Linear Regression in R

Jan-Philipp Kolb

24 Februar, 2020

Variables of the mtcars dataset

Help for the `mtcars` dataset:

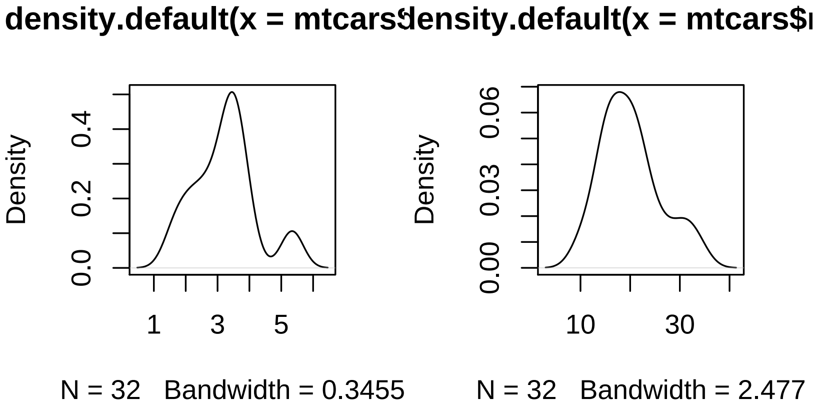
```
?mtcars
```

- mpg - Miles/(US) gallon
- cyl - Number of cylinders
- disp - Displacement (cu.in.)
- hp - Gross horsepower
- drat - Rear axle ratio
- wt - Weight (1000 lbs)
- qsec - 1/4 mile time
- vs - Engine (0 = V-shaped, 1 = straight)
- am - Transmission (0 = automatic, 1 = manual)
- gear - Number of forward gears
- carb - Number of carburetors

Dataset mtcars

	mpg	cyl	disp	hp	drat	wt	qsec	vs	ar
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	

```
par(mfrow=c(1,2))
plot(density(mtcars$wt)); plot(density(mtcars$mpg))
```



A simple regression model

Dependent variable - miles per gallon (mpg)

Independent variable - weight (wt)

```
m1 <- lm(mpg ~ wt,data=mtcars)
m1
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Coefficients:
## (Intercept)           wt
##      37.285       -5.344
```

Get the model summary

```r
summary(m1)
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
##
## Residual standard error: 3.046 on 30 degrees of freedom
## Multiple R-squared:  0.7528, Adjusted R-squared:  0.7446
## F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

The model formula

Model without intercept

```
m2 <- lm(mpg ~ - 1 + wt,data=mtcars)
summary(m2)$coefficients
```

```
##    Estimate Std. Error  t value     Pr(>|t|)
## wt 5.291624  0.5931801 8.920771 4.55314e-10
```

Adding further variables

```
m3 <- lm(mpg ~ wt + cyl,data=mtcars)
summary(m3)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 39.686261  1.7149840 23.140893 3.043182e-20
## wt          -3.190972  0.7569065 -4.215808 2.220200e-04
## cyl         -1.507795  0.4146883 -3.635972 1.064282e-03
```

Estimation based on a subsample

```r
lm(mpg~wt+disp, data=mtcars, subset=(wt>3))
```

```
##
## Call:
## lm(formula = mpg ~ wt + disp, data = mtcars, subset = (wt > 3
##
## Coefficients:
## (Intercept)           wt         disp
##    28.40497     -1.46360     -0.02016
```

- where only cars heavier than 3000 lb are considered.

The command as.formula

Creating a formula object

```
?as.formula
```

```
fo <- mpg ~ wt + cyl
```

```
class(fo)
```

```
## [1] "formula"
```

```
# The formula object can be used in the regression:
m3 <- lm(fo,data=mtcars)
```

Further possibilities to specify the formula

Take all available predictors

```
m3_a<-lm(mpg~.,data=mtcars)
```

Interaction effect

```
# effect of cyl and interaction effect:
m3a<-lm(mpg~wt*cyl,data=mtcars)

# only interaction effect:
m3b<-lm(mpg~wt:cyl,data=mtcars)
```

Take the logarithm

```
m3d<-lm(mpg~log(wt),data=mtcars)
```

Further transformations

Further transformations:

Tranformations of variables are directly included with the *I*() function:

