Lecture 20: Outliers and Influential Points

An **outlier** is a point with a large residual. An **influential point** is a point that has a large impact on the regression. Surprisingly, these are not the same thing. A point can be an outlier without being influential. A point can be influential without being an outlier. A point can be both or neither.

Figure 1 shows four famous datasets due to Frank Anscombe. If you run least squares on each dataset you will get the same output:

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0001     1.1247   2.667  0.02573 *
x             0.5001     0.1179   4.241  0.00217 **



Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared:  0.6665,Adjusted R-squared:  0.6295
F-statistic: 17.99 on 1 and 9 DF,  p-value: 0.00217
```

The top left plot has no problems. The top right plot shows a non-linear pattern. The bottom left plot has an an outlier. The bottom right plot has an influential point. Imagine what would happen if we deleted the rightmost point. If you looked at residual plots, you would see problems in the second and third case. But the resdual plot for the fourth example would look fine. You can't see influence in the usual residual plot.

1 Modified Residuals

Let $\mathbf{e}$ be the vector of residuals. Recall that

$$\mathbf{e} = (\mathbf{I} - \mathbf{H})\epsilon, \quad \mathbf{E}[\mathbf{e}] = \mathbf{0}, \quad \text{Var}(\mathbf{e}) = \sigma^2(\mathbf{I} - \mathbf{H}).$$

Thus the standard error of e_i is $\widehat{\sigma}\sqrt{1 - h_{ii}}$ where $h_{ii} \equiv \mathbf{H}_{ii}$. We then call

$$r_i = \frac{e_i}{\widehat{\sigma}\sqrt{1 - h_{ii}}}$$

the **standardized residual**.

There is another type of residual t_i which goes under various names: the **jackknife residual**, **the cross-validated residual**, **externally studentized residual** or **studentized deleted residual**. Let $\widehat{Y}_{i(-i)}$ is the predicted value for the i^{th} data point when (X_i, Y_i) is omitted from the data. Then t_i is defined by

