Lecture 18: Diagnostics in multiple linear regression

Pratheepa Jeganathan

11/01/2019

Recap

- What is a regression model?
- Descriptive statistics – graphical
- Descriptive statistics – numerical
- Inference about a population mean
- Difference between two population means
- Some tips on R
- Simple linear regression (covariance, correlation, estimation, geometry of least squares)
  - Inference on simple linear regression model
  - Goodness of fit of regression: analysis of variance.
  - F-statistics.
  - Residuals.
  - Diagnostic plots for simple linear regression (graphical methods).

Recap

- Multiple linear regression
  - Specifying the model.
  - Fitting the model: least squares.
  - Interpretation of the coefficients.
  - Matrix formulation of multiple linear regression
  - Inference for multiple linear regression
    - T-statistics revisited.
    - More F statistics.
    - Tests involving more than one β.
- Diagnostics – more on graphical methods and numerical methods (**CH** Chapter 4.1-4.2, 4.4, 4.5, 4.6)
  - Different types of residuals (**CH** Chapter 4.3)
  - Diagnostics for assumptions on errors (**CH** Chapter 4.7)
  - Influence (**CH** Chapter 4.9, 4.10)

Outline

- Outlier detection (**CH** Chapter 4.8, 4.11, 4.14)
- Multiple comparison (Bonferroni correction)
- Residual plots: (**CH** Chapter 4.12, 4.13)
  - partial regression (added variable) plot,
  - partial residual (residual plus component) plot.

Data

```r
url = 'http://www.statsci.org/data/general/hills.txt'
races.table = read.table(url,
  header=TRUE, sep='\t')
head(races.table)
```

```
##             Race Distance Climb   Time
## 1 Greenmantle       2.5   650 16.083
## 2    Carnethy       6.0  2500 48.350
## 3 CraigDunain       6.0   900 33.650
## 4      BenRha       7.5   800 45.600
## 5   BenLomond       8.0  3070 62.267
## 6    Goatfell       8.0  2866 73.217
```

Diagnostics

Outliers

- The essential definition of an *outlier* is an observation pair $(Y, X_1, \ldots, X_p)$ that does not follow the model, while most other observations seem to follow the model.
- Outlier in *predictors*: the X values of the observation may lie outside the "cloud" of other X values.
  - This means you may be extrapolating your model inappropriately.
  - The values H_{ii} can be used to measure how "outlying" the X values are.
- Outlier in *response*: the Y value of the observation may lie very far from the fitted model.
  - If the studentized residuals are large: observation may be an outlier.

Outliers

- The races at `Bens of Jura` and `Lairig Ghru` seem to be outliers in *predictors* as they were the highest and longest races, respectively.
- How can we tell if the `Knock Hill` result is an outlier?
  - It seems to have taken much longer than it should have so maybe it is an outlier in the *response*.

Outlying X values

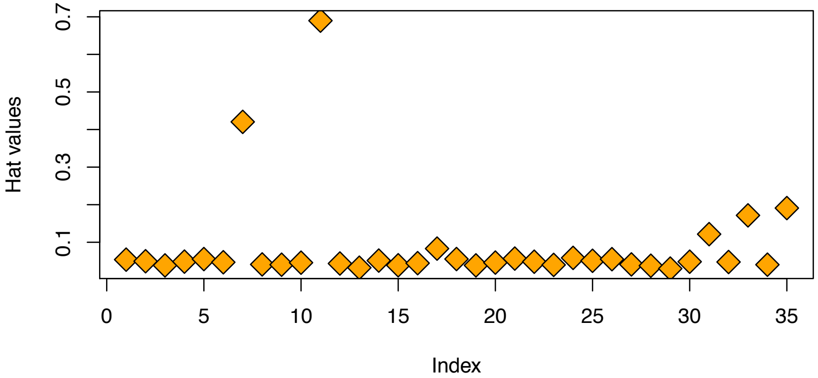
- One way to detect outliers in the *predictors*, besides just looking at the actual values themselves, is through their leverage values, defined by

$$\text{leverage}_i = H_{ii} = (X(X^T X)^{-1} X^T)_{ii}.$$

- Not surprisingly, our longest and highest courses show up again.
  - This at least reassures us that the leverage is capturing some of this "outlying in X space".