```
fo2 <- I(log(mpg))~wt+I(wt^2)+disp
lm(fo2, data=mtcars)
```

```
##
## Call:
## lm(formula = fo2, data = mtcars)
##
## Coefficients:
## (Intercept)           wt      I(wt^2)         disp
##   4.0000825   -0.3499056    0.0275548   -0.0009865
```

The command setdiff

- We can use the command to create a dataset with only the features, without the dependent variable

```
names(mtcars)
```

```
##  [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"
## [11] "carb"
```

```
features <- setdiff(names(mtcars), "mpg")
features
```

```
##  [1] "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"
```

```
featdat <- mtcars[,features]
```

The command model.matrix

- With `model.matrix` the qualitative variables are automatically dummy encoded

```
?model.matrix
```

```
model.matrix(m3d)
```

```
##                     (Intercept)   log(wt)
## Mazda RX4                     1 0.9631743
## Mazda RX4 Wag                 1 1.0560527
## Datsun 710                    1 0.8415672
## Hornet 4 Drive                1 1.1678274
## Hornet Sportabout             1 1.2354715
## Valiant                       1 1.2412686
## Duster 360                    1 1.2725656
## Merc 240D                     1 1.1600209
## Merc 230                      1 1.1474025
## Merc 280                      1 1.2354715
## Merc 280C                     1 1.2354715
## Merc 450SE                    1 1.4036430
## Merc 450SL                    1 1.3164082
```

Model matrix (II)

- We can also create a model matrix directly from the formula and data arguments
- See `Matrix::sparse.model.matrix` for increased efficiency on large dimension data.

```
ff <- mpg ~ log(wt):cyl
m <- model.frame(ff, mtcars)
```

```
(mat <- model.matrix(ff, m))
```

```
##                     (Intercept) log(wt):cyl
## Mazda RX4                     1    5.779046
## Mazda RX4 Wag                 1    6.336316
## Datsun 710                    1    3.366269
## Hornet 4 Drive                1    7.006964
## Hornet Sportabout             1    9.883772
## Valiant                       1    7.447612
## Duster 360                    1   10.180525
## Merc 240D                     1    4.640084
## Merc 230                      1    4.589610
## Merc 280                      1    7.412829
```

A model with interaction effect

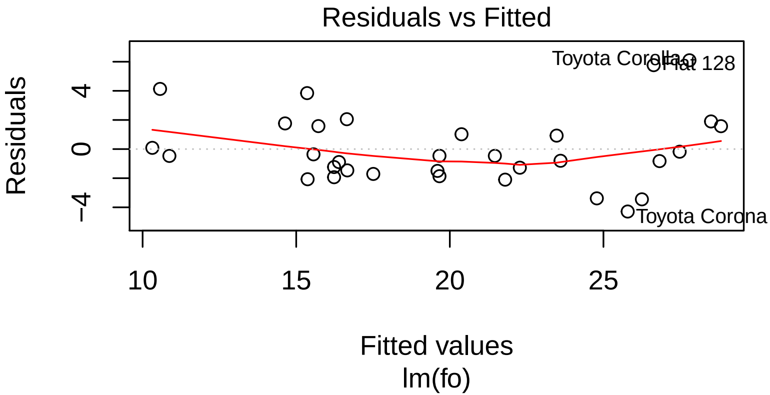
```r
# disp  -  Displacement (cu.in.)
m3d<-lm(mpg~wt*disp,data=mtcars)
m3dsum <- summary(m3d)
m3dsum$coefficients
```

