Regression Assumptions & Diagnostics

Fundamental Techniques in Data Science with R

Utrecht University

Kyle M. Lang

Department of Methodology & Statistics
Utrecht University

Outline

Assumptions & Diagnostics

Why Assume?

Consider the following equation:

$$5 = x + y$$

What are the values of x and y?

Why Assume?

Consider the following equation:

$$5 = x + y$$

What are the values of x and y?

$$y = 5 - x$$

Why Assume?

Consider the following equation:

$$5 = x + y$$

What are the values of x and y?

$$y = 5 - x$$

What if we assume that $y = x$?

Why Assume?

Consider the following equation:

$$5 = x + y$$

What are the values of x and y?

$$y = 5 - x$$

What if we assume that $y = x$?

$$5 = x + y$$
$$0 = x - y$$

Why Assume?

Consider the following equation:

$$5 = x + y$$

What are the values of x and y?

$$y = 5 - x$$

What if we assume that $y = x$?

$$5 = x + y$$
$$0 = x - y$$

Now we have enough information:

$$5 = x + x = 2x \quad \Rightarrow \quad x = y = 2.5$$

Assumptions of MLR

The assumptions of the linear model can be stated as follows:

1. The model is linear in the parameters.
   - This is OK: $Y = \beta_0 + \beta_1 X + \beta_2 Z + \beta_3 XZ + \beta_4 X^2 + \beta_5 X^3 + \varepsilon$
   - This is not: $Y = \beta_0 X^{\beta_1} + \varepsilon$
2. The predictor matrix is *full rank*.
   - $N > P$
   - No X_p can be a linear combination of other predictors.

Assumptions of MLR

3. The predictors are strictly exogenous.
   - The predictors do not correlated with the errors.
   - $\text{Cov}(\hat{Y}, \varepsilon) = 0$
   - $\text{E}[\varepsilon_n] = 0$
4. The errors have constant, finite variance.
   - $\text{Var}(\varepsilon_n) = \sigma^2 < \infty$
5. The errors are uncorrelated.
   - $\text{Cov}(\varepsilon_i, \varepsilon_j) = 0, \ i \neq j$
6. The errors are normally distributed.
   - $\varepsilon \sim \text{N}(0, \sigma^2)$

Assumptions of MLR

The assumption of *spherical errors* combines Assumptions 4 and 5.

$$\text{Var}(\varepsilon) = \begin{bmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ & & & \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & \sigma^2 \end{bmatrix} = \sigma^2 \mathbf{I}_N$$

We can combine Assumptions 3, 4, 5, and 6 by assuming independent and identically distributed normal errors:

- $\varepsilon \overset{iid}{\sim} \text{N}(0, \sigma^2)$

Consequences of Violating Assumptions

1. If the model is not linear in the parameters, then we're not even working with linear regression.
   - We need to move to entirely different modeling paradigm.
2. If the predictor matrix is not full rank, the model is not estimable.
   - The parameter estimates cannot be uniquely determined from the data.
3. If the predictors are not exogenous, the estimated regression coefficients will be biased.
4. If the errors are not spherical, the standard errors will be biased.
   - The estimated regression coefficients will be unbiased, though.
5. If errors are non-normal, small-sample inferences may be biased.
   - The justification for some tests and procedures used in regression analysis may not hold.

Regression Diagnostics

If some of the assumptions are (grossly) violated, the inferences we make using the model may be wrong.

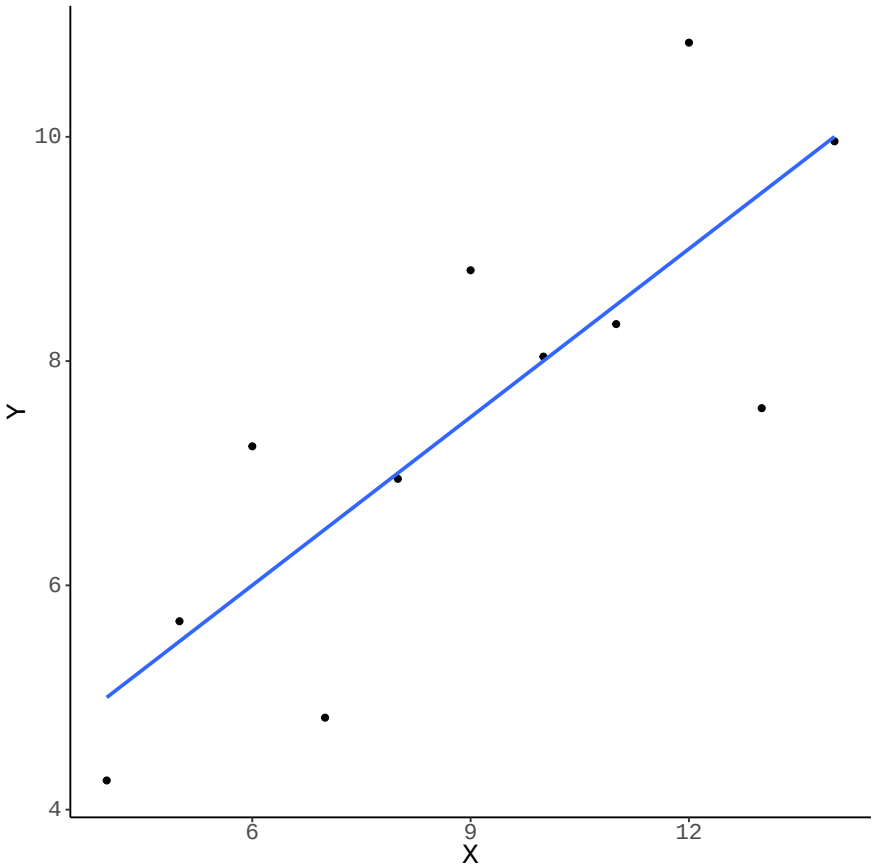
- We need to check the tenability of our assumptions before leaning too heavily on the model estimates.

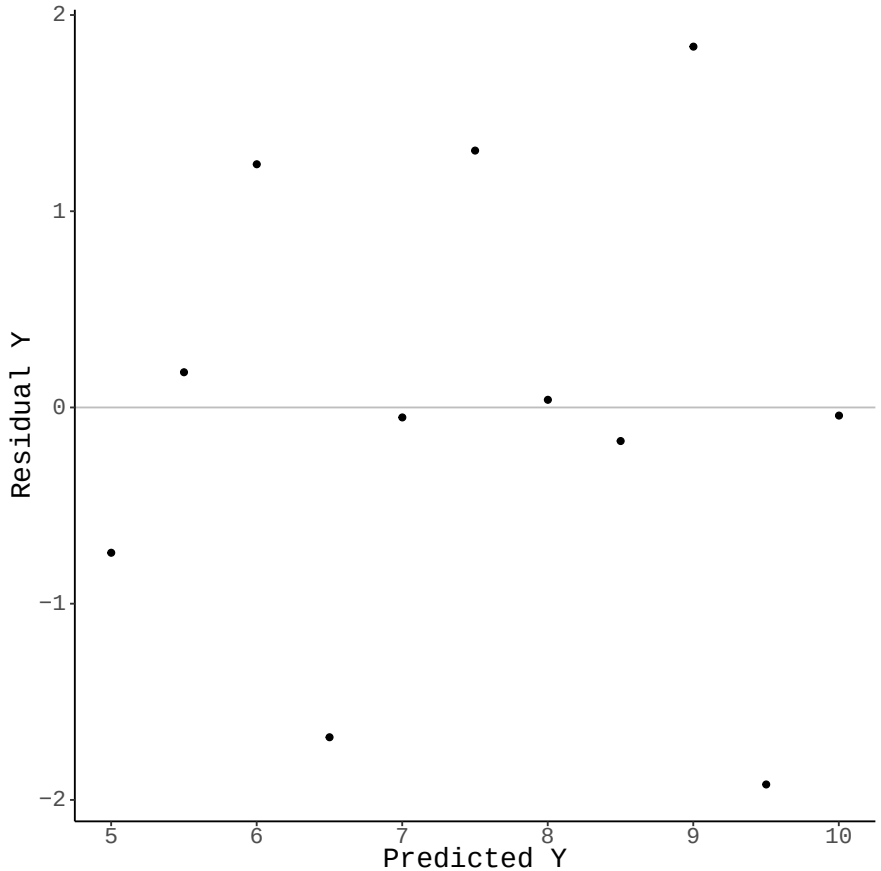
These checks are called *regression diagnostics*.