Outlying X values

```r
plot(hatvalues(races.lm), pch=23,
  bg='orange', cex=2, ylab='Hat values')
```



Outlying X values

```r
races.table[which(hatvalues(races.lm) > 0.3),]
```

```
##            Race Distance Climb    Time
## 7  BensofJura       16  7500 204.617
## 11 LairigGhru       28  2100 192.667
```

Outliers in the response

- We will consider a crude outlier test that tries to find residuals that are "larger" than they should be.
- Since `rstudent` are t distributed, we could just compare them to the T distribution and reject if their absolute value is too large.
- Doing this for every observation results in n different hypothesis tests.
- This causes a problem: if n is large, if we "threshold" at $t_{1-\alpha/2,n-p-2}$ we will get many outliers by chance even if model is correct.
- In fact, we expect to see $n \cdot \alpha$ "outliers" by this test. Every large data set would have outliers in it, even if model was entirely correct!

Outliers in the response

- Let's sample some data from our model to convince ourselves that this is a real problem.

```r
set.seed(1)
X = rnorm(100)
Y = 2 * X + 0.5 + rnorm(100)
alpha = 0.1
cutoff = qt(1 - alpha / 2, 97)
sum(abs(rstudent(lm(Y~X))) > cutoff)
```

```
## [1] 10
```

Outliers in the response

```r
# Bonferroni correction
# X = rnorm(100)
# Y = 2 * X + 0.5 + rnorm(100)
cutoff = qt(1 - (alpha / 100) / 2, 97)
sum(abs(rstudent(lm(Y~X))) > cutoff)
```

```
## [1] 0
```

Multiple comparisons

- This problem we identified is known as *multiple comparisons* or *simultaneous inference.*
- When performing many tests (say m) each at level α, we expect at least αm rejections even when *all* null hypotheses are true!
- In outlier detection, we are performing $m = n$ hypothesis tests, but might still like to control the probability of making *any* false positive errors.
- The reason we don't want to make errors here is that we don't want to throw away data unnecessarily.
- One solution: Bonferroni correction, threshold at $t_{1-\alpha/(2*n),n-p-2}$.

Bonferroni correction

- Dividing α by n, the number of tests, is known as a *Bonferroni* correction.
- If we are doing many t (or other) tests, say $m \gg 1$ we can control overall false positive rate at α by testing each one at level α/m.
- In this case $m = n$, but other times we might look at a different number of tests.

Bonferroni correction

- Essentially the *union bound* for probability.
- **Proof:** when the model is correct, with studentized residuals T_i:

$$
\begin{aligned}
P\,(\text{at least one false positive}) &= P\left(\cup_{i=1}^{m} |T_i| \geq t_{1-\alpha/(2*m),n-p-2}\right) \\
&\leq \sum_{i=1}^{m} P\left(|T_i| \geq t_{1-\alpha/(2*m),n-p-2}\right) \\
&= \sum_{i=1}^{m} \frac{\alpha}{m} = \alpha.
\end{aligned}
$$

- Let's apply this to our data. It turns out that `KnockHill` is a known error.

Example (Bonferroni correction)

```
n = nrow(races.table)
cutoff = qt(1 - 0.05 / (2*n),
  (n - 4))
races.table[which(abs(rstudent(races.lm)) > cutoff),]
```

```
##          Race Distance Climb  Time
## 18 KnockHill        3   350 78.65
```

Example (Bonferroni correction)

- The package car has a built in function to do this test.

```
library(car)
outlierTest(races.lm)
```

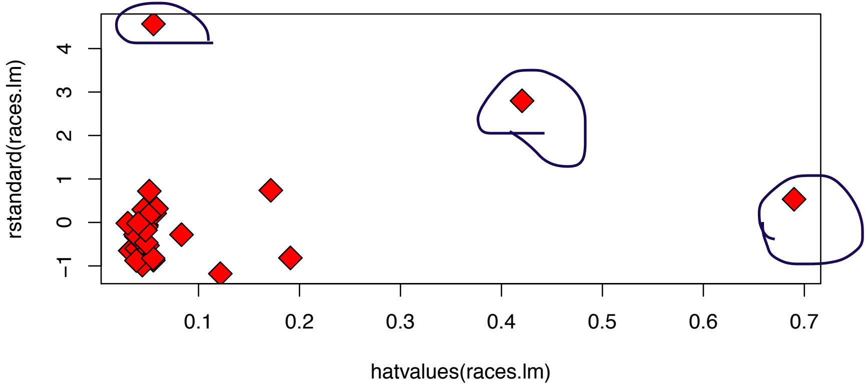
```
##    rstudent unadjusted p-value Bonferroni p
## 18 7.610845         1.3973e-08   4.8905e-07
```