```
##                Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept) 44.08199770 3.123062627 14.114990 2.955567e-14
## wt          -6.49567966 1.313382622 -4.945763 3.216705e-05
## disp        -0.05635816 0.013238696 -4.257078 2.101721e-04
## wt:disp      0.01170542 0.003255102  3.596022 1.226988e-03
```

Residual plot - model assumptions violated?

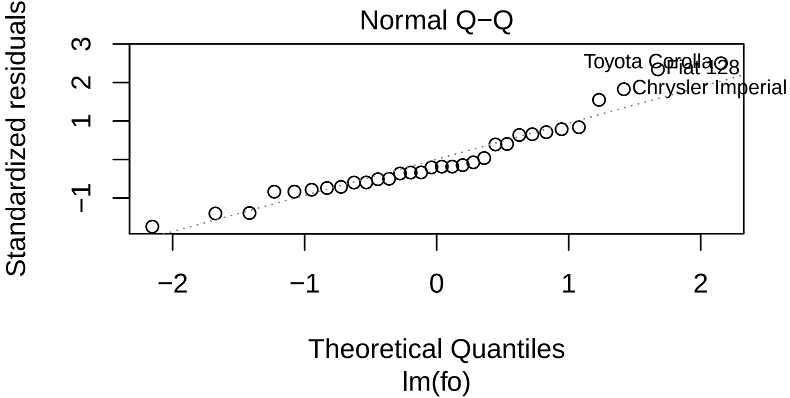
- We have model assumptions violated if points deviate with a pattern from the line

```
plot(m3,1)
```



Residual plot

```
plot(m3,2)
```



- If the residuals are normally distributed, they should be on the same line.

- m3 is now a special regression object
- Various functions can be applied to this object

```
predict(m3) # Prediction
resid(m3) # Residuals
```

```
##          Mazda RX4       Mazda RX4 Wag          Datsun 710     Horn
##           22.27914            21.46545            26.25203
## Hornet Sportabout             Valiant
##           16.64696            19.59873
```

```
##          Mazda RX4       Mazda RX4 Wag          Datsun 710     Horn
##         -1.2791447          -0.4654468          -3.4520262
## Hornet Sportabout             Valiant
##          2.0530424          -1.4987281
```

Make model prediction

```
pre <- predict(m1)
head(mtcars$mpg)
```

```
## [1] 21.0 21.0 22.8 21.4 18.7 18.1
```

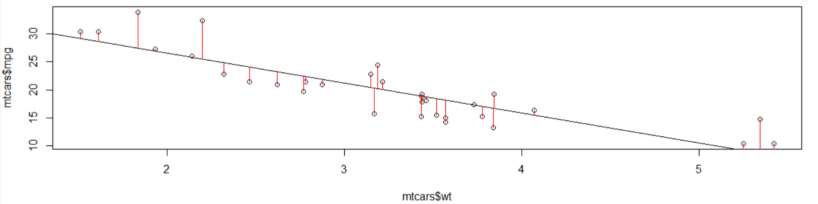
```
head(pre)
```

```
##         Mazda RX4     Mazda RX4 Wag        Datsun 710    Horn
##          23.28261          21.91977          24.88595
## Hornet Sportabout           Valiant
##          18.90014          18.79325
```

Regression diagnostic with base-R

Visualizing residuals

```
plot(mtcars$wt,mtcars$mpg)
abline(m1)
segments(mtcars$wt, mtcars$mpg, mtcars$wt, pre, col="red")
```



The bias-variance tradeoff (I)

The bias–variance tradeoff is the property of a set of predictive models whereby models with a lower bias in parameter estimation have a higher variance of the parameter estimates across samples, and vice versa.

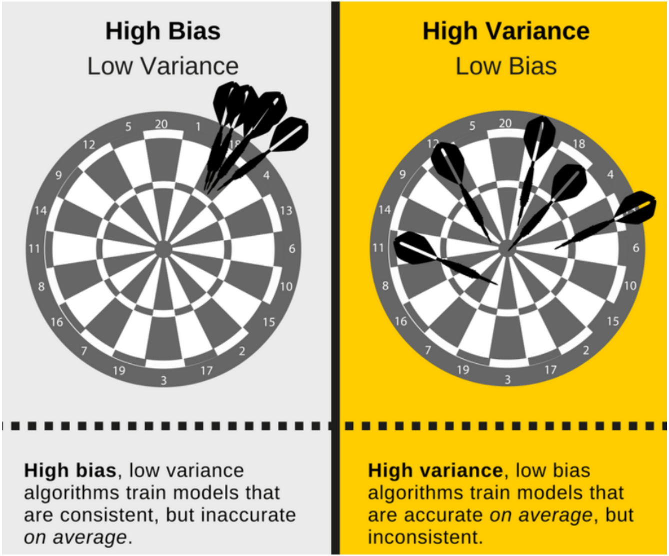
The bias error

... is an error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).

The variance

... is an error from sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

The bias-variance tradeoff (II)



The mean squared error (MSE)

- The **MSE** measures the average of the squares of the errors
- **The lower the better**

