NBA 4920/6921 Lecture 4

Linear Regression Part 2

Murat Unal

Johnson Graduate School of Management

09/09/2021

Agenda

- Quiz 3
- Modeling non-linearity
- Potential problems in linear regression
  Non-linearity
  Correlation of the error terms
  Heteroskedasticity
  Outliers
  High leverage points
  Collinearity
- Exercise

Load/install the following packages.

```
rm(list=ls())
options(digits = 3, scipen = 999)
library(tidyverse)
library(ISLR)
library(cowplot)
library(ggcorrplot)
library(stargazer)
library(corrr)
library(lmtest)
library(sandwich)
library(MASS)
library(car)
library(jtools)
```

Download the Advertising data and load it into R.

Read in the Credit and Auto data from the ISLR package

Read in the Boston data from the MASS package.

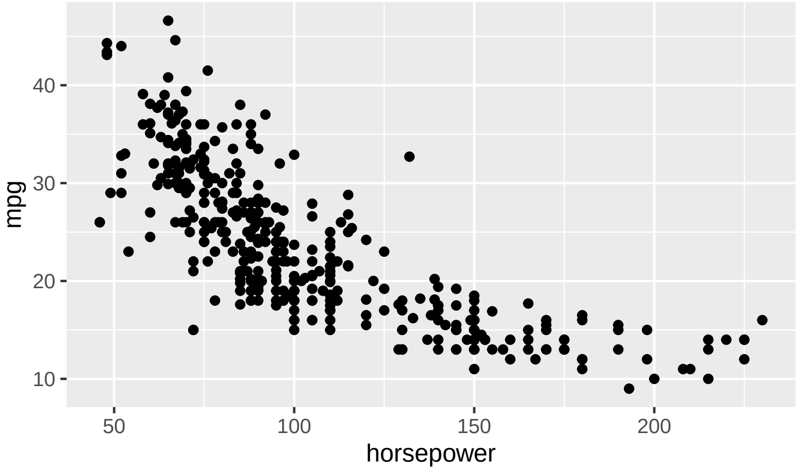
```r
data <- read.csv("Advertising.csv")
credit <- ISLR::Credit
auto <- ISLR::Auto
boston <- MASS::Boston
```

Modeling non-linearity

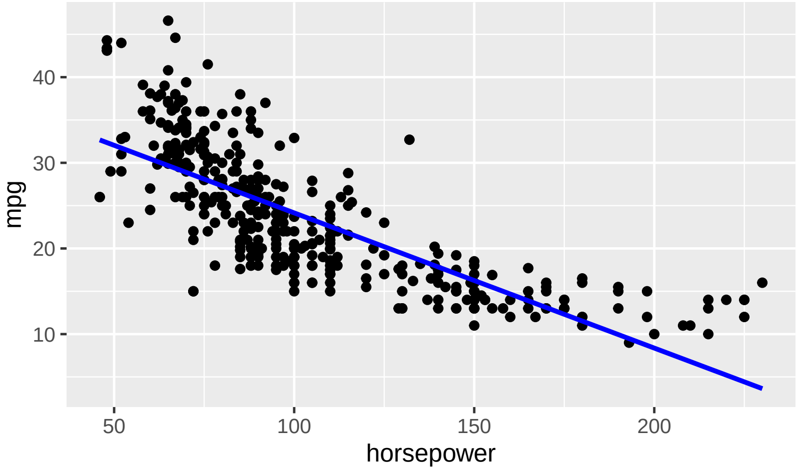
As discussed previously, the linear regression model assumes a linear relationship between the response and predictors.

But in some cases, the true relationship between the response and the predictors may be nonlinear.