- Graphical visualizations
- Quantitative indices/measures
- Formal statistical tests

Residual Plots

One of the most useful diagnostic graphics is the plot of residuals vs. predicted values.

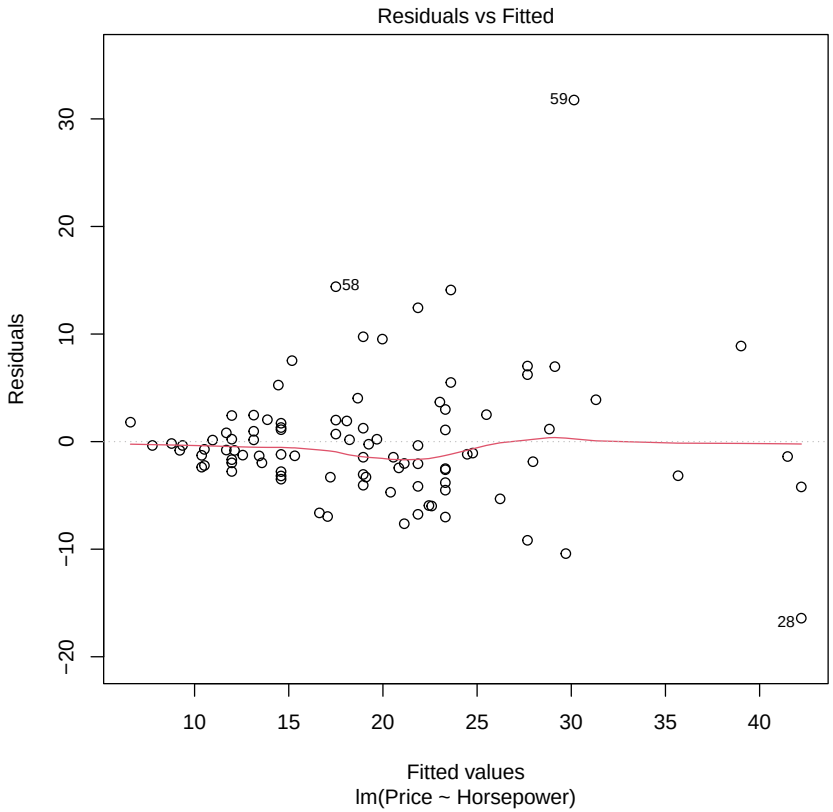




Residual Plots

We can easily generate a simple plot of residuals vs. fitted values by plotting the fitted `lm()` object in R.

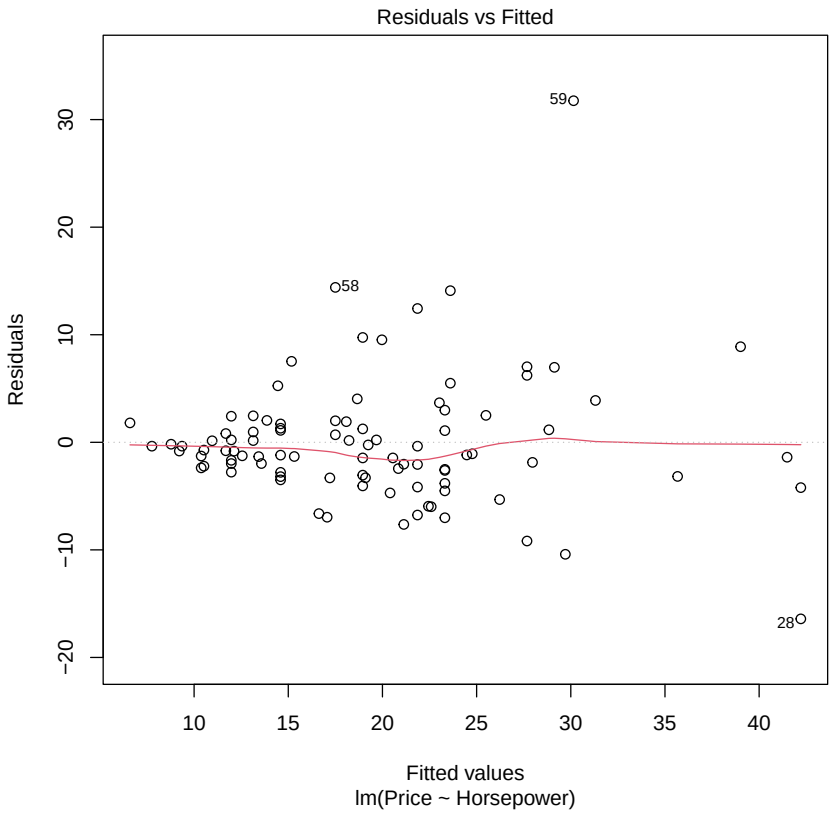
```
out1 <- lm(Price ~ Horsepower,
           data = Cars93)

plot(out1, 1)
```