$$t_i = \frac{Y_i - \widehat{Y}_{i(-i)}}{s_i} \tag{1}$$

where s_i^2 is the estimated variance of $Y_i - \widehat{Y}_{i(-i)}$. It can be shown that

$$t_i = r_i\sqrt{\frac{n - p - 2}{n - p - 1 - r_i^2}} = \frac{e_i}{\widehat{\sigma}_{(-i)}\sqrt{1 - h_{ii}}} \tag{2}$$

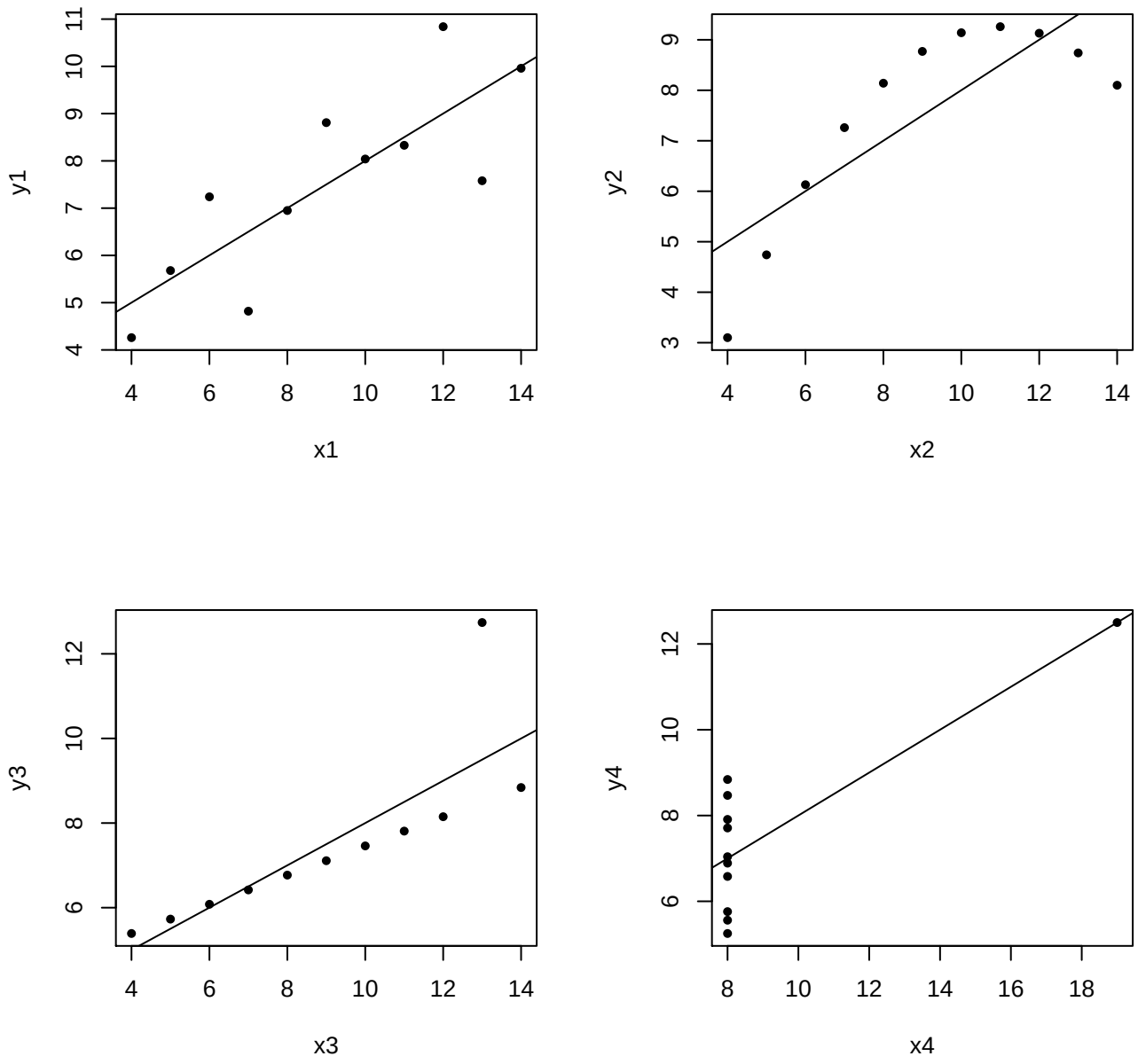

FIGURE 1: *For data sets that have the same fitted line. Top left: no problems. Top right: a non-linear pattern. Bottom left: An outlier. Bottom right: an influential point.*

$\widehat{\sigma}^2_{(-i)}$ is the estimated variance after omitting (X_i, Y_i) is omitted from the data. The cool think is that we can compute t_i without ever having to actually delete the observation and re-fit the model.

Everything you have done so far with residuals can also be done with standardized or jackknife residuals.

2 Influence

Recall that

$$\widehat{\mathbf{Y}} = \mathbf{HY}$$

where $\mathbf{H}$ is the hat matrix. This means that each $\widehat{Y}_i$ is a linear combination of elements of $\mathbf{H}$. In particular, $\mathbf{H}_{ii}$ is the contribution of the i^{th} data point to $\widehat{Y}_i$. For this reason we call $h_{ii} \equiv \mathbf{H}_{ii}$ the *leverage*.

To get a better idea of how influential the i^{th} data point is, we could ask: how much do the fitted values change if we omit an observation? Let $\mathbf{Y}^{(-i)}$ be the vector of fitted values when we remove observation i. Then **Cook's distance** is defined by

$$D_i = \frac{(\mathbf{Y} - \mathbf{Y}^{(-i)})^T(\mathbf{Y} - \mathbf{Y}^{(-i)})}{(p+1)\widehat{\sigma}^2}.$$

It turns out that there is a handy formula for computing D_i, namely:

$$D_i = \left(\frac{r_i^2}{p+1}\right)\left(\frac{h_{ii}}{1-h_{ii}}\right).$$

This means that the influence of a point is determined by both its residual and its leverage. Often, people interpret $D_i > 1$ as an influential point.