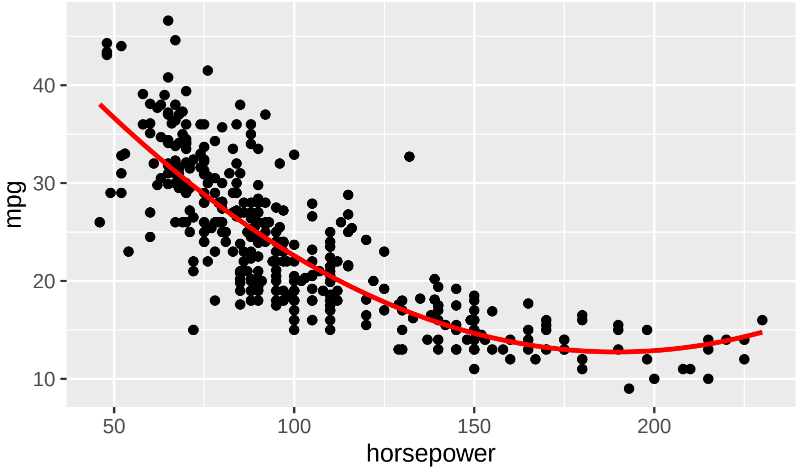
The following data points from the `auto` data suggest a quadratic shape between `mpg` and `horsepower`



The following data points from the `auto` data suggest a quadratic shape between `mpg` and `horsepower`



The following data points from the `auto` data suggest a quadratic shape between `mpg` and `horsepower`



A simple way to extend the linear model for accommodating non-linear relationships is to use **polynomial regression**.

This is done using a model with higher order variables.

The model now looks like this:

$$mpg = \beta_0 + \beta_1 horsepower + \beta_2 horsepower^2 + \epsilon$$

This model predicts mpg using a non-linear function of horsepower. But it is still a **linear model** becuase the model parameters are linear.

```r
summary(lm(mpg~horsepower, data=auto))$r.squared
```

```
[1] 0.606
```

```r
lm.q2 <-lm(mpg~horsepower+I(horsepower^2),
                                    data=auto)
summary(lm.q2)$r.squared
```

```
[1] 0.688
```

The quadratic fit appears to be substantially better than the fit obtained when just the linear term is included.

The R^2 of the quadratic fit is 0.688, compared to 0.606 for the linear fit

```r
summ(lm.q2,model.info = FALSE)
```

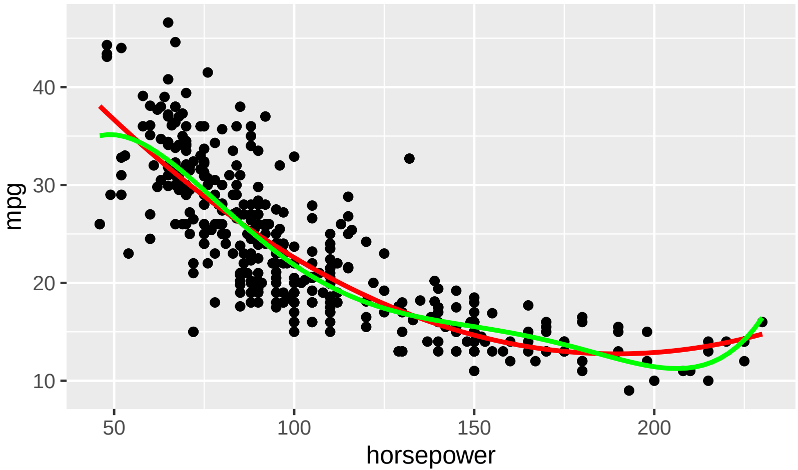
```
MODEL FIT:
F(2,389) = 428.02, p = 0.00
R² = 0.69
Adj. R² = 0.69

Standard errors: OLS
------------------------------------------------------
                         Est.   S.E.    t val.       p
---------------------- ------- ------ -------- ------
(Intercept)              56.90   1.80    31.60   0.00
horsepower               -0.47   0.03   -14.98   0.00
I(horsepower^2)           0.00   0.00    10.08   0.00
------------------------------------------------------
```

Also, the *p-value* for the quadratic term is highly significant justifying its inclusion.

What about a higher degree polynomial, such as 5?

Use the `poly()` function to create the polynomial within `lm()`.

```
lm.q5 <- lm(mpg~poly(horsepower,5), data=auto)
```



mpg
40
30
20
10
50
100
150
200
horsepower

The resulting fit seems unnecessarily wiggly—that is, including the additional terms does not lead to a substantial improvement in fit to the data.

```
lm.q5 <- lm(mpg~poly(horsepower,5), data=auto)
summary(lm.q5)$r.squared
```

```
[1] 0.697
```

Potential problems

When we fit a linear regression model to a particular data set, many problems may occur.