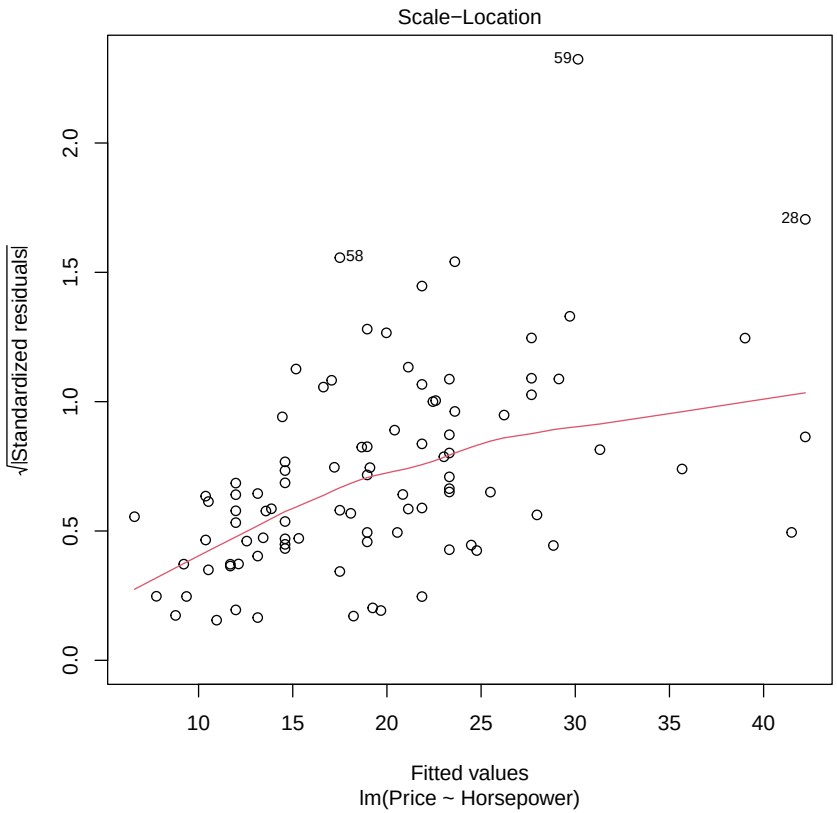
Heteroscedasticity

Non-constant error variance (*heteroscedasticity*) violates Assumption 4.

```
plot(out1, 1)
```




```
plot(out1, 3)
```



Consequences of Heteroscedasticity

Non-constant error variance will not bias the parameter estimates.

- The best fit line is still correct.
- Our measure of uncertainty around that best fit line is wrong.

Heteroscedasticity will bias standard errors (usually downward).

- Test statistics will be too large.
- CIs will be too narrow.
- We will have inflated Type I error rates.

To get valid inference, we need to address (severe) heteroscedasticity.

Treating Heteroscedasticity

1. Transform your outcome using a concave function (e.g., $\ln(Y)$, $\sqrt{Y}$).
   - These transformations will shrink extreme values more than small/moderate ones.
   - It's usually a good idea to first shift the variable's scale by setting the minimum value to 1.

2. Refit the model using *weighted least squares*.
   - Create inverse weights using functions of the residual variances or quantities highly correlated therewith.

3. Use a *Heteroscedasticity Consistent* (HC) estimate of the asymptotic covariance matrix.
   - Robust standard errors, Huber-White standard errors, Sandwich estimators
   - HC estimators correct the standard errors for non-constant error variance.

Example

```r
## The 'sandwich' package provides several HC estimators:
library(sandwich)

## the 'lmtest' package provides fancy testing tools for linear models:
library(lmtest)

## Use sandwich estimator to compute ACOV matrix:
hcCov <- vcovHC(out1)

## Test coefficients with robust SEs:
robTest <- coeftest(out1, vcov = hcCov)

## Test coefficients with default SEs:
defTest <- summary(out1)$coefficients
```

Example

```
## Compare robust and default approaches:
robTest


t test of coefficients:

             Estimate Std. Error t value  Pr(>|t|)
(Intercept) -1.398769   2.078200 -0.6731    0.5026
Horsepower   0.145371   0.017164  8.4696 4.051e-13 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

defTest

              Estimate Std. Error    t value     Pr(>|t|)
(Intercept) -1.3987691  1.8200164 -0.7685475 4.441519e-01
Horsepower   0.1453712  0.0118978 12.2183251 6.837464e-21
```

Correlated Errors

Errors can become correlated in two basic ways:

1. Serial dependence
   - When modeling longitudinal data, the errors for a given observational unit are correlated over time.
   - We can detect temporal dependence by examining the *autocorrelation* of the residuals.

2. Clustering
   - Your data have some important, unmodeled, grouping structure.
     - Children nested within classrooms
     - Romantic couples
     - Departments within a company
   - We can detect problematic levels of clustering with the *intraclass correlation coefficient* (ICC).
     - We need to know the clustering variable to apply the ICC.

Treating Correlated Errors

Serially dependent errors in a longitudinal model usually indicate an inadequate model.

- Your model is ignoring some important aspect of the temporal variation that is being absorbed by the error terms.
- Hopefully, you can add the missing component to your model.

Treating Correlated Errors

Clustering can be viewed as theoretically meaningful or as a nuisance factor that just needs to be controlled.

- If the clustering is meaningful, you should model the data using *multilevel modeling*.
  - Hierarchical linear regression
  - Mixed models
  - Random effects models
- If the clustering is an uninteresting nuisance, you can use specialized HC variance estimators that deal with clustering.

Example

```
## Read in some data:
LeeBryk <- readRDS(paste0(dataDir, "lee_bryk.rds"))

## Check the data:
str(LeeBryk, vec.len = 3)

'data.frame': 7185 obs. of  5 variables:
 $ schoolid: int  1 1 1 1 1 1 1 1 ...
 $ math    : num  5.88 19.71 20.35 8.78 ...
 $ ses     : num  -1.53 -0.59 -0.53 -0.67 -0.16 0.02 -0.62 -1 ...
 $ mses    : num  -0.43 -0.43 -0.43 -0.43 -0.43 -0.43 -0.43 -0.43 ...
 $ sector  : Factor w/ 2 levels "public","private": 1 1 1 1 1 1 1 1 ...

## Estimate a linear regression model:
fit <- lm(math ~ ses + sector, data = LeeBryk)

## Calculate the residual ICC:
ICC::ICCbare(x = LeeBryk$schoolid, y = resid(fit))

[1] 0.07487712
```