The leave-one-out idea can also be applied to the coefficients. Write $\widehat{\beta}^{(-i)}$ for the vector of coefficients we get when we drop the i^{th} data point. One can show that

$$\widehat{\beta}^{(-i)} = \widehat{\beta} - \frac{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}_i^T e_i}{1-h_{ii}}. \tag{3}$$

Cook's distance can actually be computed from this, since the change in the vector of fitted values is $\mathbf{x}(\widehat{\beta}^{(-i)} - \widehat{\beta})$, so

$$D_i = \frac{(\widehat{\beta}^{(-i)} - \widehat{\beta})^T\mathbf{X}^T\mathbf{X}(\widehat{\beta}^{(-i)} - \widehat{\beta})}{(p+1)\widehat{\sigma}^2}. \tag{4}$$

Sometimes, whole clusters of nearby points might be potential outliers. In such cases, removing just one of them might change the model very little, while removing them all might change it a great deal. Unfortunately there are $\binom{n}{k} = O(n^k)$ groups of k points you could consider deleting at once, so while looking at all leave-one-out results is feasible, looking at all leave-two- or leave-ten-out results is not.

3 Diagnostics in Practice

We have three ways of looking at whether points are outliers:

1. We can look at their leverage, which depends only on the value of the predictors.

2. We can look at their studentized residuals, either ordinary or cross-validated, which depend on how far they are from the regression line.

3. We can look at their Cook's statistics, which say how much removing each point shifts all the fitted values; it depends on the product of leverage and residuals.

The model assumptions don't put any limit on how big the leverage can get (just that it's ≤ 1 at each point) or on how its distributed across the points (just that it's got to add up to $p+1$). Having most of the leverage in a few super-inferential points doesn't break the model, exactly, but it should make us worry.

The model assumptions *do* say how the studentized residuals should be distributed. In particular, the cross-validated studentized residuals should follow a t distribution. This is something we can test, either for specific points which we're worried about (say because they showed up on our diagnostic plots), or across all the points.

3.1 In R

Almost everything we've talked — leverages, studentized residuals, Cook's statistics — can be calculated using the `influence` function. However, there are more user-friendly functions which call that in turn, and are probably better to use. Leverages come from the 'hatvalues' function, or from the 'hat' component of what 'influence' returns:

```
out = lm(Mobility ~ Commute,data=mobility)
hatvalues(out)
influence(out)$hat  ### this is the same as the previous line
rstandard(out)      ### standardized residuals
rstudent(out)       ### jackknife residuals
cooks.distance(out) ### Cook's distance
```

Often the most useful thing to do with these is to plot them, and look at the most extreme points. The standardized and studentized residuals can also be put into our usual diagnostic plots, since they should average to zero and have constant variance when plotted against the fitted values or the predictors.