Most common among these are the following:

1. Non-linearity of the response-predictor relationships.
2. Correlation of error terms.
3. Non-constant variance of error terms.
4. Outliers.
5. High-leverage points.
6. Collinearity.

Non-linearity

Residual plots are a useful graphical tool for identifying non-linearity.
Plot the residuals versus the predicted (or fitted) values $\hat{Y}$.
Ideally, the residual plot will show no fitted discernible pattern.
The presence of a pattern may indicate a problem with some aspect of the linear model.

If the residual plot indicates that there are non-linear associations in the data, then a simple approach is to use non-linear transformations of the predictors, such as $log(X)$,$\sqrt{X}$, and X^2 in the regression model

Use the `resid()` function to get the residuals and the `fitted()` function to get the fitted values and plot the residuals against the fitted values.

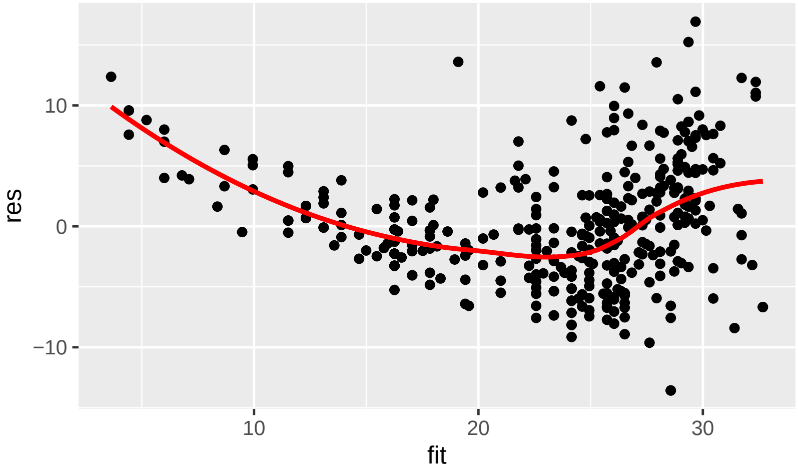
Let's implement this for two regressions:

$$mpg = \beta_0 + \beta_1 horsepower + \epsilon$$

$$mpg = \beta_0 + \beta_1 horsepower + \beta_2 horsepower^2 + \epsilon$$

```r
auto.lm <- lm(mpg~horsepower,data=auto)
res <- resid(auto.lm)
fit <- fitted(auto.lm)
resfit <- data.frame("res" = res, "fit" = fit)
```

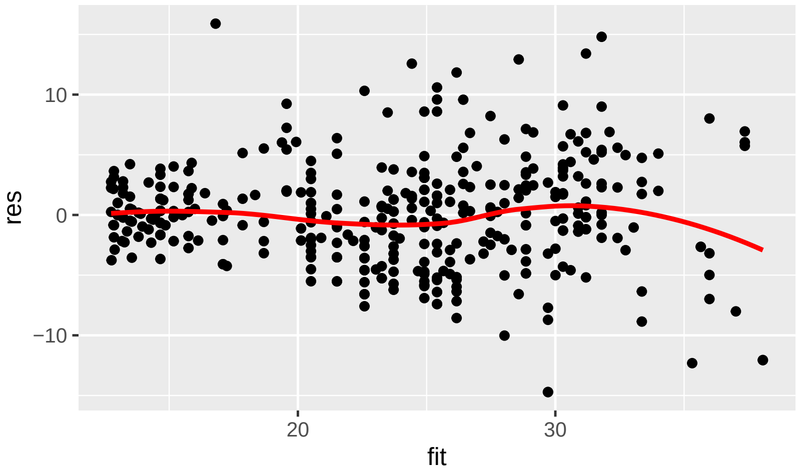
```r
ggplot(resfit, mapping = aes(x=fit, y=res)) +
      geom_point()+
      geom_smooth(method = "loess",
                  se=FALSE, colour="red")
```



At any fitted value, the mean of the residuals should be roughly 0. Clearly this is violated, thus the linearity assumption does not hold.

```r
auto.qm <- lm(mpg~horsepower+I(horsepower^2),
                   data=auto)
res <- resid(auto.qm)
fit <- fitted(auto.qm)
resfit <- data.frame("res" = res, "fit" = fit)
```

```r
ggplot(resfit, mapping = aes(x=fit, y=res)) +
      geom_point()+
      geom_smooth(method = "loess",
                  se=FALSE, colour="red")
```



Adding the transformed variable into the model corrected the violation.