Influential observation - leverage

- The last plot that `R` produces is a plot of residuals against leverage.
- Points that have high leverage and large residuals are particularly influential.

Example (leverage versus residuals)

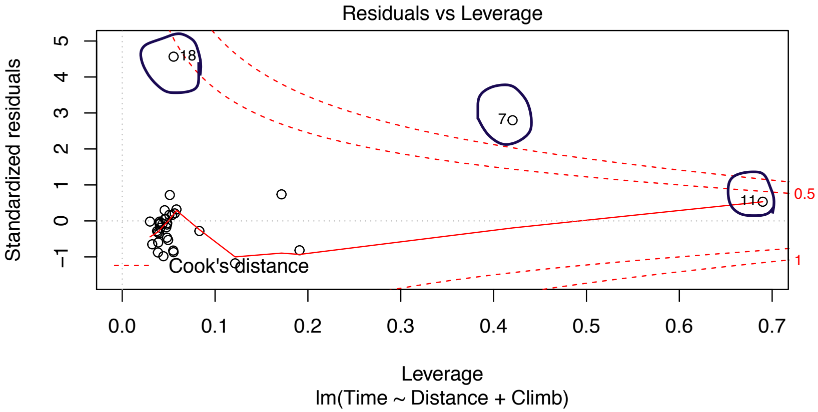
```
plot(hatvalues(races.lm), rstandard(races.lm),
  pch=23, bg='red', cex=2)
```



Example (leverage versus residuals)

- R will put the IDs of cases that seem to be influential in these (and other plots).
  - Not surprisingly, we see our usual three suspects.

```r
plot(races.lm, which=5)
```



Influence measures

- As mentioned above, `R` has its own rules for flagging points as being influential.
- To see a summary of these, one can use the `influence.measures` function.

Influence measures (in R)

```r
#influence.measures(races.lm)
knitr::include_graphics("Lecture_17_influence_measure.png")
```

