Regression in R

Jan-Philipp Kolb

21 Mai, 2019

Why a part on simple regression

- some machine learning concepts are based on regression
- I would like to remind you how simple regression in R works.
- I also want to show the constraints
- In a next step we will learn, how to coop with that

Variables of the mtcars dataset

Help for the `mtcars` dataset:

```
?mtcars
```

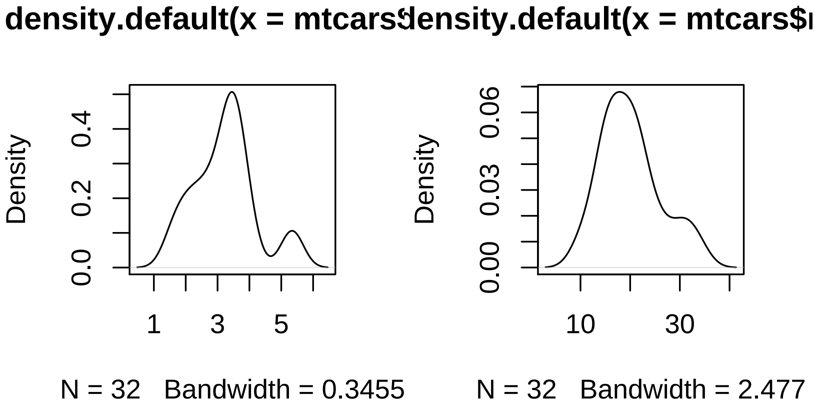
- mpg - Miles/(US) gallon
- cyl - Number of cylinders
- disp - Displacement (cu.in.)
- hp - Gross horsepower
- drat - Rear axle ratio
- wt - Weight (1000 lbs)
- qsec - 1/4 mile time
- vs - Engine (0 = V-shaped, 1 = straight)
- am - Transmission (0 = automatic, 1 = manual)
- gear - Number of forward gears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

Dataset mtcars

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
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			

Distributions of two variables of mtcars

```
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

```
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
```

The model formula

Model without intercept

```r
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

```r
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
```

The command as.formula

```
?as.formula
class(fo <- mpg ~ wt + cyl)
## [1] "formula"
m3 <- lm(fo,data=mtcars)
```

The command `model.matrix`

```
?model.matrix
```

- See `Matrix::sparse.model.matrix` for increased efficiency on large dimension data.

Further possibilities to specify the formula

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)
```

A model with interaction effect

-disp - Displacement (cu.in.)

```
m3d<-lm(mpg~wt*disp,data=mtcars)
m3dsum <- summary(m3d)
m3dsum$coefficients
```

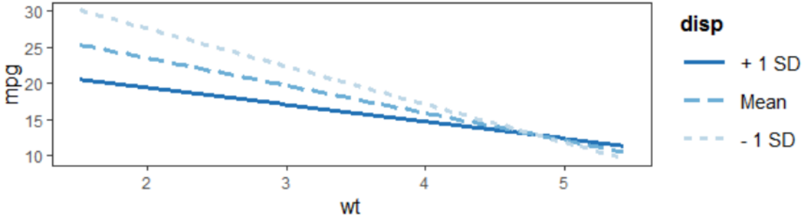
```
##                Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept) 44.08199770 3.123062627 14.114990 2.955567e-14
## wt          -6.49567966 1.313382622 -4.945763 3.216705e-05
## disp        -0.05635816 0.013238696 -4.257078 2.101721e-04
## wt:disp      0.01170542 0.003255102  3.596022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

Exploring interactions

```r
install.packages("jtools")
```

```r
library(jtools)
interact_plot(m3d, pred = "wt", modx = "disp")
```

- With a continuous moderator (in our case `disp`) you get three lines — 1 standard deviation above and below the mean and the mean itself.



