

# Methods 3: Multilevel Statistical Modeling and Machine Learning

Week 6: *Mid-way evaluation and Machine Learning Intro*

November 2, 2021

*by:* Lau Møller Andersen

These slides are distributed according to the CC  
BY 4.0 licence:

<https://creativecommons.org/licenses/by/4.0/>



# **Study Café**

**how is it going?**

**Reminder:**  
**Office hours**  
**(If assignments are hard, why is**  
**no one coming?)**

# Python book in Stakbogladden

<https://www.stakbogladden.dk/soegning.asp?phrase=9781783555130>

Also available online at the  
Royal Library (thanks, Emil!)



BOG

**Python machine learning : unlock deeper insights into machine learning with this vital guide to cutting-edge predictive analytics**

Sebastian Raschka author; Randal S Olson author of foreword  
2015; 1st edition

[Tilgængelig online >](#)

# Practical exercise tomorrow

To make sure that *Python* runs within *R Markdown*, make sure you have the *reticulate* package installed

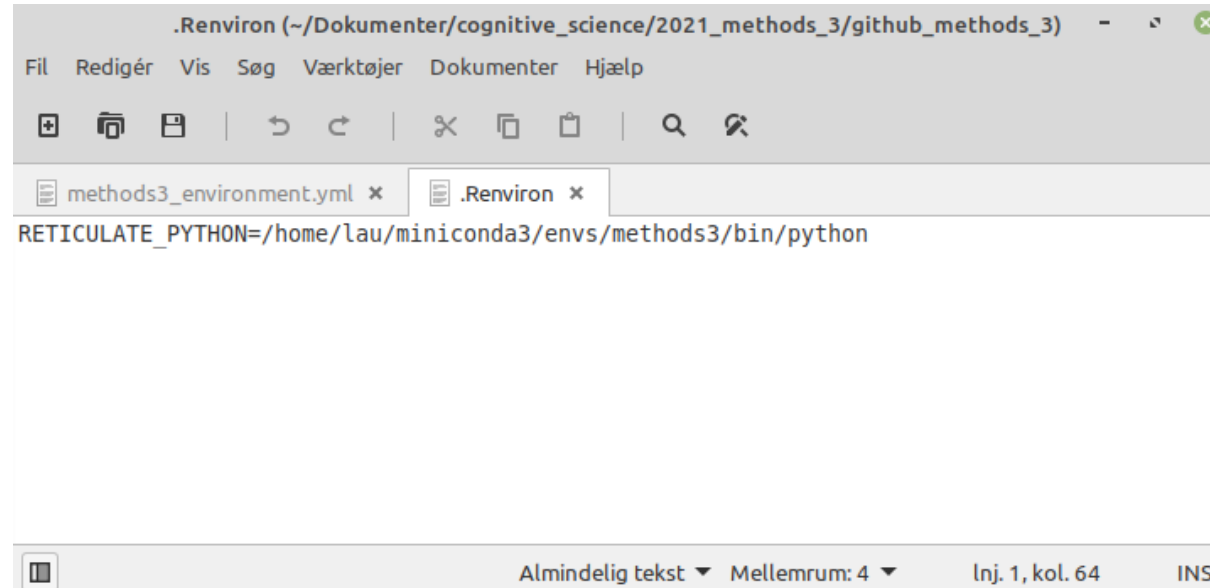
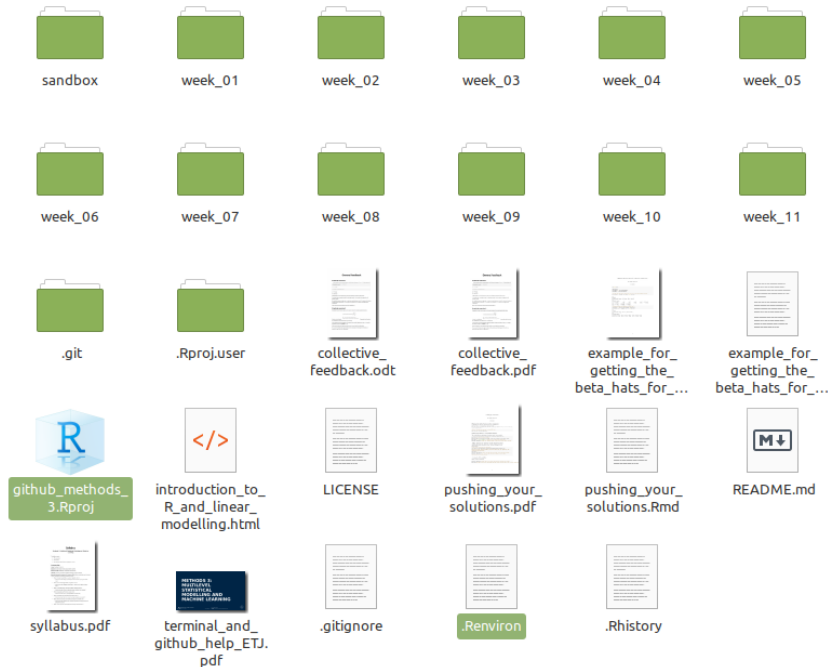
```
install.packages('reticulate')
```

