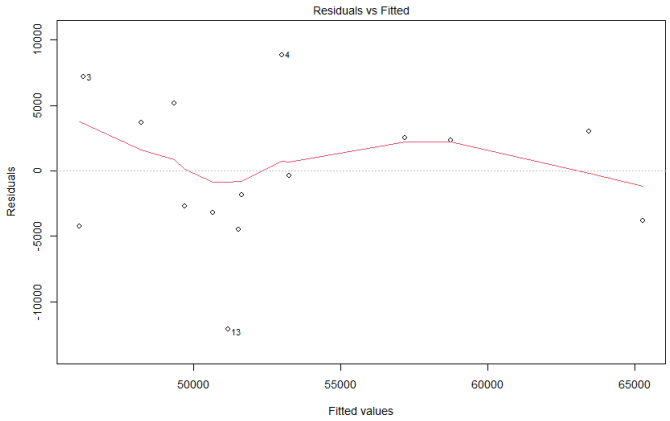
```
#selecting cases
#you may need to create id variable first
or skip if one already exists
id <- 1:nrow(d)
library(dplyr)
d <-filter(d,id != '1')
#removes the first case, but add
& id != '' to remove others
```



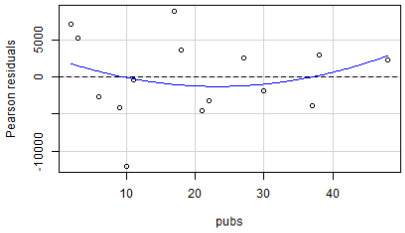


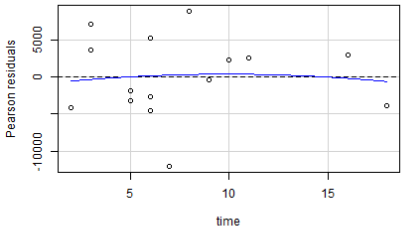


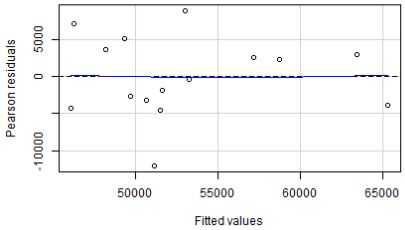

```
> #single residual plot by fitted values
> plot(model, which = 1)
```



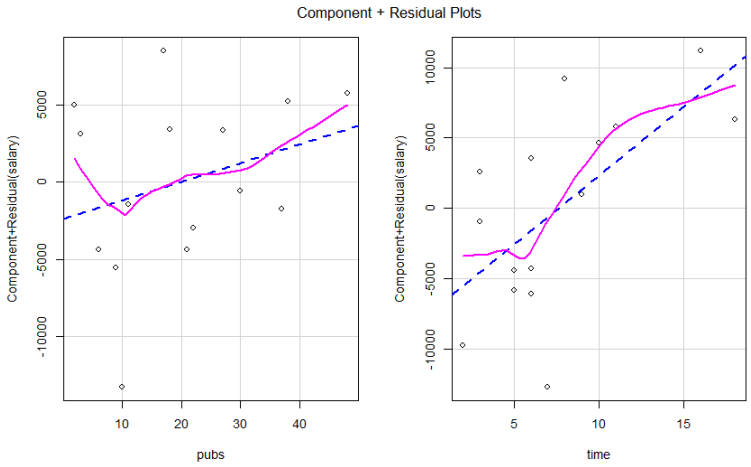

```
> #unadjusted residual plots by IV and predicted (fitted) values
> residualPlots(lm(salary ~ pubs + time, data = d))
           Test stat Pr(>|t|)
pubs           0.795    0.443
time          -0.208    0.839
Tukey test     0.069    0.945
```



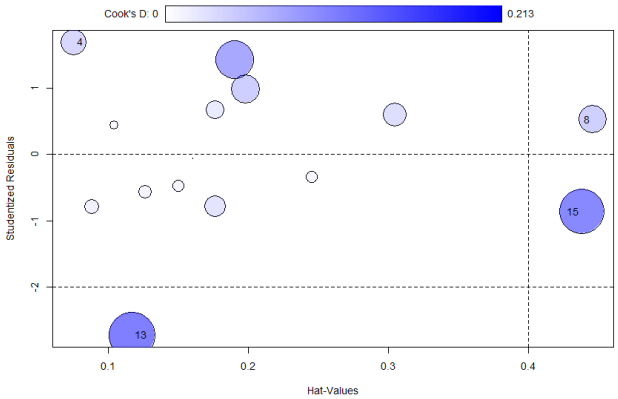





```
> #partial residual plots
> crPlots(lm(salary ~ pubs + time, data = d))
```




```
> #outlier test (may want to ignore significance tests and arbitrary definitions of outliers)
> outlierTest(model)
No Studentized residuals with Bonferonni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferonni p
13 -2.724396           0.019775      0.29663
> #diagnostic values and their plots, id.n= specifies number of cases desired
> influencePlot(lm(salary ~ pubs + time, data = d),id.n=15)
       StudRes        Hat        CookD
5  -0.06122461 0.16081925 0.0002611283
13 -2.72439562 0.11691982 0.2133750958
1   0.67327258 0.17612995 0.0338443252
6  -0.48107171 0.15038019 0.0145884979
14  0.43597027 0.10468261 0.0079439855
10 -0.57369635 0.12645875 0.0168226027
7   0.59862113 0.30497194 0.0553740752
11 -0.34525315 0.24586129 0.0139797912
9  -0.78142323 0.17619687 0.0449937170
12 -0.79300659 0.08869675 0.0210533507
15 -0.86429067 0.43838654 0.1985511620
2   0.98695808 0.19820783 0.0804403603
3   1.41695579 0.19042193 0.1452205801
4   1.69413249 0.07553642 0.0676304353
8   0.52255208 0.44632988 0.0781055999
```




```
> #obtain dfbetas (unstandardized)
> dfbetas(lm(salary ~ pubs + time, data = d),id.n=15)
    (Intercept)         pubs        time
1   0.231797720  0.139207147 -0.24384059
2   0.316875745 -0.385101202  0.17198426
3   0.639714806 -0.350417532 -0.09351616
4   0.229474144 -0.162360695  0.13247809
5  -0.008538538  0.019836564 -0.01694985
6  -0.140872717  0.142209195 -0.05507805
7  -0.210934645  0.028386377  0.24485383
8  -0.088903415  0.422006697 -0.20495510
9  -0.346822183 -0.006696766  0.21923153
10 -0.131247020 -0.115733245  0.14807598
11 -0.066378185 -0.156498763  0.14945109
12 -0.151232594 -0.093099374  0.12210115
13 -0.605522978  0.640035327 -0.33517230
14 -0.018987989  0.005396021  0.06408802
15  0.418823562  0.101232379 -0.59126091
```

```
> #multicollinearity diagnostic
> vif(model)
    pubs     time
1.758073 1.758073
```

```
> #test of autocorrelations or residuals (nonindepen
dence)
> durbinWatsonTest(model)
 lag Autocorrelation D-W Statistic p-value
   1       0.3622956      1.207088   0.054
 Alternative hypothesis: rho != 0
```

```
> #normal probability plot (car package)
> qqPlot(lm(salary ~ pubs + time, data = d))
```