Example: object orientation

- 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
```

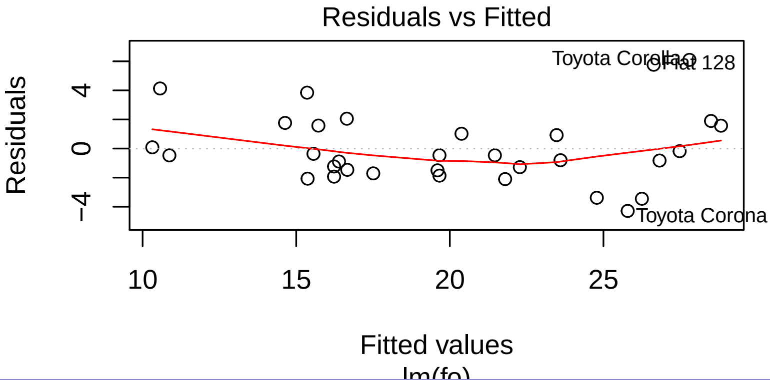
```
head(pre)
```

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

Residual plot - model assumptions violated?

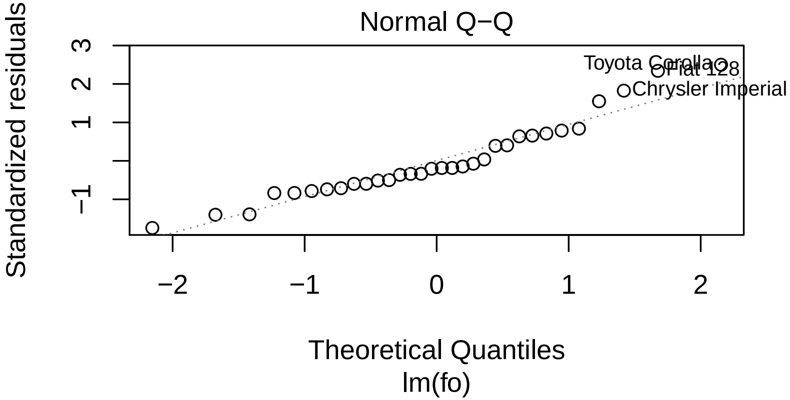
- the case if a deviation pattern from line

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



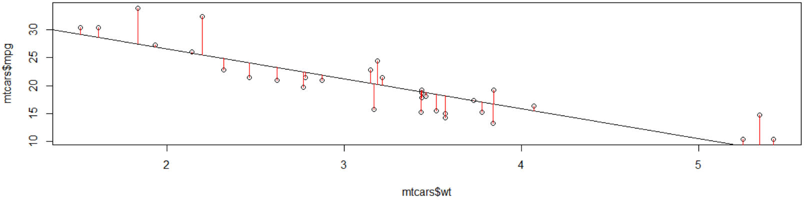
Residual plot

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



Regression diagnostic with base-R

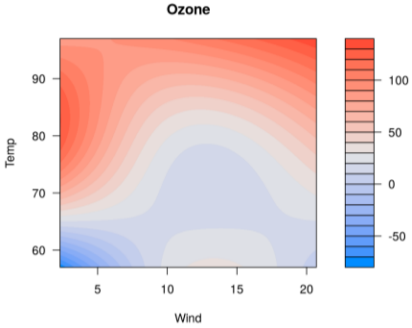
```
plot(mtcars$wt,mtcars$mpg)
abline(m1)
segments(mtcars$wt, mtcars$mpg, mtcars$wt, pre, col="red")
```

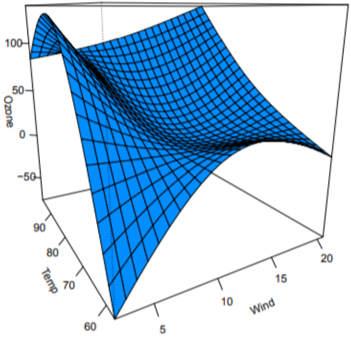


The visreg-package