Also create a text file that is called *.Renviron* (remember the dot) placed in the folder where your *RProj* file is. It should have a single line: `RETICULATE_PYTHON=PATH` where `PATH` is the path to your *methods3* conda environment. Use the commands below to find the paths:

```
library(reticulate)
print(conda_list())
```

```
##           name                               python
## 1  miniconda3          /home/lau/miniconda3/bin/python
## 2   methods3    /home/lau/miniconda3/envs/methods3/bin/python
## 3      mne          /home/lau/miniconda3/envs/mne/bin/python
## 4  mne_0.17    /home/lau/miniconda3/envs/mne_0.17/bin/python
## 5 mne_func_sig /home/lau/miniconda3/envs/mne_func_sig/bin/python
## 6   mnedev          /home/lau/miniconda3/envs/mnedev/bin/python
## 7  psychopy    /home/lau/miniconda3/envs/psychopy/bin/python
## 8  fslpython    /usr/local/fsl/fslpython/bin/python
## 9  fslpython /usr/local/fsl/fslpython/envs/fslpython/bin/python
```

# Practical exercise tomorrow



**NB! No spaces around equals sign!**

# Practical exercise tomorrow

## Update environment

```
conda env create --force -f methods3_environment.yml
```

**Overwrite old environment**

Update environment (new packages have been added). Run this command from the folder *week\_06*



# Mid-way evaluation

# Mid-way evaluation

- 1) Write something you liked about the course so far
- 2) Write something you did not like about the course so far
- 3) What would you change?

Next time: I'll summarise the feedback on the three points and what I'll change

# Learning goals

*Evaluating and comparing models*

## 1) Learning tools for comparing models

- 1) Variance explained

- 2) Likelihood ratio tests

- 3) Information criteria

## 2) Bridging to out-of-sample

# Why are we modelling?

Remember Emil's slides from week 03

- To be able to understand the world
- To be able to predict and manipulate the world

$$F = G \frac{m_1 m_2}{r^2}$$

**EXPLANATION**



*NASA/Bill Ingalls*

**PREDICTION**

# What constitutes a good model?

Remember Emil's slides from week 03

- Accurate estimation of the underlying parameters of the population distribution
- Generalisation to new data

**EXPLANATION**

**PREDICTION**

Within an **explanatory** framework, how can we assess whether we have done a good job?

# Variance explained

- Pros
  - $R^2$  is intuitive
- Cons
  - More complex models will always explain more variance
  - Hard to interpret in the case of collinearity
  - $R^2$  doesn't give us what we want

the likelihood of this particular model to be true



# Likelihood ratio

- Pros
  - Models can be compared in a principled way by reference to a theoretical distribution,  $\chi^2$ . (In the single level case,  $F$  can be calculated)
- Cons
  - Models have to be nested in one another can only compare when one of the model is a subset of the other
  - Maximum likelihood fitting may be biased for complex models
  - Requires large sample sizes
  - Be careful if collinearity is high

# Information criteria

- Pros
  - Models can be compared even though one is not nested within the other (response data has to be the same though)
- Cons
  - Number of effective parameters not well defined for multilevel models
  - Maximum likelihood fitting may be biased for complex models