```
(mse5 <- mean((mtcars$mpg -  pre)^2)) # model 5
```

```
## [1] 8.697561
```

```
(mse3 <- mean((mtcars$mpg -  predict(m3))^2))
```

```
## [1] 5.974124
```

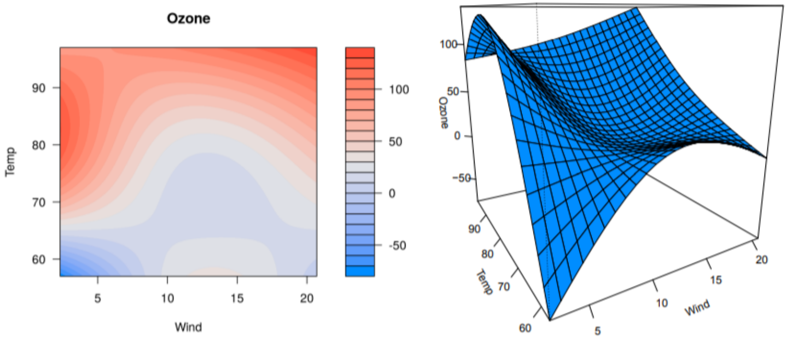
Package Metrics to compute MSE

```
library(Metrics)
mse(mtcars$mpg,predict(m3))
```

```
## [1] 5.974124
```

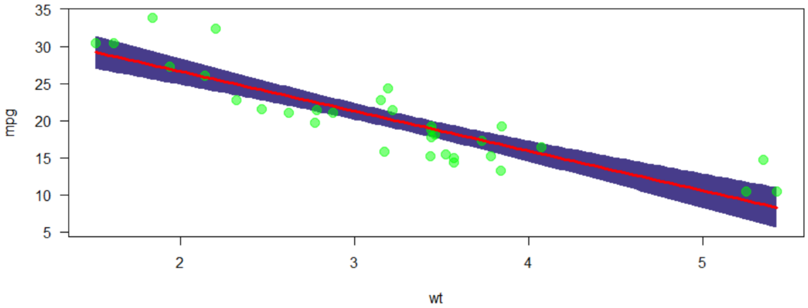
```
install.packages("visreg")
install.packages("Metrics")
```

```
library(visreg)
```



- The default-argument for type is conditional.
- Scatterplot of mpg and wt plus regression line and confidence bands

```
visreg(m1, "wt", type = "conditional")
```



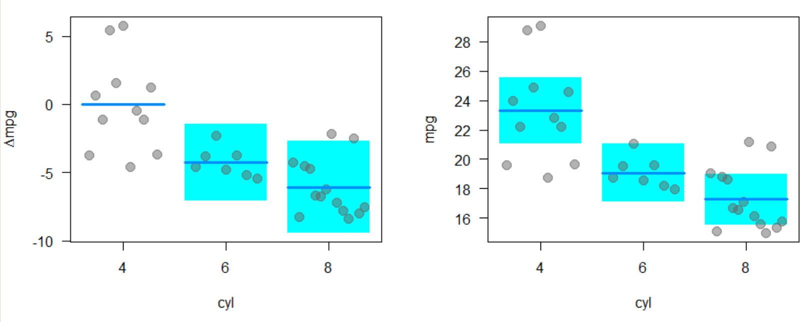
- The effects of factors can also be visualized with `visreg`:

```
mtcars$cyl <- as.factor(mtcars$cyl)
m4 <- lm(mpg ~ cyl + wt, data = mtcars)
# summary(m4)
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 33.990794  1.8877934 18.005569 6.257246e-17
## cyl6        -4.255582  1.3860728 -3.070244 4.717834e-03
## cyl8        -6.070860  1.6522878 -3.674214 9.991893e-04
## wt          -3.205613  0.7538957 -4.252065 2.130435e-04
```

Effects of factors