```
install.packages("visreg")

library(visreg)
```

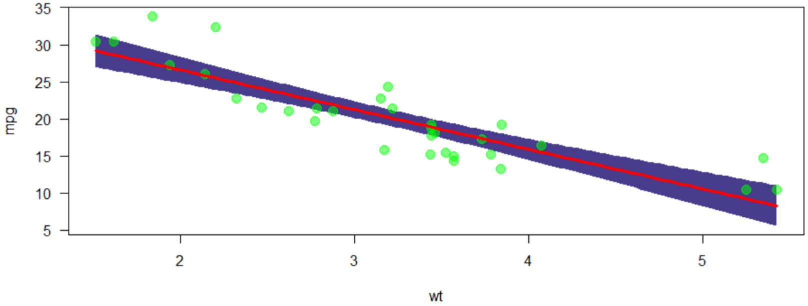




The visreg-package

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

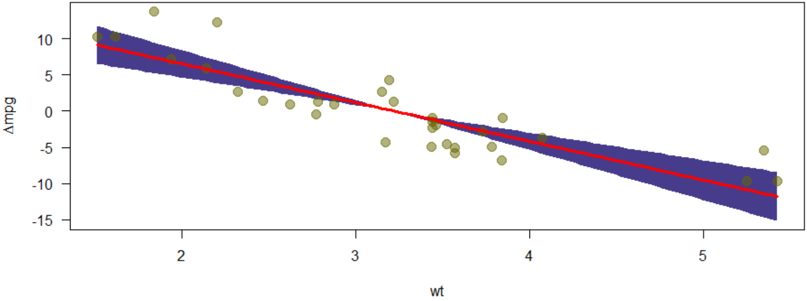
```
visreg(m1, "wt", type = "conditional")
```



Visualisation with visreg

- Second argument - Specification covariate for visualisation
- plot shows the effect on the expected value of the response by moving the x variable away from a reference point on the x-axis (for numeric variables, the mean).

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



Regression with factors

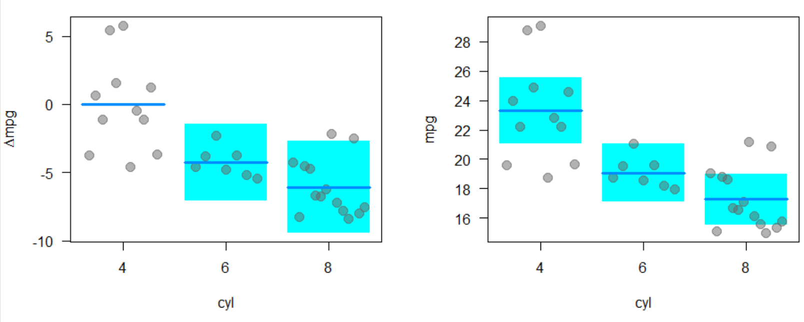
- The effects of factors can also be visualized with `visreg`:

```
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 command model.matrix

```
?model.matrix
```

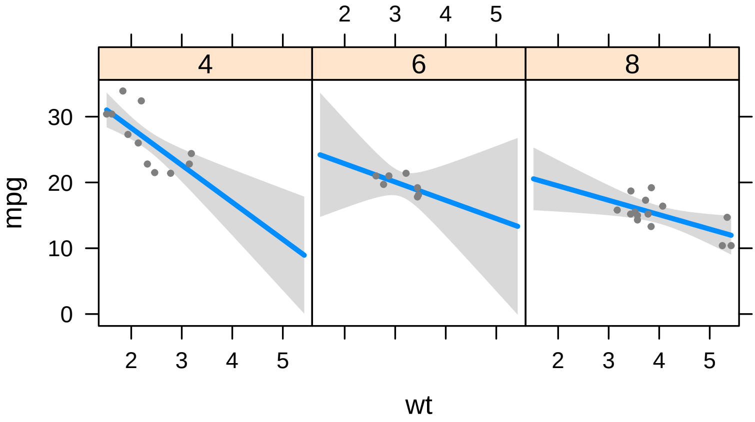
The package visreg - Interactions

```
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
```

Control of the graphic output with layout.