	dfb.1_ <dbl>	dfb.Dstn <dbl>	dfb.Clmb <dbl>	dffit <dbl>	cov.r <dbl>	cook.d <dbl>	hat <dbl>	inf <fctr>
1	0.037811462	−0.0166142583	−0.0047435625	0.038617999	1.15946791	5.127185e-04	0.05375572	
2	−0.059579714	0.0672153961	−0.0733958853	−0.119560402	1.12694345	4.875401e-03	0.04946414	
3	−0.048576860	−0.0067065451	0.0280327646	−0.063095302	1.13289525	1.365422e-03	0.03840444	
4	−0.007664971	−0.0056751901	0.0087636598	−0.013674117	1.15557120	6.433010e-05	0.04848872	
5	−0.050460528	0.0847092735	−0.1450046113	−0.209472340	1.08370625	1.474139e-02	0.05527121	
6	0.003484456	−0.0043160647	0.0075759389	0.012209989	1.15360029	5.129264e-05	0.04680469	
7	−0.890654684	−0.7127735478	2.3646184862	2.699090776	0.81780209	1.893349e+00	0.42043463	*
8	−0.008442784	−0.0016484093	0.0055619075	−0.011150263	1.14667200	4.277564e-05	0.04103328	
9	−0.014368912	0.0009131396	0.0061606560	−0.016631781	1.14533663	9.515950e-05	0.04025783	
10	0.047034115	0.0130569237	−0.0365191836	0.063994414	1.14312971	1.405255e-03	0.04570891	
11	−0.301182091	0.7687159937	−0.4798493184	0.785688287	3.45248137	2.105214e-01	0.68981613	*
12	−0.011491649	0.0096557210	−0.0074877550	−0.016715572	1.14921244	9.612212e-05	0.04345357	
13	−0.031729063	−0.0299106792	−0.0007066754	−0.117700687	1.09223183	4.703839e-03	0.03231875	
14	0.118031242	0.0420335396	−0.1048840576	0.166101911	1.10391065	9.339448e-03	0.05126338	
15	−0.100376388	0.0577007540	−0.0223168727	−0.119202733	1.10615460	4.834282e-03	0.03877135	
16	−0.018520294	0.0067888268	−0.0998617172	−0.211352135	1.05013369	1.490749e-02	0.04436257	
17	0.011963729	−0.0665049703	0.0344553620	−0.083367689	1.19081472	2.385559e-03	0.08313942	
18	1.758274832	−0.4065452697	−0.6559341889	1.842374528	0.04932992	4.071560e-01	0.05535523	*
19	−0.158890179	0.0443113962	0.0294135680	−0.174838362	1.06346131	1.026539e-02	0.03850209	
20	0.008658369	0.0014243902	−0.0059464022	0.011018523	1.15257413	4.177135e-05	0.04590867	
21	0.047765462	−0.0100187391	−0.0191985978	0.050317950	1.16113850	8.700051e-04	0.05657466	
22	−0.018888912	0.0138562806	−0.0064653159	−0.022336402	1.15460132	1.716152e-04	0.04825780	
23	−0.041306482	0.0340969664	−0.0330224386	−0.069613005	1.13261824	1.661162e-03	0.03977381	
24	0.074833295	−0.0463850912	0.0064278105	0.078393718	1.15705550	2.107872e-03	0.05842537	
25	0.036911463	−0.0126332955	−0.0082568154	0.038084608	1.15566363	4.986386e-04	0.05072281	
26	−0.137724315	0.1361238983	−0.1013060816	−0.197816078	1.09137481	1.317865e-02	0.05499644	
27	−0.029204736	−0.0057020716	0.0192393928	−0.038570272	1.14314393	5.113116e-04	0.04103328	
28	−0.047641080	0.0069360885	0.0149895347	−0.054458683	1.13452136	1.017978e-03	0.03758135	
29	−0.002137967	0.0006466224	−0.0003281076	−0.003091995	1.13382999	3.289579e-06	0.02992818	
30	−0.085315881	−0.0077051500	0.0548379624	−0.103619059	1.13232031	3.669350e-03	0.04824732	
31	0.020993820	0.1701241625	−0.3736338993	−0.441381238	1.09600056	6.412250e-02	0.12158212	
32	−0.028579099	−0.0086935116	0.0232754469	−0.039310491	1.15127772	5.311898e-04	0.04746275	
33	−0.158227428	0.0970139844	0.1557016520	0.333844863	1.26094323	3.769491e-02	0.17158482	

1–33 of 35 rows

Previous 1 2 Next

Influence measures (in R)

- While not specified in the documentation, the meaning of the asterisks can be found by reading the code.
- The function `is.influential` makes the decisions to flag cases as influential or not.
- We see that the `DFBETAS` are thresholded at 1.
- We see that `DFFITS` is thresholded at $3 * sqrt((p+1)/(n-p-1))$.
- Etc.