Correlation of the error terms

An important assumption of the linear regression model is that the error terms are uncorrelated.

This means the fact that ϵ_i is positive provides little or no information about the sign of ϵ_{i+1}.

If in fact there is correlation among the error terms, then the estimated standard errors will tend to underestimate the true standard errors.

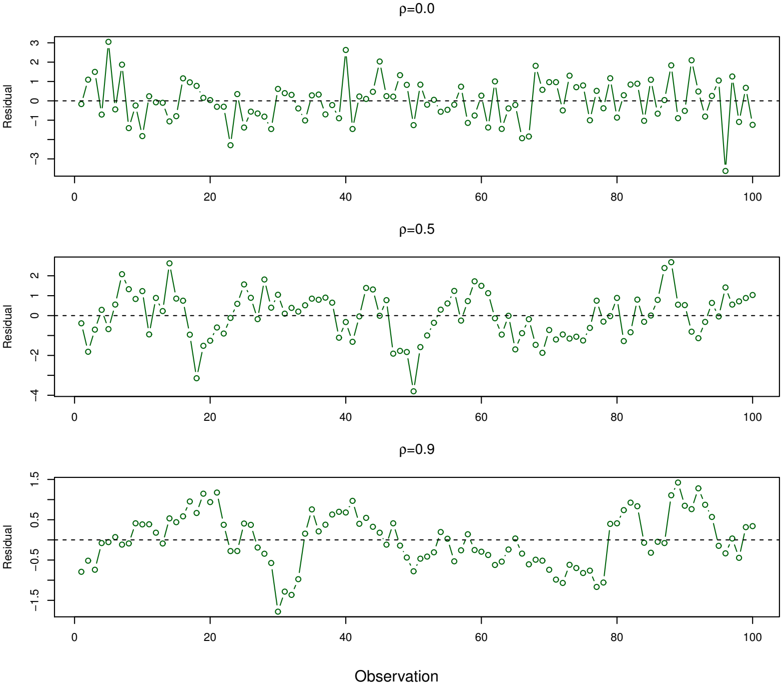
As a result, confidence and prediction intervals will be narrower than they should be, which will lead to wrong conclusions.

In order to determine if this is the case for a given data set, we can plot the residuals from our model as a function of time.

If the errors are uncorrelated, then there should be no discernible pattern.

If the error terms are positively correlated, then we may see tracking in the residuals—that is, adjacent residuals may have tracking similar values.

The next plots show residuals from simulated time series data sets generated with differing levels of correlation ρ between error terms for adjacent time points.

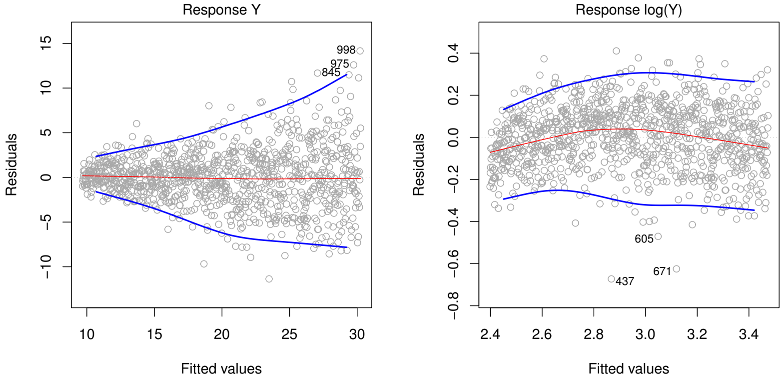


ρ=0.0
ρ=0.5
ρ=0.9
Residual
Observation

Heteroskedasticity

Another important assumption of the linear regression model is that the error terms have a constant variance, i.e. $Var(\epsilon_i) = \sigma^2$
The standard errors, confidence intervals, and hypothesis tests associated with the linear model rely upon this assumption.
Unfortunately, it is often the case case that the variances of the error terms are non-constant.
For instance, the variances of the error terms may increase with the value of the response.