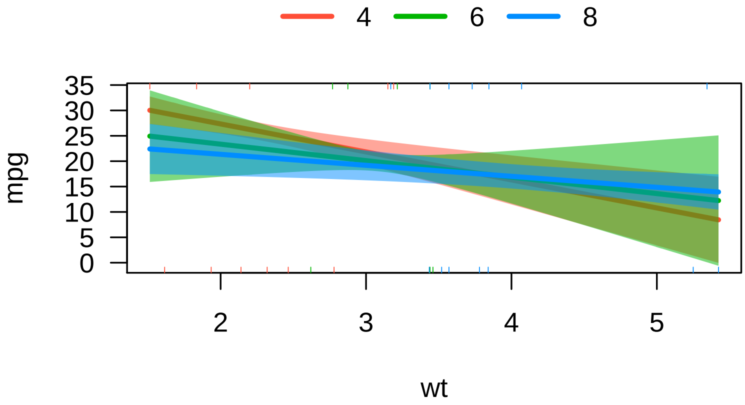
```
visreg(m5, "wt", by = "cyl",layout=c(3,1))
```



The package visreg - Interactions overlay

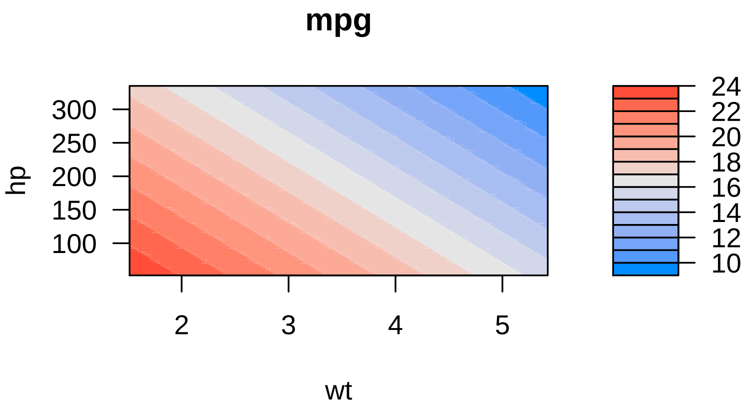
```
m6 <- lm(mpg ~ hp + wt * cyl, data = mtcars)

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



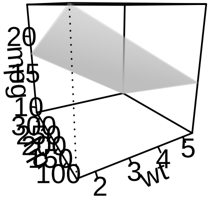
The package visreg - visreg2d

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



The package visreg - surface

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



Producing nice table output with Package 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)

Exercise

- Install the package `AmesHousing` and create a processed Version of the Ames housing data with the variables `Sale_Price`, `Gr_Liv_Area` and `TotRms_AbvGrd`
- Create a suitable regression model with `Sale_Price` as dependent and `Gr_Liv_Area` and `TotRms_AbvGrd` as independent variables. What do you think?

Multicollinearity

- 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`).

```
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
```

Multicollinearity

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

```
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.