```
par(mfrow=c(1,2))
visreg(m4, "cyl", type = "contrast")
visreg(m4, "cyl", type = "conditional")
```

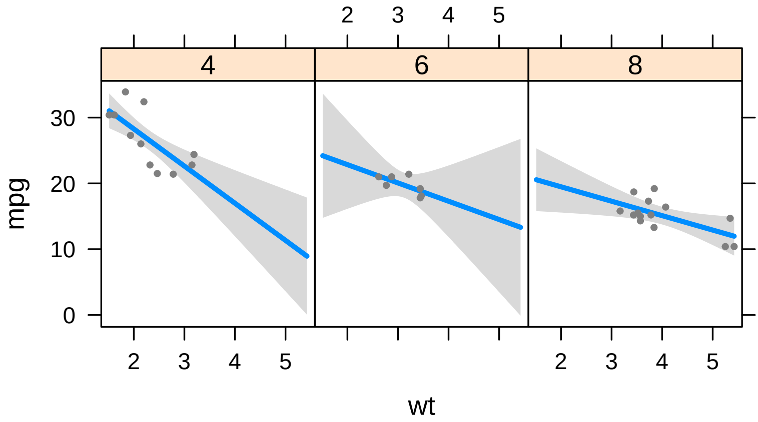


The package visreg - Interactions

```r
m5 <- lm(mpg ~ cyl*wt, data = mtcars)
# summary(m5)
```

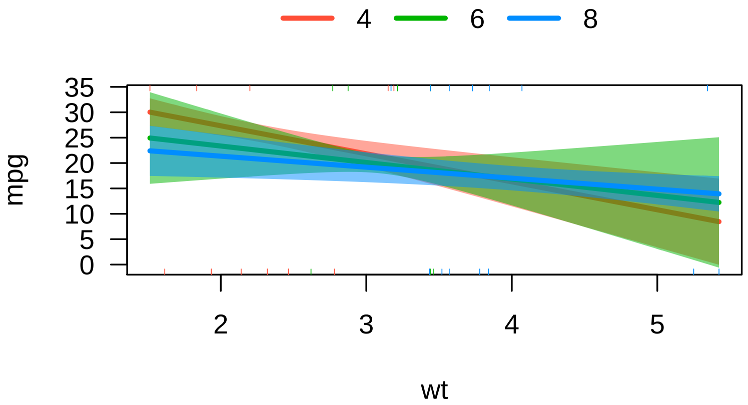
```
##                Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  39.571196   3.193940 12.3894599 2.058359e-12
## cyl6        -11.162351   9.355346 -1.1931522 2.435843e-01
## cyl8        -15.703167   4.839464 -3.2448150 3.223216e-03
## wt           -5.647025   1.359498 -4.1537586 3.127578e-04
## cyl6:wt       2.866919   3.117330  0.9196716 3.661987e-01
## cyl8:wt       3.454587   1.627261  2.1229458 4.344037e-02
```

```
visreg(m5, "wt", by = "cyl",layout=c(3,1))
```

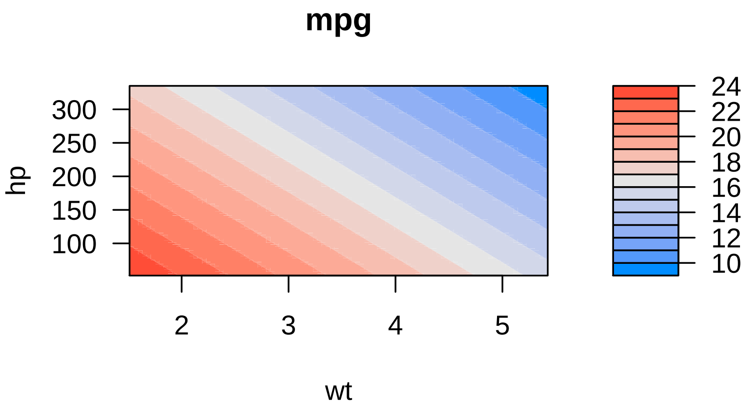


```
m6 <- lm(mpg ~ hp + wt * cyl, data = mtcars)
```

```
visreg(m6, "wt", by="cyl", overlay=TRUE, partial=FALSE)
```



```
visreg2d(m6, "wt", "hp", plot.type = "image")
```



Exercise: regression Ames housing data

1. Install the package `AmesHousing` and create a **processed version** of the Ames housing data with (at least) the variables `Sale_Price`, `Gr_Liv_Area` and `TotRms_AbvGrd`
2. Create a regression model with `Sale_Price` as dependent and `Gr_Liv_Area` and `TotRms_AbvGrd` as independent variables. Then create seperated models for the two independent variables. Compare the results. What do you think?