We can identify non-constant variances in the errors, or **heteroskedasticity**, from the presence of a funnel shape in the residual plot.



A good practice in regression analysis is to always use heteroskedasticity robust standard errors.

Use the function `coeftest()` from the package `lmtest` and adjust your standard errors for potentially non-constant variance of the error terms.

```r
adformula <- formula(sales~TV+radio+newspaper)
lm1 <- lm(adformula, data = data)
summary(lm1)$coefficients[,1:3]
```

```
             Estimate Std. Error t value
(Intercept)  2.93889    0.31191   9.422
TV           0.04576    0.00139  32.809
radio        0.18853    0.00861  21.893
newspaper   -0.00104    0.00587  -0.177
```

```r
coeftest(lm1, vcov = vcovHC(lm1, type = "HC0"))[,1:3]
```

```
             Estimate Std. Error t value
(Intercept)  2.93889    0.33310   8.823
TV           0.04576    0.00190  24.141
radio        0.18853    0.01073  17.574
newspaper   -0.00104    0.00636  -0.163
```

Alternatively, just set `robust=HC0` in `summ()`

```r
summ(lm1,robust="HC0", pvals=FALSE,digits=4,
          model.info=FALSE, model.fit=FALSE)
```

```
Standard errors: Robust, type = HC0
----------------------------------------------
                        Est.     S.E.    t val.
------------------ --------- -------- ---------
(Intercept)           2.9389   0.3339    8.8006
TV                    0.0458   0.0019   24.0806
radio                 0.1885   0.0108   17.5296
newspaper            -0.0010   0.0064   -0.1628
----------------------------------------------
```

Outliers

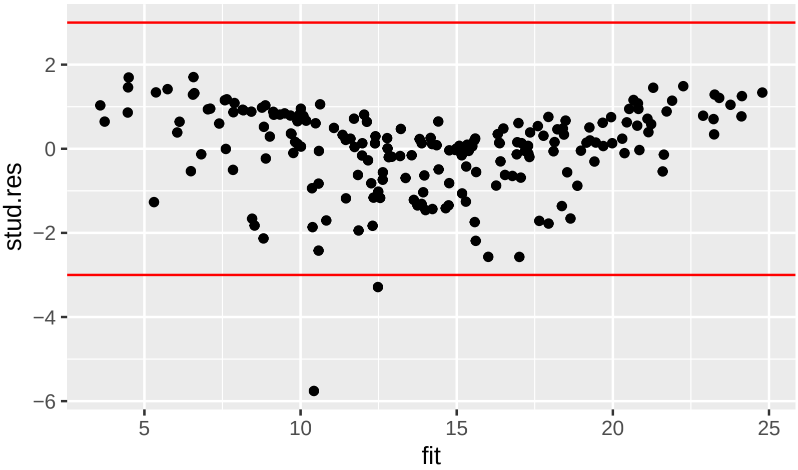
An outlier is a point for which Y_i is far from the value predicted by the model.

To detect outliers compute the studentized residuals, i.e. residuals divided by their standard errors.

Observations whose studentized residuals are greater than 3 in absolute value are possible outliers.