Seperated models

- 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
```

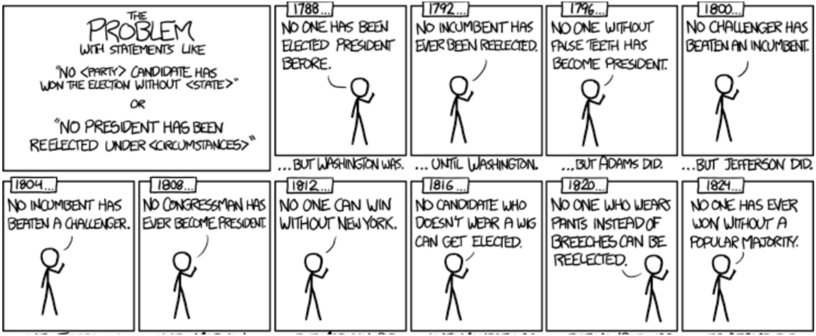
- This is a common result when collinearity exists.
- Coefficients for correlated features become over-inflated and can fluctuate significantly.

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 inflation 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.

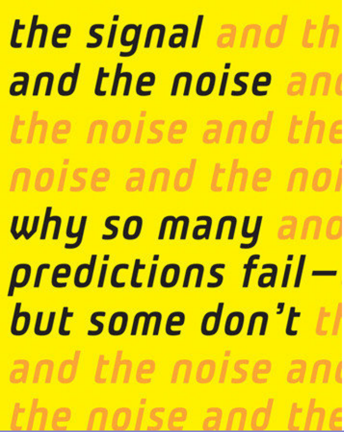
The problem - Overfitting

- Our model doesn't generalize well from our training data to unseen data.



The Signal and the Noise

- In predictive modeling, you can think of the "signal" as the true underlying pattern that you wish to learn from the data.
- "Noise," on the other hand, refers to the irrelevant information or randomness in a dataset.



What can be done against overvitting

- Cross Validation
- Train with more data
- Remove features
- Regularization - e.g. ridge and lasso regression
- Ensembling - e.g. bagging and boosting

Cross-validation

- Cross-validation is a powerful preventative measure against overfitting.
- Use your initial training data to generate multiple mini train-test splits. Use these splits to tune your model.

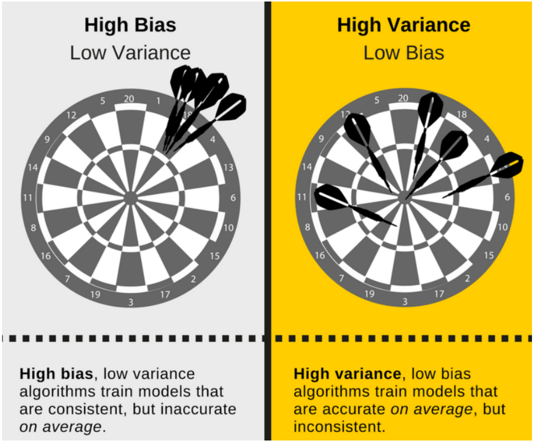
Cross Validation in R

```
library(tidyverse)
library(caret)

data("swiss")

training.samples <- swiss$Fertility %>%
createDataPartition(p = 0.8, list = FALSE)
train.data  <- swiss[training.samples, ]
test.data <- swiss[-training.samples, ]
# Build the model
model <- lm(Fertility ~., data = train.data)
# Make predictions and compute the R2, RMSE and MAE
predictions <- model %>% predict(test.data)
```

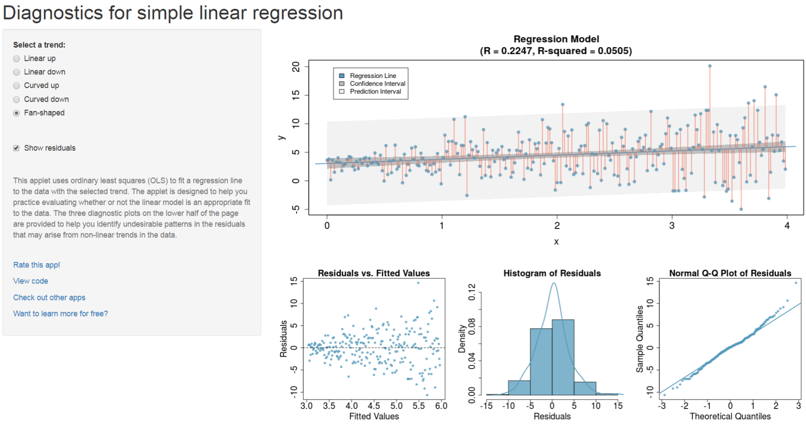
The bias variance tradeoff



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**

Shiny App - Diagnostics for simple linear regression



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