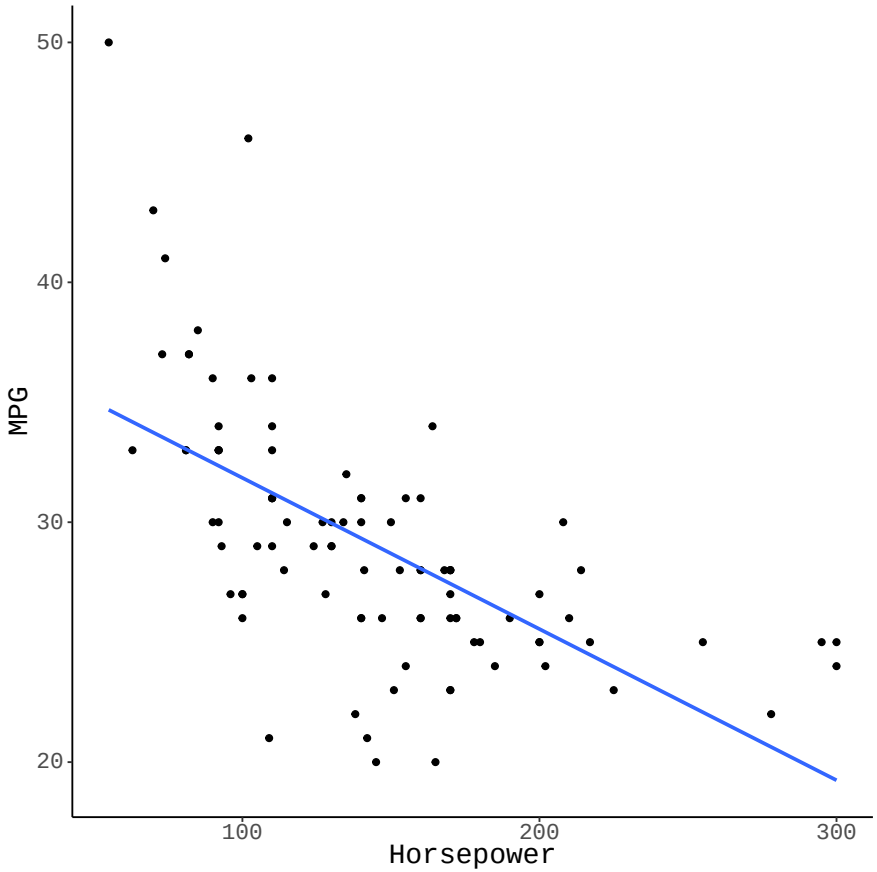
Example

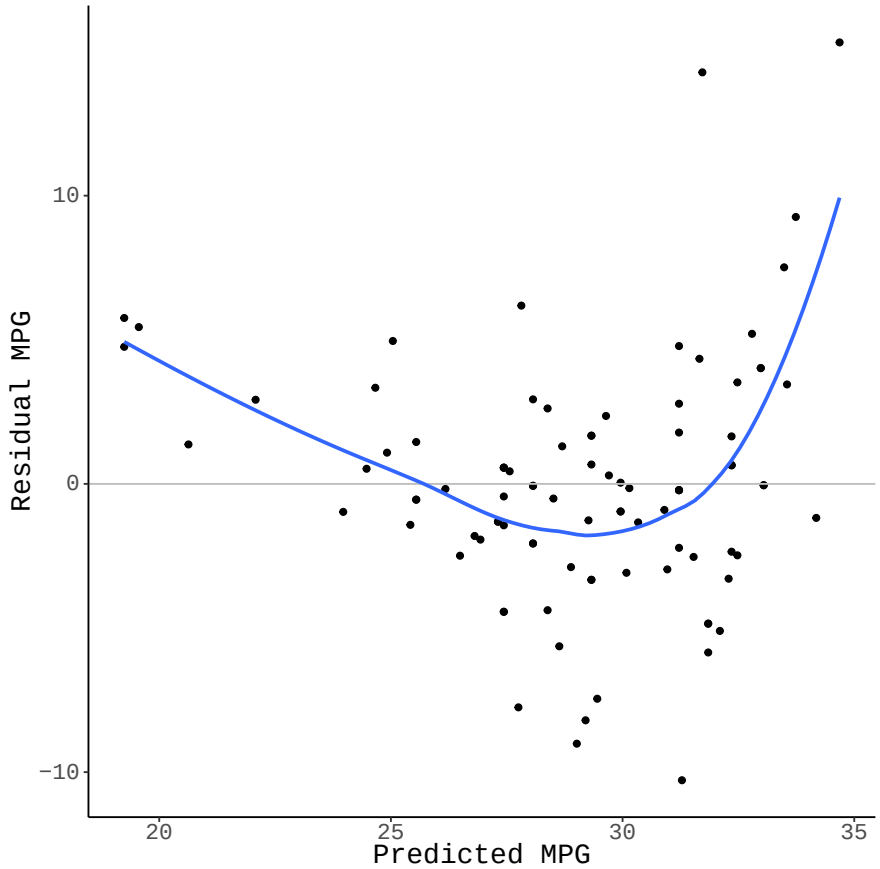
```
## Robust tests:
coeftest(fit, vcov = vcovCL(fit, ~ schoolid))


t test of coefficients:

              Estimate Std. Error t value  Pr(>|t|)
(Intercept)   11.79965    0.20318 58.0746 < 2.2e-16 ***
ses            2.94860    0.12794 23.0475 < 2.2e-16 ***
sectorprivate  1.93495    0.31717  6.1006 1.111e-09 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## Raw tests:
summary(fit)$coefficients

               Estimate Std. Error   t value      Pr(>|t|)
(Intercept)   11.799654 0.10612759 111.18366  0.000000e+00
ses            2.948605 0.09782968  30.14019 5.002687e-188
sectorprivate  1.934953 0.15249200  12.68888  1.676478e-36
```

Linearity

Each modeled X must exhibit a linear relation with Y.

- We can define X via nonlinear transformations of the orgininal data.





Treating Residual Nonlinearity

Nonlinearity in the residual plots is usually a sign of either:

1. Model misspecification
2. Influential observations

This type of model misspecification usually implies omitted functions of modeled variables.

- Polynomial terms
- Interactions

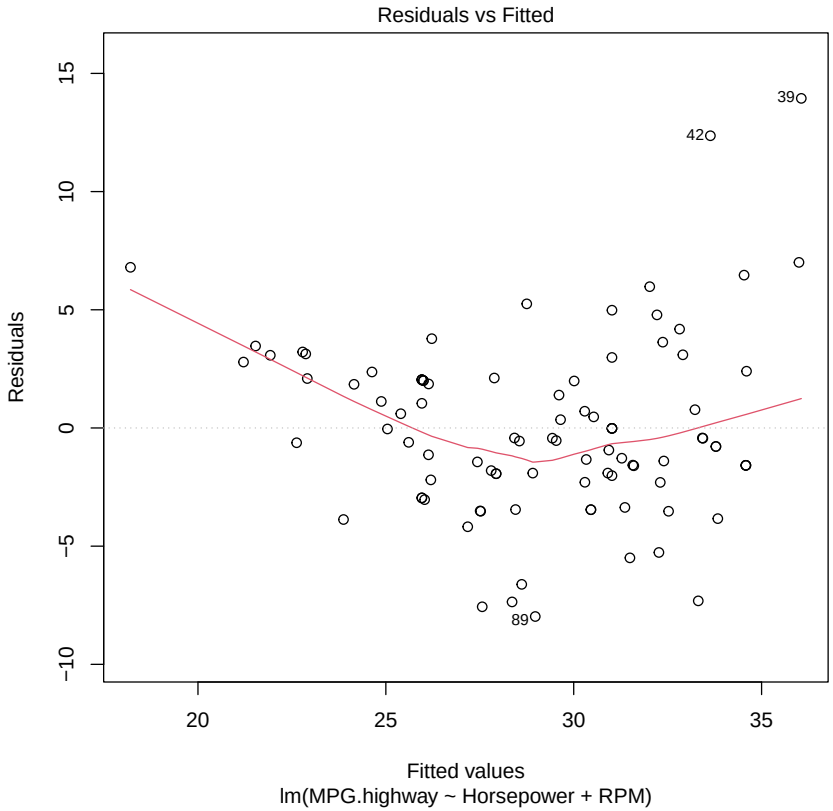
The solution is to include the omitted term into the model and refit.

- This is very much easier said than done.

Limitations of Residual Plots

In multiple regression models, basic residual plots won't tell us which predictors exhibit nonlinear associations.