```
par(mfrow=c(2,2))
n = nrow(mobility)
out = lm(Mobility ~ Commute,data=mobility)
plot(hatvalue(out),ylab="Leverage")
plot(rstandard(out),ylab="Standardized Residuals")
plot(rstudent(out),ylab="Cross-Validated Residuals")
abline(h=qt(0.025,df=n-2,col="red")
abline(h=qt(1-0.025,df=n-2,col="red")
plot(cooks.distance(out),ylab="Cook's Distance")
```

We can now look at exactly which points have the extreme values, say the 10 most extreme residuals, or largest Cook's statistics:

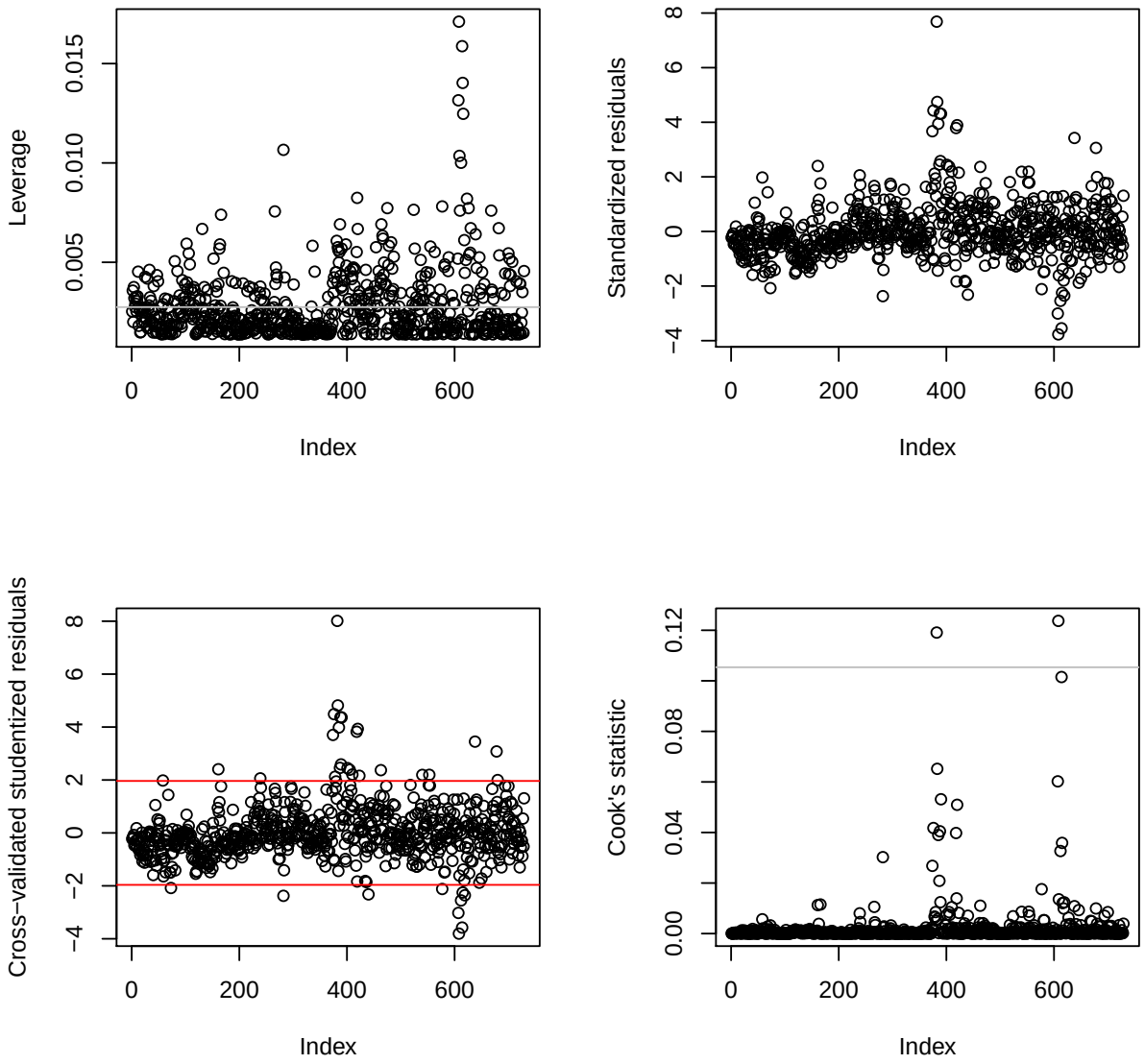

FIGURE 2: *Leverages, two sorts of standardized residuals, and Cook's distance statistic for each point in a basic linear model of economic mobility as a function of the fraction of workers with short commutes. The horizontal line in the plot of leverages shows the average leverage. The lines in studentized residual plot shows a 95% t-distribution sampling interval. Note the clustering of extreme residuals* and *leverage around row 600, and another cluster of points with extreme residuals around row 400.*

```
n = nrow(mobility)
out = lm(Mobility ~ Commute,data=mobility)
r = rstudent(out)
I = (1:n)[rank(-abs(r) <= 10)]   ## indices 10 largest residuals
mobility[I,]
```

```
##       X       Name   Mobility State Commute  Longitude Latitude
## 374 375     Linton 0.29891303    ND   0.646 -100.16075 46.31258
## 376 378 Carrington 0.33333334    ND   0.656  -98.86684 47.59698
## 382 384     Bowman 0.46969697    ND   0.648 -103.42526 46.33993
## 383 385     Lemmon 0.35714287    ND   0.704 -102.42011 45.96558
## 385 388 Plentywood 0.31818181    MT   0.681 -104.65381 48.64743
## 388 391  Dickinson 0.32920793    ND   0.659 -102.61354 47.32696
## 390 393  Williston 0.33830845    ND   0.702 -103.33987 48.25441
## 418 422     Miller 0.31506848    SD   0.697  -99.27758 44.53313
## 420 424 Gettysburg 0.32653061    SD   0.729 -100.19547 45.05100
## 608 618       Nome 0.04678363    AK   0.928 -162.03012 64.47514
```

```
C = cooks.distance(out)
I = (1:n)[rank(-abs(C) <= 10)]   ## indices 10 largest Cook's distances
mobility[I,]
```

```