```r
ames_data <- AmesHousing::make_ames()
```

Some Variables

- `Gr_Liv_Area`: Above grade (ground) living area square feet
- `TotRms_AbvGrd`: Total rooms above grade (does not include bathrooms
- `MS_SubClass`: Identifies the type of dwelling involved in the sale.
- `MS_Zoning`: Identifies the general zoning classification of the sale.
- `Lot_Frontage`: Linear feet of street connected to property
- `Lot_Area`: Lot size in square feet
- `Street`: Type of road access to property
- `Alley`: Type of alley access to property
- `Lot_Shape`: General shape of property
- `Land_Contour`: Flatness of the propert

- As p increases we are more likely to capture multiple features that have some multicollinearity.
- When multicollinearity exists, we often see high variability in our coefficient terms.
- E.g. we have a correlation of 0.801 between `Gr_Liv_Area` and `TotRms_AbvGrd`
- Both variables are strongly correlated to the response variable (`Sale_Price`).

```
ames_data <- AmesHousing::make_ames()
cor(ames_data[,c("Sale_Price","Gr_Liv_Area","TotRms_AbvGrd")])
```

```
##               Sale_Price Gr_Liv_Area TotRms_AbvGrd
## Sale_Price     1.0000000   0.7067799     0.4954744
## Gr_Liv_Area    0.7067799   1.0000000     0.8077721
## TotRms_AbvGrd  0.4954744   0.8077721     1.0000000
```

```r
lm(Sale_Price ~ Gr_Liv_Area + TotRms_AbvGrd, data = ames_data)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Gr_Liv_Area + TotRms_AbvGrd, data =
##
## Coefficients:
##   (Intercept)    Gr_Liv_Area  TotRms_AbvGrd
##       42767.6          139.4       -11025.9
```

- When we fit a model with both these variables we get a positive coefficient for `Gr_Liv_Area` but a negative coefficient for `TotRms_AbvGrd`, suggesting one has a positive impact to Sale_Price and the other a negative impact.

- If we refit the model with each variable independently, they both show a positive impact.
- The `Gr_Liv_Area` effect is now smaller and the `TotRms_AbvGrd` is positive with a much larger magnitude.

```
lm(Sale_Price ~ Gr_Liv_Area, data = ames_data)$coefficients
```

```
## (Intercept) Gr_Liv_Area
##   13289.634     111.694
```

```
lm(Sale_Price ~ TotRms_AbvGrd, data = ames_data)$coefficients
```

```
##   (Intercept) TotRms_AbvGrd
##      18665.40      25163.83
```

- This is a common result when collinearity exists.
- Coefficients for correlated features become over-inflated and can fluctuate significantly.

Consequences

- One consequence of these large fluctuations in the coefficient terms is **overfitting**, which means we have high variance in the bias-variance tradeoff space.
- We can use tools such as **variance inflaction factors** (Myers, 1994) to identify and remove those strongly correlated variables, but it is not always clear which variable(s) to remove.
- Nor do we always wish to remove variables as this may be removing signal in our data.

Links - linear regression

- Regression - **r-bloggers**
- The complete book of **Faraway**- very intuitive
- Good introduction on **Quick-R**
- **Multiple regression**
- **15 Types of Regression you should know**
- `ggeffects` - **Create Tidy Data Frames of Marginal Effects for 'ggplot' from Model Outputs**
- **Machine learning iteration**

Nice table output with stargazer

```r
library(stargazer)
stargazer(m3, type="html")
```

Example HTML output:

	Dependent variable:
	mpg
wt	-3.125***
	(0.911)
cyl	-1.510***
	(0.422)
am	0.176
	(1.304)
Constant	39.418***
	(2.641)
Observations	32
R^2	0.830

Shiny App - Diagnostics for linear regression

- Shiny App - **Simple Linear Regression**
- Shiny App - **Multicollinearity in multiple regression**