Use the `studres()` function from the `MASS` package to calculate the studentized residuals for each observation in the dataset.

```
fit <- fitted(lm1)
stud.res <- studres(lm1)
stud.fit <- data.frame("fit"=fit,"stud.res"=stud.res)
ggplot(stud.fit, mapping = aes(x=fit,y=stud.res))+
  geom_point()
```


2
0
−2
−4
−6
5
10
15
20
25
stud.res
fit

Clearly we have one outlier. Let's remove that and check how it affects the fit.

```
index <- which.min(stud.res)
summary(lm(adformula, data = data))$sigma
```

```
[1] 1.69
```

```
summary(lm(adformula, data = data))$r.squared
```

```
[1] 0.897
```

```
summary(lm(adformula, data = data[-index,]))$sigma
```

```
[1] 1.56
```

```
summary(lm(adformula, data = data[-index,]))$r.squared
```

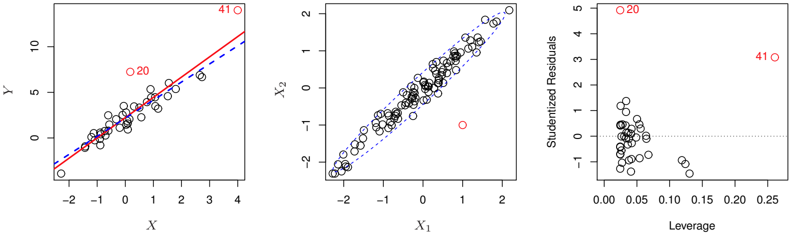
```
[1] 0.91
```

Removing the single outlier improved both the RSE and R^2

High leverage points

An observation is considered to have high leverage if it has a value for the predictors that are much more extreme compared to the rest of the observations in the data.

High leverage points could have a large impact on the results of a given model.



Left: The red line is the fit to all the data, and the blue line is the fit with observation 41 removed.

Center: The red observation is not unusual in terms of its X_1 value or its X_2 value, but still falls outside the bulk of the data, and hence has high leverage.

Right: Observation 41 has a high leverage and a high residual

The leverage statistic is a measure of the distance between the X value for the i_{th} data point and the mean of the X values for all n data points.

The leverage statistic is a number between $1/n$ and 1, inclusive.

The average leverage for all the observations is always equal to $(p+1)/n$.

As a rule of thumb, we say an observation has high leverage if its leverage is greater than 2 times the average leverage.

Use the `hatvalues()` function to calculate the leverage statistic observation in the dataset.

```
leverage <- hatvalues(lm1)
stud.res <- studres(lm1)
stud.lev <- data.frame("leverage"=leverage,
                       "stud.res"=stud.res)
```

```
ggplot(stud.fit, mapping = aes(x=leverage,y=stud.res))+
  geom_point()
```

You can find high leverage points and look at them more carefully

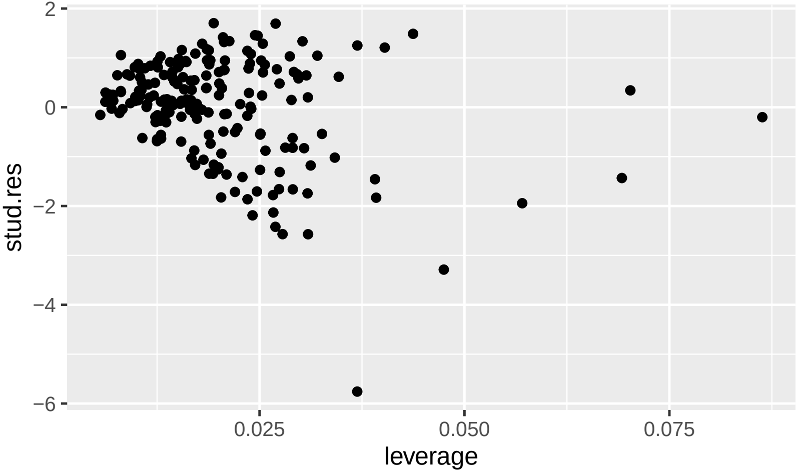
```
high.lev <- which(leverage>(2*mean(leverage)))
unname(high.lev)
```

```
[1]   6  17  37  76 102 129 166
```

```
data[high.lev,]
```