##       X       Name   Mobility State Commute  Longitude Latitude
## 376 378 Carrington 0.33333334    ND   0.656  -98.86684 47.59698
## 382 384     Bowman 0.46969697    ND   0.648 -103.42526 46.33993
## 383 385     Lemmon 0.35714287    ND   0.704 -102.42011 45.96558
## 388 391  Dickinson 0.32920793    ND   0.659 -102.61354 47.32696
## 390 393  Williston 0.33830845    ND   0.702 -103.33987 48.25441
## 418 422     Miller 0.31506848    SD   0.697  -99.27758 44.53313
## 420 424 Gettysburg 0.32653061    SD   0.729 -100.19547 45.05100
## 607 617   Kotzebue 0.06451613    AK   0.864 -159.43781 67.02818
## 608 618       Nome 0.04678363    AK   0.928 -162.03012 64.47514
## 614 624     Bethel 0.05186386    AK   0.909 -158.38213 61.37712
```

3.2 plot

We have not used the `plot` function on an `lm` object yet. This is because most of what it gives us is in fact related to residuals (Figure 3).

```
par(mfrow=c(2,2))
plot(out)
```

The first plot is of residuals versus fitted values, plus a smoothing line, with extreme residuals marked by row number. The second is a Q-Q plot of the standardized residuals, again with extremes marked by row number. The third shows the square root of the absolute standardized

residuals against fitted values (ideally, flat); the fourth plots standardized residuals against leverage, with contour lines showing equal values of Cook's distance. There are many options, described in `help(plot.lm)`.

4 Dealing With Outliers

There are essentially three things to do when we're convinced there are outliers: delete them; change the model; or change how we estimate.

4.1 Deletion

Deleting data points should never be done lightly, but it is sometimes the right thing to do.

The best case for removing a data point is when you have good reasons to think it's just wrong (and you have no way to fix it). Medical records which give a patient's blood pressure as 0, or their temperature as 200 degrees, are just impossible and have to be errors[1]. Those points aren't giving you useful information about the process you're studying so getting rid of them makes sense.

The next best case is if you have good reasons to think that the data point isn't *wrong*, exactly, but belongs to a different phenomenon or population from the one you're studying. (You're trying to see if a new drug helps cancer patients, but you discover the hospital has included some burn patients and influenza cases as well.) Or the data point does belong to the right population, but also somehow to another one which isn't what you're interested in right now. (All of the data is on cancer patients, but some of them were also sick with the flu.) You should be careful about that last, though. (After all, some proportion of future cancer patients are also going to have the flu.)