influence.measures() code

influence.measures

```
function (model, infl = influence(model))
{
    is.influential <- function(infmat, n) {
        d <- dim(infmat)
        k <- d[[length(d)]] - 4L
        if (n <= k)
            stop("too few cases i with h_ii > 0), n < k")
        absmat <- abs(infmat)
        r <- if (is.matrix(infmat)) {
            cbind(absmat[, 1L:k] > 1, absmat[, k + 1] > 3 * sqrt(k/(n -
                k)), abs(1 - infmat[, k + 2]) > (3 * k)/(n -
                k), pf(infmat[, k + 3], k, n - k) > 0.5, infmat[,
                k + 4] > (3 * k)/n)
        }
        else {
            c(absmat[, , 1L:k] > 1, absmat[, , k + 1] > 3 * sqrt(k/(n -
                k)), abs(1 - infmat[, , k + 2]) > (3 * k)/(n -
                k), pf(infmat[, , k + 3], k, n - k) > 0.5, infmat[,
                , k + 4] > (3 * k)/n)
        }
        attributes(r) <- attributes(infmat)
        r
    }
    p <- model$rank
    e <- weighted.residuals(model)
    s <- sqrt(sum(e^2, na.rm = TRUE)/df.residual(model))
    mqr <- qr.lm(model)
    xxi <- chol2inv(mqr$qr, mqr$rank)
    si <- infl$sigma
    h <- infl$hat
    is.mlm <- is.matrix(e)
    cf <- if (is.mlm)
        aperm(infl$coefficients, c(1L, 3:2))
    else infl$coefficients
    dfbetas <- cf/outer(infl$sigma, sqrt(diag(xxi)))
    vn <- variable.names(model)
    vn[vn == "(Intercept)"] <- "1_"
    dimnames(dfbetas)[[length(dim(dfbetas))]] <- paste0("dfb.",
        abbreviate(vn))
    dffits <- e * sqrt(h)/(si * (1 - h))
    if (any(ii <- is.infinite(dffits)))
        dffits[ii] <- NaN
    cov.ratio <- (si/s)^(2 * p)/(1 - h)
    cooks.d <- if (inherits(model, "glm"))
        (infl$pear.res/(1 - h))^2 * h/(summary(model)$dispersion *
            p)
    else ((e/(s * (1 - h)))^2 * h)/p
    infmat <- if (is.mlm) {
        dns <- dimnames(dfbetas)
        dns[[3]] <- c(dns[[3]], "dffit", "cov.r", "cook.d", "hat")
        a <- array(dfbetas, dim = dim(dfbetas) + c(0, 0, 3 +
            1), dimnames = dns)
        a[, , "dffit"] <- dffits
        a[, , "cov.r"] <- cov.ratio
        a[, , "cook.d"] <- cooks.d
        a[, , "hat"] <- h
        a
    }
    else {
        cbind(dfbetas, dffit = dffits, cov.r = cov.ratio, cook.d = cooks.d,
            hat = h)
    }
    infmat[is.infinite(infmat)] <- NaN
    is.inf <- is.influential(infmat, sum(h > 0))
    ans <- list(infmat = infmat, is.inf = is.inf, call = model$call)
    class(ans) <- "infl"
    ans
}
```

Problems in the regression function

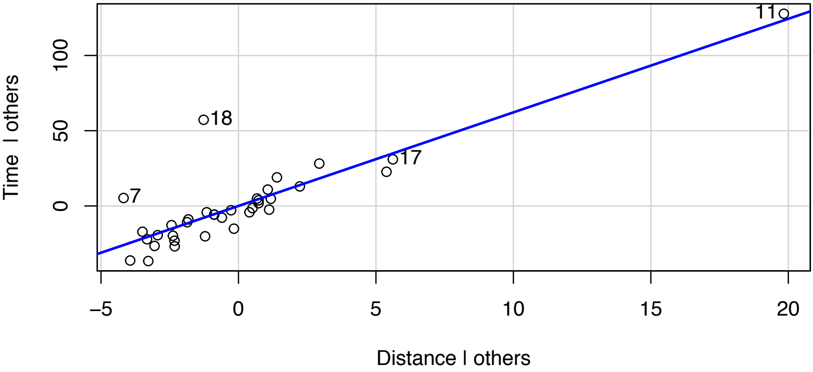
- True regression function may have higher-order non-linear terms, polynomial or otherwise.
- We may be missing terms involving more than one $X_{(\cdot)}$, i.e. $X_i \cdot X_j$ (called an *interaction*).
- Some simple plots: *added-variable* and *component plus residual* plots can help to find nonlinear functions of *one variable*.
- We will find these plots of somewhat limited use in practice, but we will go over them as possibly useful diagnostic tools.

Added variable plots

- Enable to see the magnitude f the regression coefficient of the new variable that is being considered for inclusion.
- Can also identify influential observations.
- The functions can be found in the `car` package.
- Procedure:
  - Let $\tilde{e}_{X_j,i}, 1 \le i \le n$ be the residuals after regressing X_j onto all columns of X except X_j;
  - Let $e_{X_j,i}$ be the residuals after regressing Y onto all columns of X except X_j;
  - Plot $\tilde{e}_{X_j}$ against e_{X_j}.
  - If the (partial regression) relationship is linear this plot should look linear.