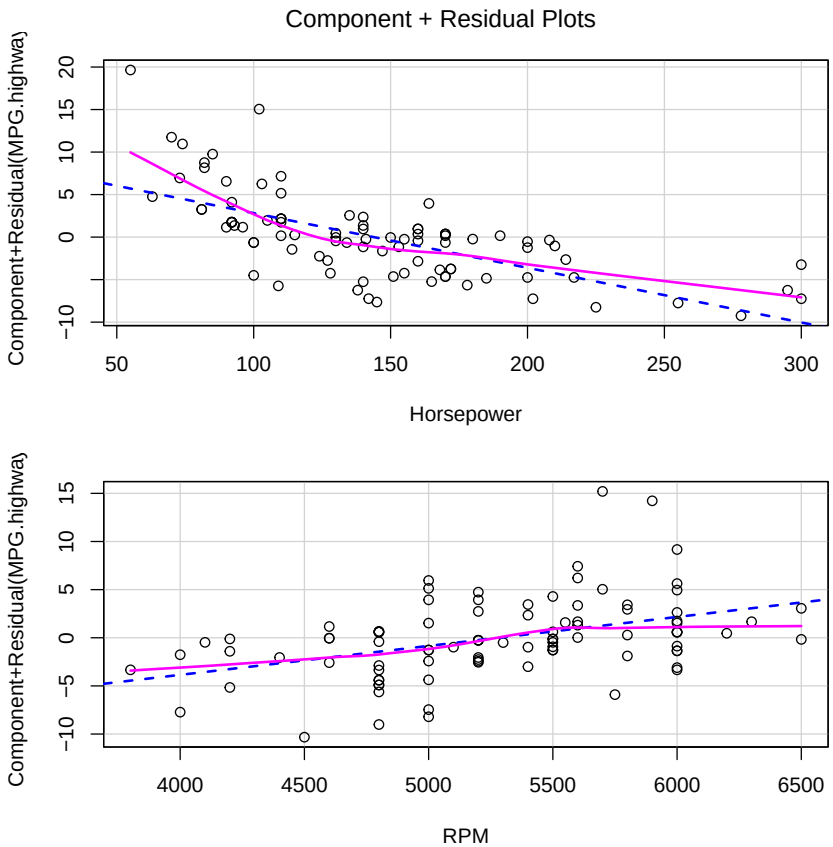
```
out3 <-
  lm(MPG.highway ~ Horsepower + RPM,
     data = Cars93)
```



Component + Residual Plots

We can use *Component + Residual Plots* (AKA, partial residual plots) to visualize the unique effects of each X variable.

```
library(car)
crPlots(out3)
```



Omitted Variables

The most common cause of endogeneity (i.e., violating Assumption 3) is *omitted variable bias*.

- If we leave an important predictor variable out of our equation, some modeled predictors will become endogenous and their estimated regression slopes will be biased.
- The omitted variable must be correlated with Y and at least one of the modeled X_p, to be a problem.

Omitted Variables

Assume the following is the true regression model.

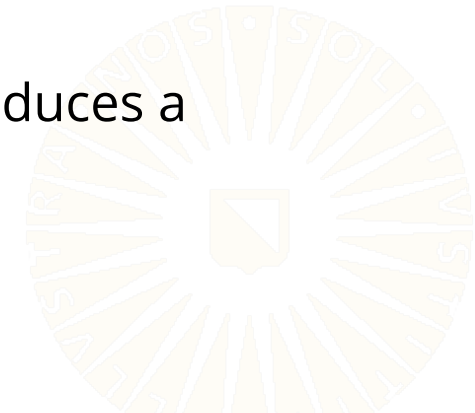
$$Y = \beta_0 + \beta_1 X + \beta_2 Z + \varepsilon$$

Now, suppose we omit Z from the model:

$$Y = \beta_0 + \beta_1 X + \omega$$
$$\omega = \varepsilon + \beta_2 Z$$

Our new error, ω, is a combination of the true error, ε, and the omitted term, $\beta_2 Z$.

- Consequently, if X and Z are correlated, omitting Z induces a correlation between X and ω (i.e., endogeneity).



Treating Omitted Variable Bias

Omitted variable bias can have severe consequences, but you can't really test for it.

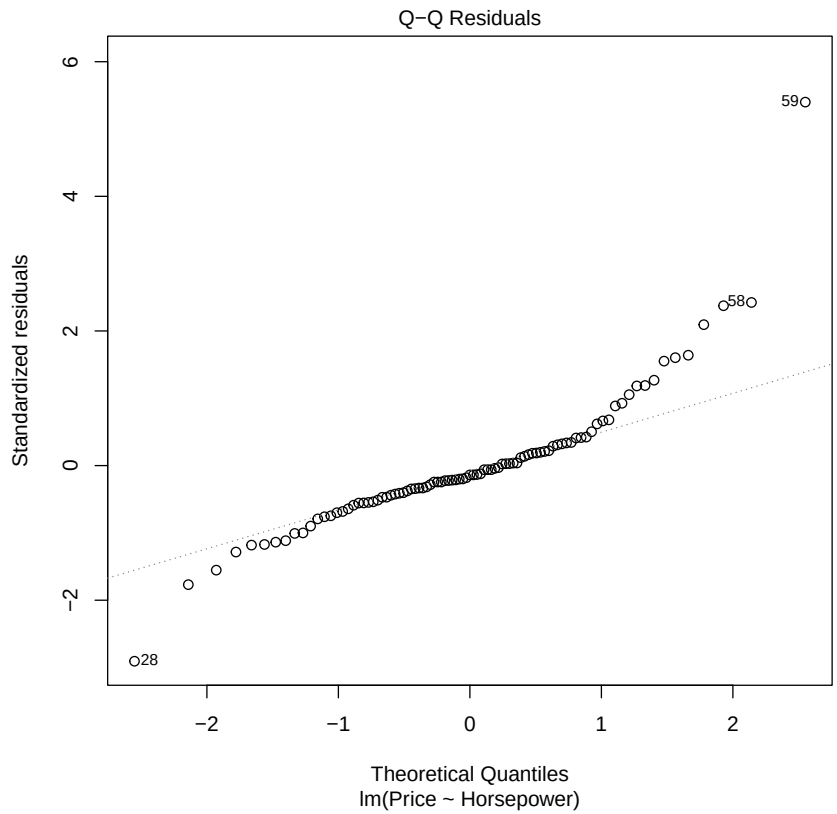
- The *errors* are correlated with the predictors, but our model is estimated under the assumption of exogeneity, so the *residuals* from our model will generally be uncorrelated with the predictors.
- We mostly have to pro-actively work to include all relevant variables in our model.

Normality Assumption

```
plot(out1, 2)
```

One of the best ways to evaluate the normality of the error distribution with a Q-Q Plot.

- Plot the quantiles of the residual distribution against the theoretically ideal quantiles.
- We can actually use a Q-Q Plot to compare any two distributions.



Consequences of Violating Normality

In small samples, with *fixed* predictors, normally distributed errors imply normal sampling distributions for the regression coefficients.

- In large samples, the central limit theorem implies normal sampling distributions for the coefficients, regardless of the error distribution.

Consequences of Violating Normality

In small samples, with *fixed* predictors, normally distributed errors imply normal sampling distributions for the regression coefficients.

- In large samples, the central limit theorem implies normal sampling distributions for the coefficients, regardless of the error distribution.

Prediction intervals require normally distributed errors.

- Confidence intervals for predictions share the same normality requirements as the coefficients' sampling distributions.

Consequences of Violating Normality

In small samples, with *fixed* predictors, normally distributed errors imply normal sampling distributions for the regression coefficients.