The next best scenario after that is that there's nothing quite so definitely wrong about the data point, but it just looks really weird compared to all the others. Here you are really making a judgment call that either the data really are mistaken, or not from the right population, but you can't put your finger on a concrete reason why. The rules-of-thumb used to identify outliers, like "Cook's distance shouldn't be too big", or "Tukey's rule" which flags any point more than 1.5 times the inter-quartile range above the third quartile, or below the first quartile. It is always more satisfying, and more reliable, if investigating how the data were gathered lets you turn cases of this sort into one of the two previous kinds.

The least good case for getting rid of data points which isn't just bogus is that you've got a model which almost works, and would work a lot better if you just get rid of a few stubborn points. This is really a sub-case of the previous one, with added special pleading on behalf of your favorite model. You are here basically trusting your model more than your data, so it had better be either a really good model or really bad data.

4.2 Changing the Model

Outliers are points that break a pattern. This can be because the points are bad, or because we made a bad guess about the pattern. Figure 4 shows data where the cloud of points on the right are definite outliers for any linear model. But I drew those points following a quadratic model, and they fall perfectly along it (as they should). Deleting them, in order to make a linear model work better, would have been short-sighted at best.

[1] This is true whether the temperature is in degrees Fahrenheit, degrees centigrade, or kelvins.

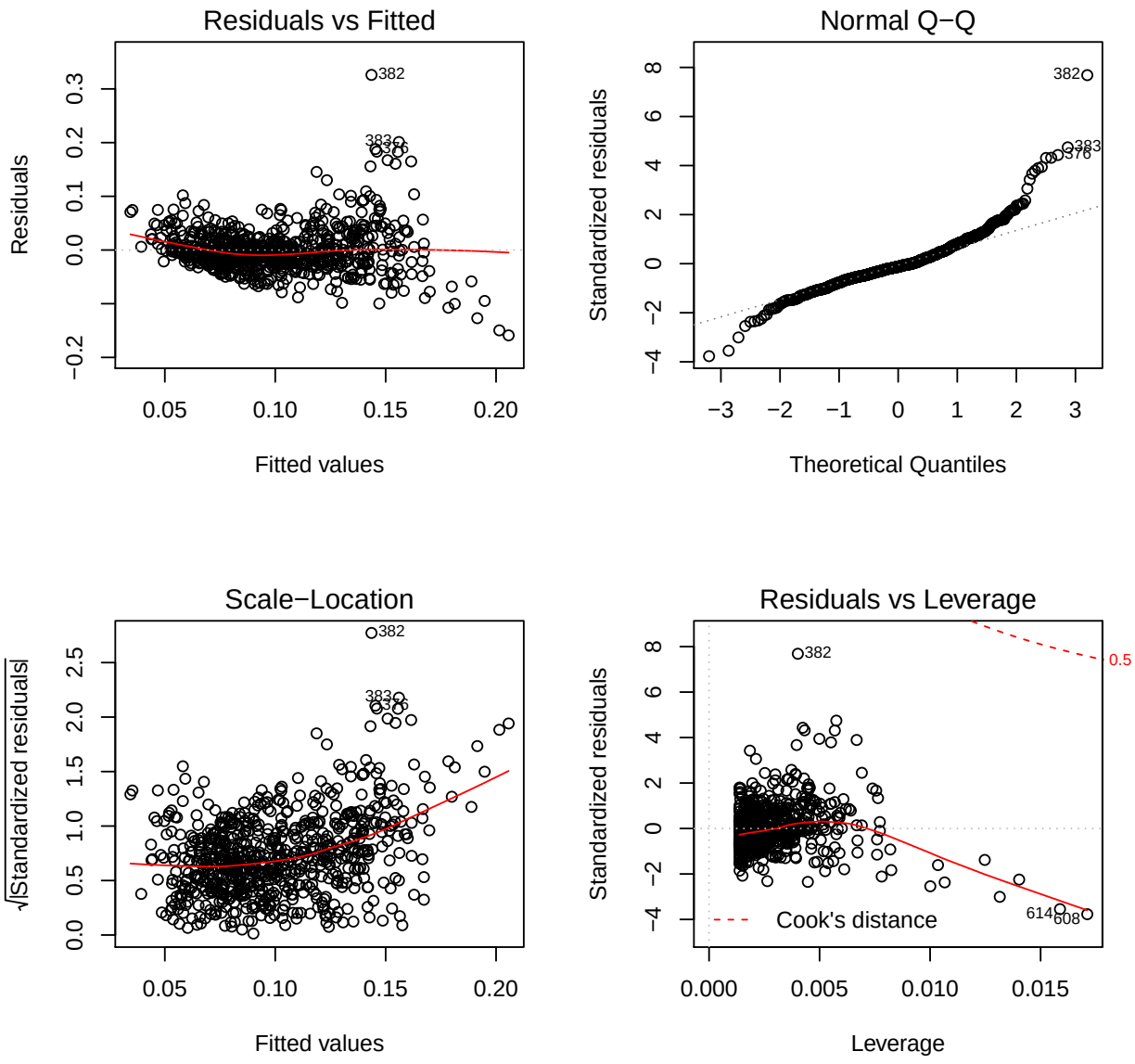

FIGURE 3: *The basic* `plot` *function applied to our running example model.*

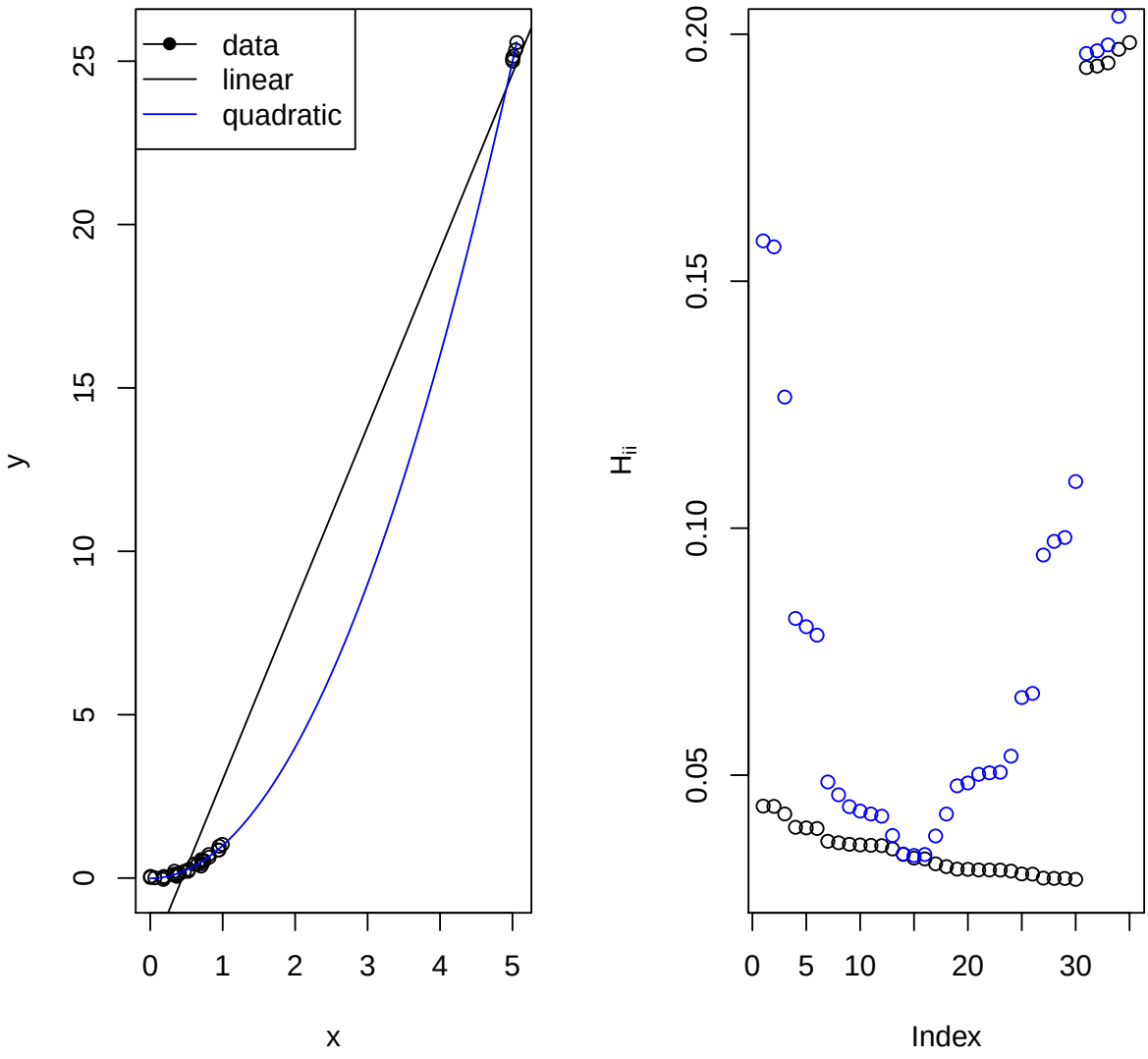

FIGURE 4: *The points in the upper-right are outliers for any linear model fit through the main body of points, but dominate the line because of their very high leverage; they'd be identified as outliers. But all points were generated from a quadratic model.*

The moral of Figure 4 is that data points can look like outliers because we're looking for the wrong pattern. If when we find apparent outliers and we can't convince ourselves that data is erroneous or irrelevant, we should consider changing our model, before, or as well as, deleting them.

4.3 Robust Linear Regression

A final alternative is to change how we estimate our model. Everything we've done has been based on ordinary least-squares (OLS) estimation. Because the squared error grows very rapidly with the error, OLS can be very strongly influenced by a few large residuals. We might, therefore, a different method of estimating the parameters. Estimation techniques which are less influenced by outliers in the residuals than OLS are called **robust estimators**, or (for regression models) **robust regression**.

Usually robust estimation, like OLS, is based on minimizing a function of the form: function of the errors:

$$\tilde{\beta} = \underset{\mathbf{b}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \rho(y_i - \mathbf{x}_i \mathbf{b}). \tag{5}$$