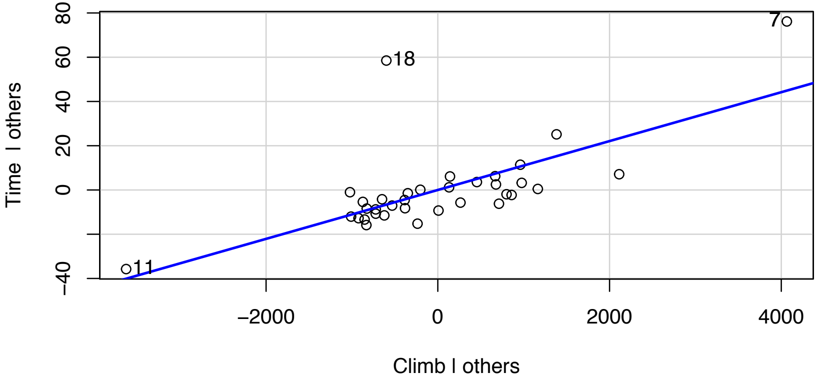
Example (Added variable plots)

```r
avPlots(races.lm, 'Distance')
```



Example (Added variable plots)

```r
avPlots(races.lm, 'Climb')
```

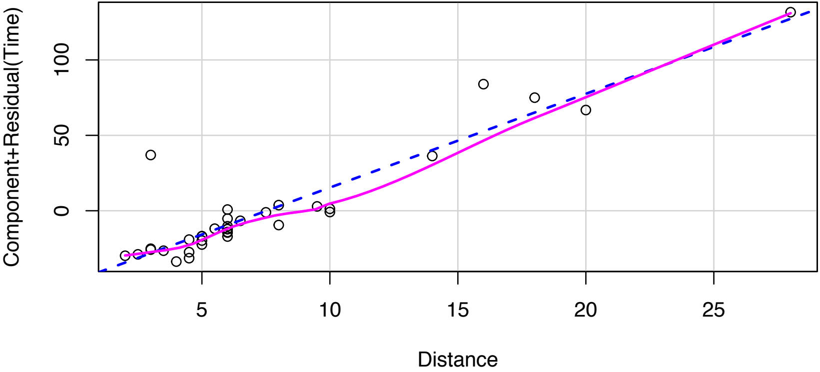


Residual + component plots

- Similar to added variable, but may be more helpful in identifying nonlinear relationships (horizontal axis is variable itself).
- Procedure: plot $X_{ij}, 1 \leq i \leq n$ vs. $e_i + \widehat{\beta}_j \cdot X_{ij}, 1 \leq i \leq n$.
- The violet line is a non-parametric smooth of the scatter plot that may suggest relationships other than linear.

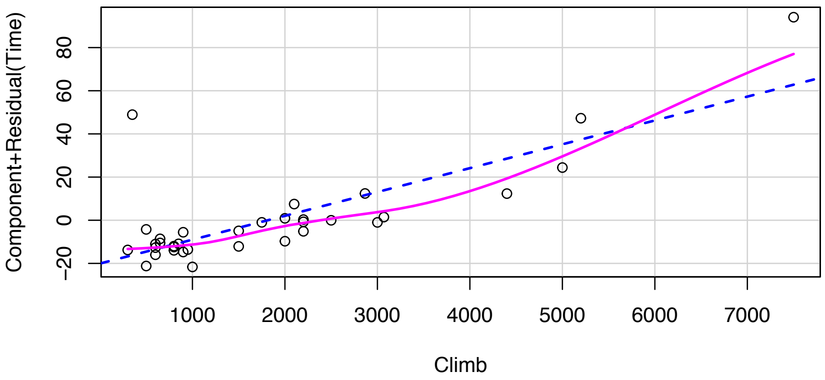
Example (Residual + component plots)

```r
crPlots(races.lm, 'Distance')
```



Example (Residual + component plots)

```r
crPlots(races.lm, 'Climb')
```



Reference

- **CH**: Chapter 4.
- Lecture notes of Jonathan Taylor .