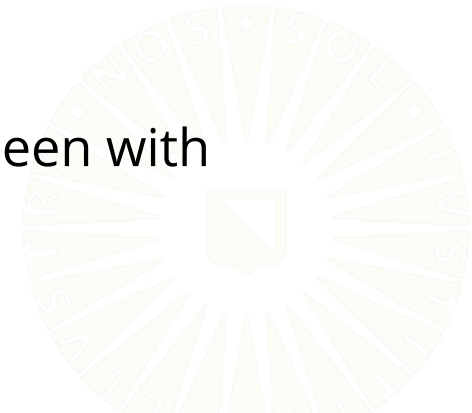
- In large samples, the central limit theorem implies normal sampling distributions for the coefficients, regardless of the error distribution.

Prediction intervals require normally distributed errors.

- Confidence intervals for predictions share the same normality requirements as the coefficients' sampling distributions.

Parameter estimates will not be fully efficient.

- Standard errors will be larger than they would have been with normally distributed errors.



Treating Violations of Normality

We usually don’t need to do anything about non-normal errors.

- The CLT will protect our inferences.

Treating Violations of Normality

We usually don't need to do anything about non-normal errors.

- The CLT will protect our inferences.

We can use *bootstrapping* to get around the need for normality.

1. Treat your sample as a synthetic population from which you draw many new samples (with replacement).
2. Estimate your model in each new sample.
3. The replicates of your estimated parameters generate an empirical sampling distribution that you can use for inference.

Treating Violations of Normality

We usually don't need to do anything about non-normal errors.

- The CLT will protect our inferences.

We can use *bootstrapping* to get around the need for normality.

1. Treat your sample as a synthetic population from which you draw many new samples (with replacement).
2. Estimate your model in each new sample.
3. The replicates of your estimated parameters generate an empirical sampling distribution that you can use for inference.

Bootstrapping can be used for inference on pretty much any estimable parameter, but it won't work with small samples.

- Need to assume that your sample is representative of the population

Influential Observations

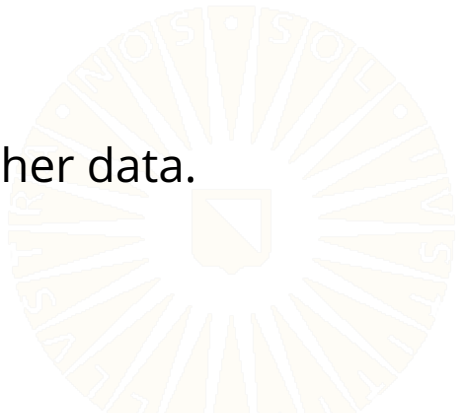
Influential Observations

Influential observations contaminate analyses in two ways:

1. Exert too much influence on the fitted regression model
2. Invalidate estimates/inferences by violating assumptions

There are two distinct types of influential observations:

1. Outliers
    - Observations with extreme outcome values, relative to the other data.
    - Observations with outcome values that fit the model very badly.
2. High-leverage observations
    - Observation with extreme predictor values, relative to other data.



Outliers

Outliers can be identified by scrutinizing the residuals.

- Observations with residuals of large magnitude may be outliers.
- The difficulty arises in quantifying what constitutes a “large” residual.

If the residuals do not have constant variance, then we cannot directly compare them.

- We need to standardize the residuals in some way.

Detecting Outliers

We are specifically interested in *externally studentized residuals*.

- We can't simply standardize the ordinary residuals.
  - *Internally studentized residuals*
  - Outliers can pull the regression line towards themselves.
  - The internally studentized residuals for outliers will be too small.

Begin by defining the concept of a *deleted residual*:

$$\hat{\varepsilon}_{(n)} = Y_n - \hat{Y}_{(n)}$$

- $\hat{\varepsilon}_{(n)}$ quantifies the distance of Y_n from the regression line estimated after excluding the nth observation.

Studentized Residuals

If we standardize the deleted residual, $\hat{\varepsilon}_{(n)}$, we get the externally studentized residual:

$$t_{(n)} = \frac{\hat{\varepsilon}_{(n)}}{SE_{\hat{\varepsilon}_{(n)}}}$$

The externally studentized residuals have two very useful properties:

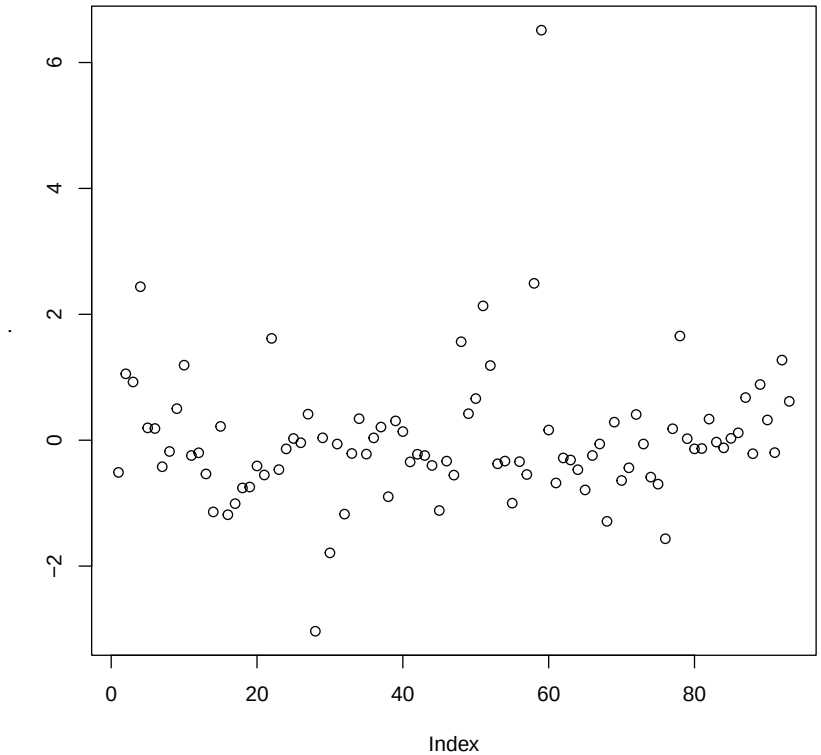
1. Each $t_{(n)}$ is scaled equivalently.
   - We can directly compare different $t_{(n)}$.
2. The $t_{(n)}$ are *Student's t* distributed.
   - We can quantify outliers in terms of quantiles of the t distribution.
   - $|t_{(n)}| > 3.0$ is a common rule of thumb for flagging outliers.

Studentized Residual Plots

```
rstudent(out1) %>% plot()
```

Index plots of the externally studentized residuals can help spotlight potential outliers.

- Look for observations that clearly "stand out from the crowd."



High-Leverage Points

We identify high-leverage observations through their *leverage* values.

- An observation's leverage, h_n, quantifies the extent to which its predictors affect the fitted regression model.
- Observations with X values very far from the mean, $\bar{X}$, affect the fitted model disproportionately.

High-Leverage Points

We identify high-leverage observations through their *leverage* values.

- An observation's leverage, h_n, quantifies the extent to which its predictors affect the fitted regression model.
- Observations with X values very far from the mean, $\bar{X}$, affect the fitted model disproportionately.