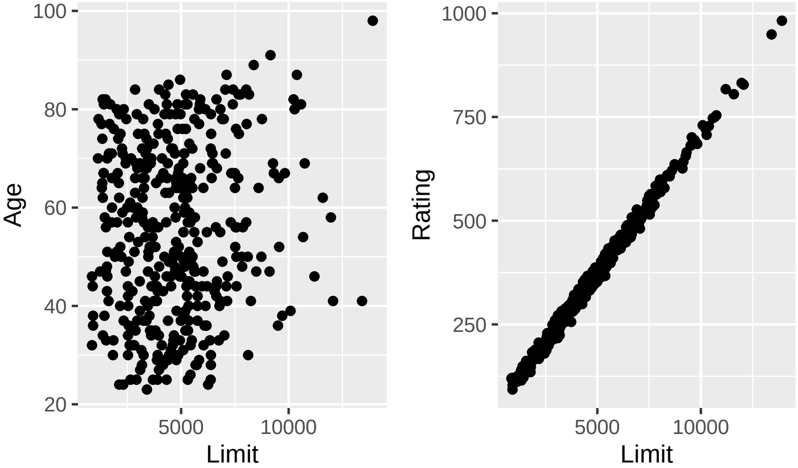
	TV	radio	newspaper	sales
6	8.7	48.9	75.0	7.2
17	67.8	36.6	114.0	12.5
37	266.9	43.8	5.0	25.4
76	16.9	43.7	89.4	8.7
102	296.4	36.3	100.9	23.8
129	220.3	49.0	3.2	24.7
166	234.5	3.4	84.8	11.9

You can also plot them against the studentized residuals to identify particularly problematic cases



Collinearity

Collinearity refers to the situation in which two or more predictor variables are closely related to one another.



No collinearity vs. high collinearity.

Collinearity can be problematic in regression because it makes it difficult to tease out the effects of collinear variables on the response.

Standard errors: OLS

	Est.	S.E.	t val.	p
(Intercept)	-173.41	43.83	-3.96	0.00
Age	-2.29	0.67	-3.41	0.00
Limit	0.17	0.01	34.50	0.00

Standard errors: OLS

	Est.	S.E.	t val.	p
(Intercept)	-377.54	45.25	-8.34	0.00
Rating	2.20	0.95	2.31	0.02
Limit	0.02	0.06	0.38	0.70

To avoid such a situation, it is desirable to identify and address potential collinearity problems while fitting the model.

1. Look at the correlation matrix
2. Compute the `variance inflation factor (VIF)`

The VIF for each variable can be computed using:

$VIF(\hat{\beta}_j) = \frac{1}{1-R^2_{X_j|X_{j-1}}}$, where

$R^2_{X_j|X_{j-1}}$ is the R^2 from a regression of X_j onto all of the other predictors.

If $R^2_{X_j|X_{j-1}}$ is close to one, then collinearity is present, and the `VIF` will be large.

`VIF` value that exceeds 5 indicates a problematic amount of collinearity.

Use the `vif()` function from the car package to compute `VIF` after fitting a model.

```
balance.lm1 <- lm(Balance~Rating+Limit+Age,data=credit)
vif(balance.lm1)
```

```
Rating  Limit    Age
160.67 160.59   1.01
```

```
balance.lm2 <- lm(Balance~Limit+Age,data=credit)
vif(balance.lm2)
```

```
Limit   Age
 1.01  1.01
```

Alternatively, set `vifs=TRUE` in `summ()`

```
summ(balance.lm1,vifs=TRUE,model.info=FALSE,
                model.fit=FALSE,pvals=FALSE)
```

```
Standard errors: OLS
-----------------------------------------------------
                        Est.    S.E.   t val.     VIF
----------------- --------- ------- -------- --------
(Intercept)          -259.52   55.88    -4.64
Rating                  2.31    0.94     2.46   160.67
Limit                   0.02    0.06     0.30   160.59
Age                    -2.35    0.67    -3.51     1.01
-----------------------------------------------------
```

```
summary(balance.lm1)$r.squared
```

```
[1] 0.754
```

```
summary(balance.lm2)$r.squared
```

```
[1] 0.75
```