Different choices of ρ, the **loss function**, yield different estimators. $\rho(u) = |u|$ is **least absolute deviation** (LAD) estimation. Using $\rho(u) = u^2$ corresponds to OLS. A popular compromise is to use **Huber's** loss function

$$\rho(u) = \begin{cases} u^2 & |u| \leq c \\ 2c|u| - c^2 & |u| \geq c. \end{cases} \tag{6}$$

Notice that Huber's loss looks like squared error for small errors, but like absolute error for large errors. Huber's loss is designed to be continuous at c, and have a continuous first derivative there as well (which helps with optimization). We need to pick the scale c at which it switches over from acting like squared error to acting like absolute error; this is usually done using a robust estimate of the noise standard deviation σ.

Robust estimation with Huber's loss can be conveniently done with the `rlm` function in the `MASS` package, which, as the name suggests, is designed to work very much like `lm`.

```
library(MASS)
out = rlm(Mobility ~ Commute,data=mobility)
summary(out)
```

```
##
## Call: rlm(formula = Mobility ~ Commute, data = mobility)
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.148719 -0.019461 -0.002341  0.021093  0.332347
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept)  0.0028  0.0043     0.6398
## Commute      0.2077  0.0091    22.7939
##
## Residual standard error: 0.0293 on 727 degrees of freedom
```

Robust linear regression is designed for the situation where it's still true that $Y = X\beta + \epsilon$, but the noise ϵ is not very close to Gaussian, and indeed is sometimes "contaminated" by wildly larger values. It does nothing to deal with non-linearity, or correlated noise, or even some points having excessive leverage because we're insisting on a linear model.