In simple linear regression, the nth leverage is given by:

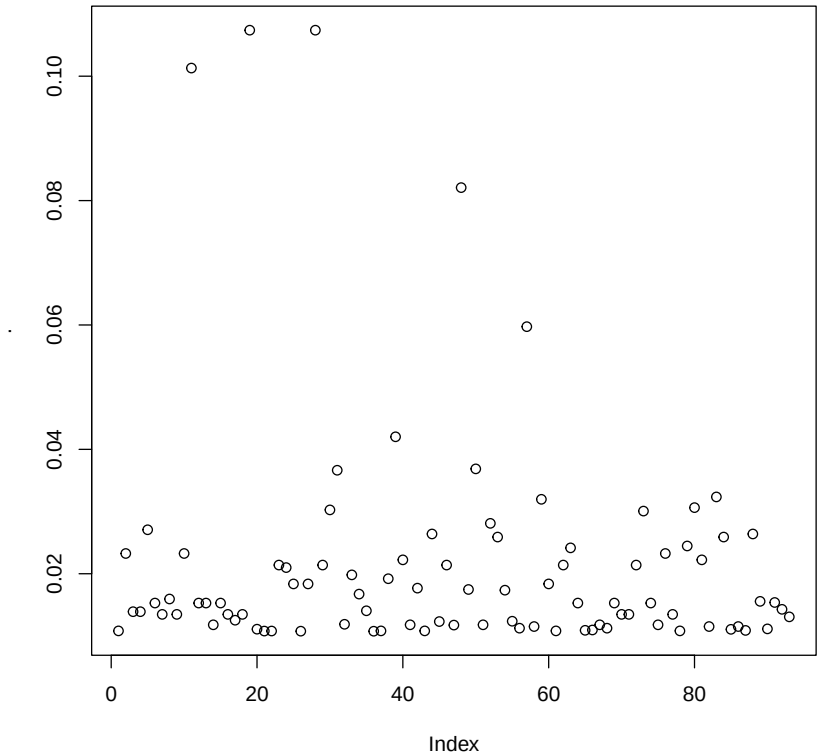
$$h_n = \frac{1}{N} + \frac{\left(X_n - \bar{X}\right)^2}{\sum_{m=1}^{N}\left(X_m - \bar{X}\right)^2}$$

Leverage Plots

```
hatvalues(out1) %>% plot()
```

Index plots of the leverage values can help spotlight high-leverage points.

- Again, look for observations that clearly "stand out from the crowd."



Outliers & Leverages → Influential Points

Observations with high leverage or large (externally) studentized residuals are not necessarily influential.

- High-leverage observations tend to be more influential than outliers.
- The worst problems arise from observations that are both outliers and have high leverage.

Measures of influence simultaneously consider extremity in both X and Y dimensions.

- Observations with high measures of influence are very likely to cause problems.

Measures of Influence

All measures of influence use the same logic as the deleted residual.

- Compare models estimated from the whole sample to models estimated from samples excluding individual observations.

One of the most common measures of influence is *Cook's Distance*.

$$\text{Cook's } D_n = \frac{\sum_{n=1}^{N} \left(\hat{Y}_n - \hat{Y}_{(n)} \right)^2}{(P+1)\,\hat{\sigma}^2}$$

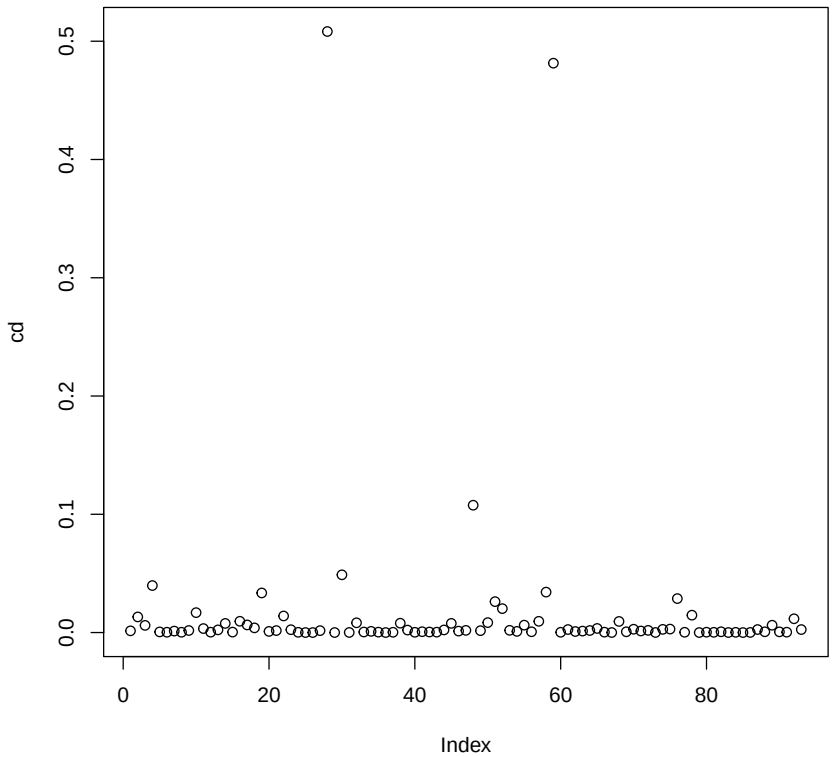
$$= (P+1)^{-1} t_n^2 \frac{h_n}{1-h_n}$$

Plots of Cook's Distance

```
cd <- cooks.distance(out1)
plot(cd)
```

Index plots of Cook's distances can help spotlight the influential points.

- Look for observations that clearly "stand out from the crowd."

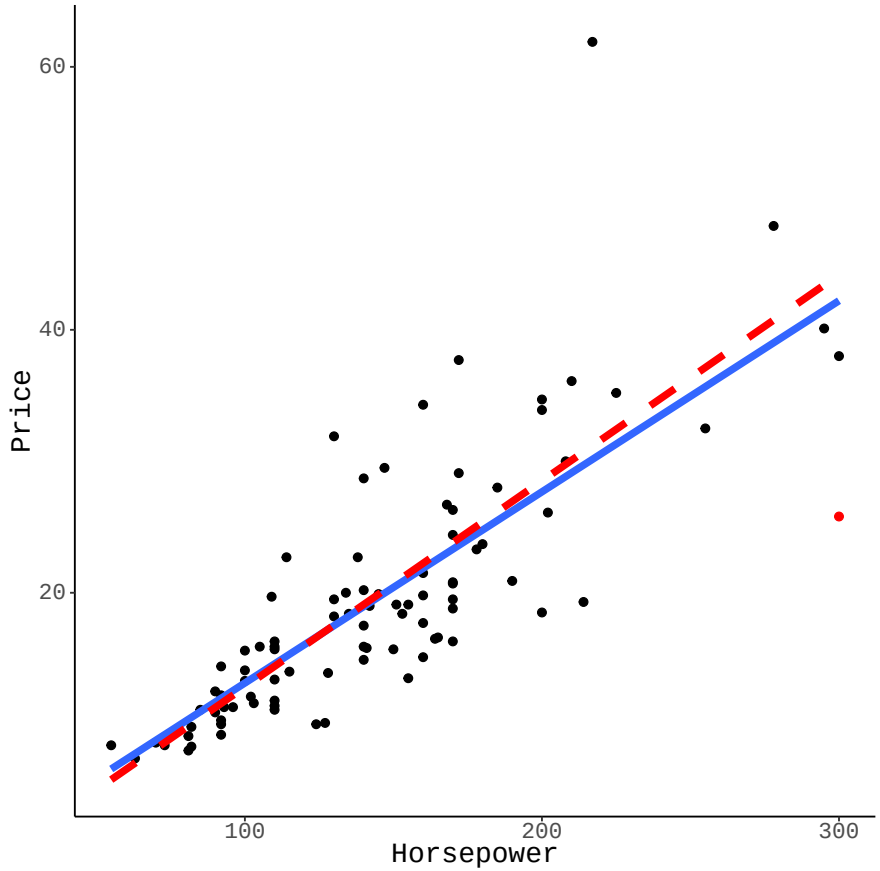


Removing Influential Observations

```
(maxD <- which.max(cd))

28
28
```

Observation number 28 was the most influential according to Cook's Distance.