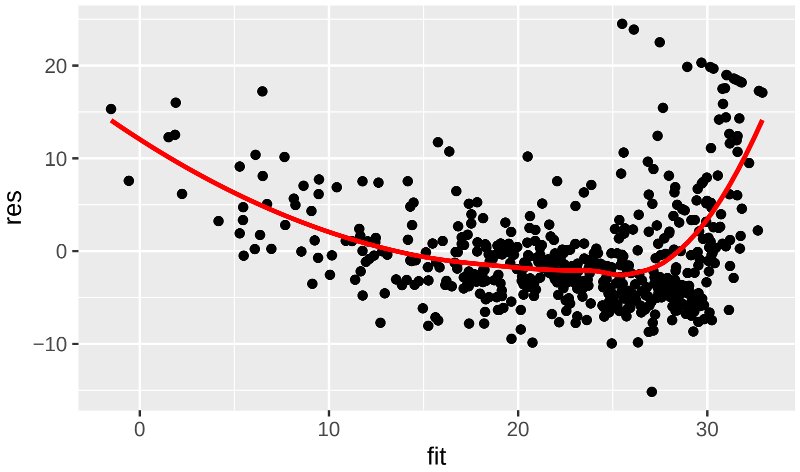
Dropping `rating` from the set of predictors has effectively solved the collinearity problem without compromising the fit.

Exercise

1. Read in the `Boston` data set and run a regresion of `medv` against `lstat`.
2. Plot the residuals against the fitted values and check whether your linearity assumption holds.

```
boston.lm <- lm(medv~lstat,data=boston)
res <- resid(boston.lm)
fit <- fitted(boston.lm)
resfit <- data.frame("res" = res, "fit" = fit)
```

```
ggplot(resfit, mapping = aes(x=fit, y=res)) +
      geom_point()+
      geom_smooth(method = "loess",
                  se=FALSE, colour="red")
```



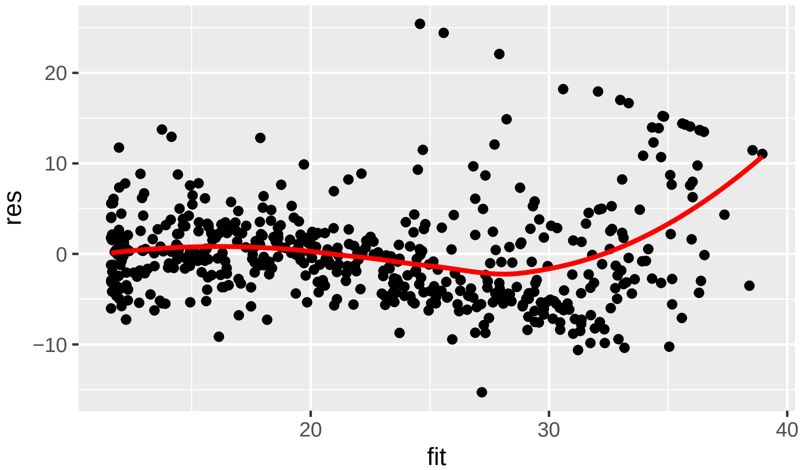
3. Now, include $lstat^2$ in the model and check if it is significant and whether it corrects the violation.

```
boston.lm2 <- lm(medv~poly(lstat,2),data=boston)
summ(boston.lm2, model.info = FALSE, model.fit=FALSE)
```

```
Standard errors: OLS
------------------------------------------------------------
                               Est.    S.E.    t val.      p
--------------------- --------- ------ -------- ------
(Intercept)                   22.53    0.25     91.76   0.00
poly(lstat, 2)1             -152.46    5.52    -27.60   0.00
poly(lstat, 2)2               64.23    5.52     11.63   0.00
------------------------------------------------------------
```

```
res <- resid(boston.lm2)
fit <- fitted(boston.lm2)
resfit <- data.frame("res" = res, "fit" = fit)
```

```r
ggplot(resfit, mapping = aes(x=fit, y=res)) +
      geom_point()+
      geom_smooth(method = "loess",
                  se=FALSE, colour="red")
```



4. Compare the R^2 values and test whether the quadratic model is justified.

```
summary(boston.lm)$r.squared
```

```
[1] 0.544
```

```
summary(boston.lm2)$r.squared
```

```
[1] 0.641
```

```
anova(boston.lm,boston.lm2)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
504	19472	NA	NA	NA	NA
503	15347	1	4125	135	0