- Removing that observation has a small impact on the fitted regression line.
- Influential observations don't only affect the regression line, though.



Removing Influential Observations

```
## Exclude the influential case:
Cars93.2 <- Cars93[-maxD, ]

## Fit model with reduced sample:
out2 <- lm(Price ~ Horsepower, data = Cars93.2)

round(summary(out1)$coefficients, 6)

             Estimate Std. Error   t value Pr(>|t|)
(Intercept) -1.398769   1.820016 -0.768548 0.444152
Horsepower   0.145371   0.011898 12.218325 0.000000

round(summary(out2)$coefficients, 6)

             Estimate Std. Error   t value Pr(>|t|)
(Intercept) -2.837646   1.806418 -1.570868 0.119722
Horsepower   0.156750   0.011996 13.066942 0.000000
```

Removing Influential Observations

```
partSummary(out1, 2)

Residuals:
    Min      1Q  Median      3Q     Max
-16.413  -2.792  -0.821   1.803  31.753

partSummary(out2, 2)

Residuals:
     Min       1Q   Median       3Q      Max
-11.4069  -3.0349  -0.5912   1.8530  30.7229
```

Removing Influential Observations

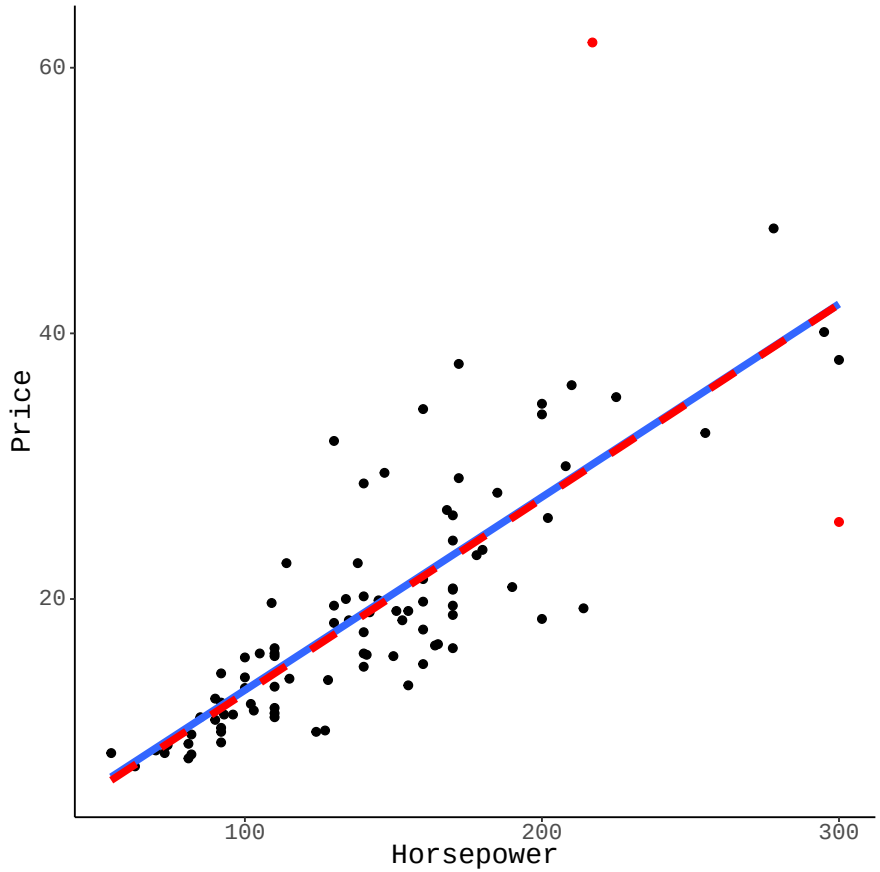
```
summary(out1)[c("sigma", "r.squared", "fstatistic")] %>%
    unlist() %>%
    head(3)

           sigma        r.squared fstatistic.value
        5.976953         0.621287       149.287468

summary(out2)[c("sigma", "r.squared", "fstatistic")] %>%
    unlist() %>%
    head(3)

           sigma        r.squared fstatistic.value
       5.7243112        0.6548351      170.7449721
```

Removing Influential Observations

```
(maxDs <- sort(cd) %>% names() %>% tail(2) %>% as.numeric())

[1] 59 28
```

If we remove the two most influential observations, 59 and 28, the fitted regression line barely changes at all.

- The influences of these two observations were counteracting one another.
- We're probably still better off, though.



Removing Influential Observations

```
## Exclude influential cases:
Cars93.2 <- Cars93[-maxDs, ]

## Fit model with reduced sample:
out2.2 <- lm(Price ~ Horsepower, data = Cars93.2)

round(summary(out1)$coefficients, 6)

             Estimate Std. Error   t value Pr(>|t|)
(Intercept) -1.398769   1.820016 -0.768548 0.444152
Horsepower   0.145371   0.011898 12.218325 0.000000

round(summary(out2.2)$coefficients, 6)

             Estimate Std. Error   t value Pr(>|t|)
(Intercept) -1.695315   1.494767 -1.134166  0.25977
Horsepower   0.146277   0.009986 14.648807  0.00000
```

Removing Influential Observations

```
partSummary(out1, 2)

Residuals:
    Min      1Q  Median      3Q     Max
-16.413  -2.792  -0.821   1.803  31.753

partSummary(out2.2, 2)

Residuals:
     Min       1Q   Median       3Q      Max
-10.3079  -2.5786  -0.6084   1.9775  14.5793
```

Removing Influential Observations

```
summary(out1)[c("sigma", "r.squared", "fstatistic")] %>%
    unlist() %>%
    head(3)

           sigma        r.squared fstatistic.value
        5.976953         0.621287       149.287468

summary(out2.2)[c("sigma", "r.squared", "fstatistic")] %>%
    unlist() %>%
    head(3)

           sigma        r.squared fstatistic.value
       4.7053314        0.7068391      214.5875491
```

Treating Influential Points

The most common way to address influential observations is simply to delete them and refit the model.

- This approach is often effective—and always simple—but it is not fool-proof.
- Although an observation is influential, we may not be able to justify excluding it from the analysis.

Robust regression procedures can estimate the model directly in the presence of influential observations.

- Observations in the tails of the distribution are weighted less in the estimation process, so outliers and high-leverage points cannot exert substantial influence on the fit.
- We can do robust regression with the `rlm()` function from the **MASS** package.