# Did you learn? (it's not easy)

*Evaluating and comparing models*

## 1) Learning tools for comparing models

- 1) Variance explained

- 2) Likelihood ratio tests

- 3) Information criteria

## 2) Bridging to out-of-sample

# Learning goals

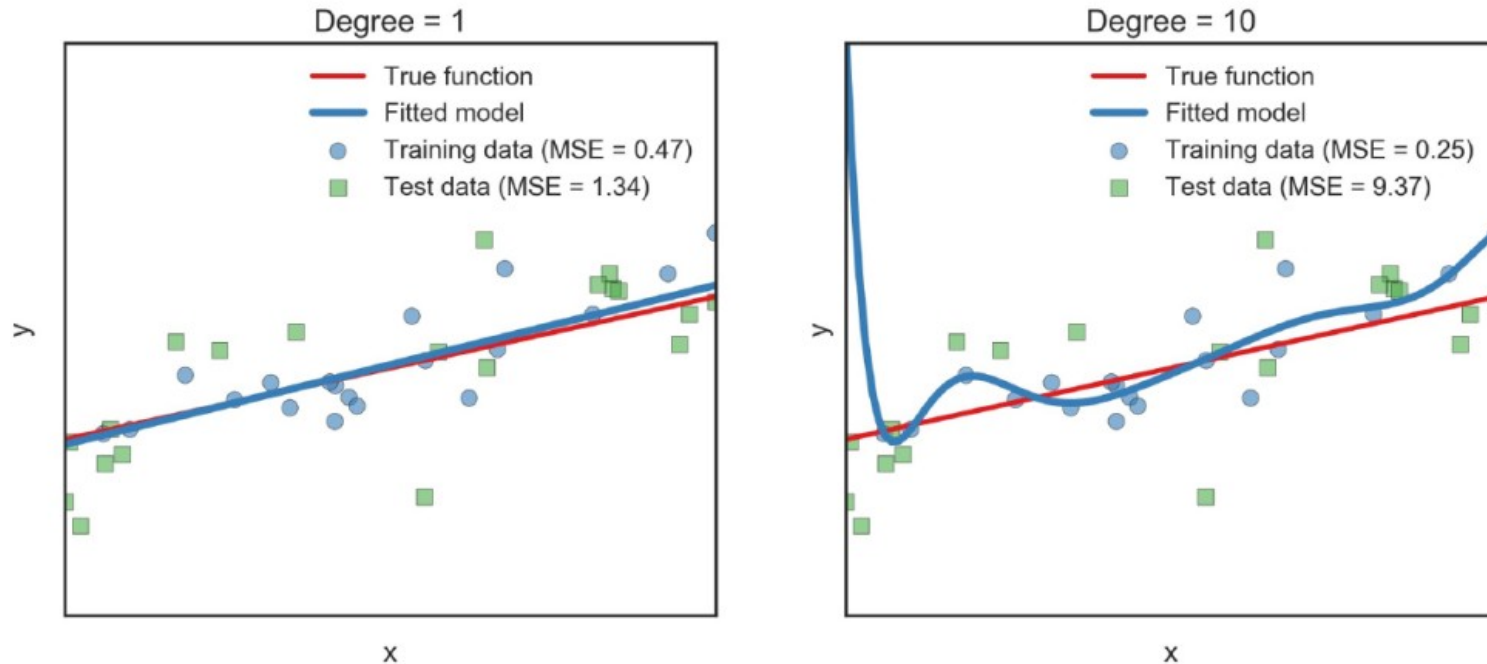
*Explanation and prediction*

- 1) Understanding that fitting (explaining) often leads to overfitting
- 2) Learning methods to prevent overfitting by introducing *bias*
- 3) Understanding how the error can be decomposed into *bias* and *variance*

# To fit is to overfit

(Yarkoni and Westfall, 2017)

**Overfitting:** fitting sample-specific noise,  
which is thus not representative of the  
population

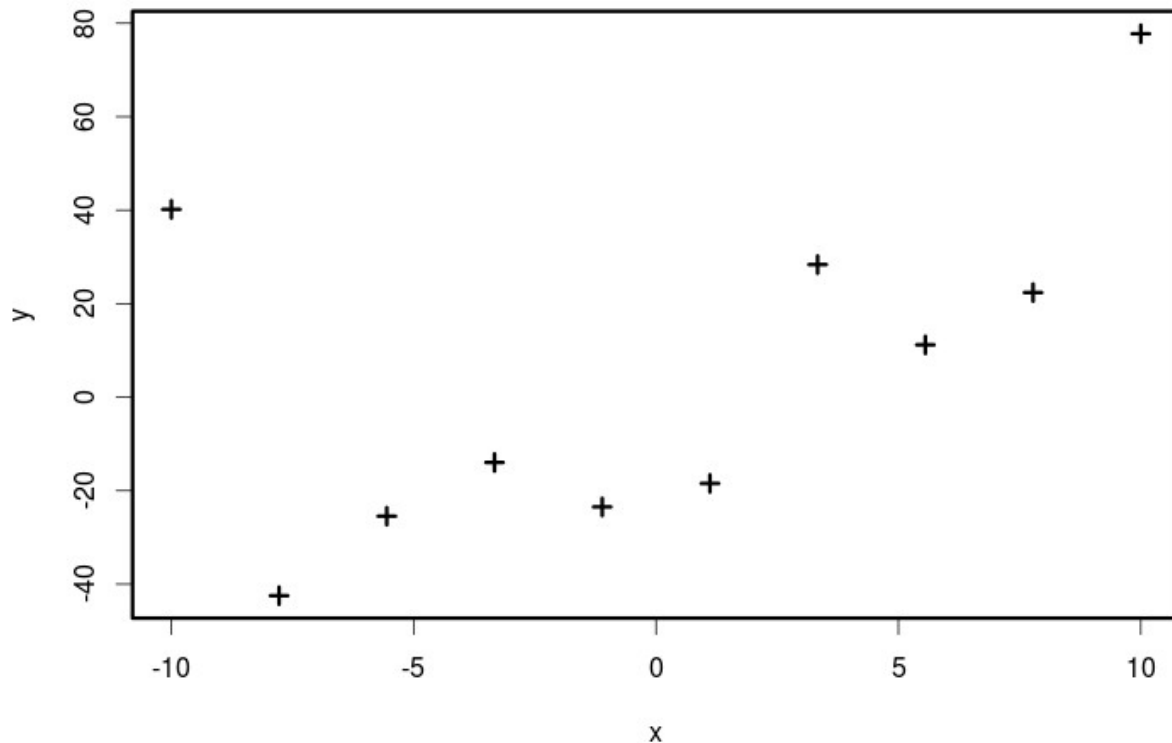


**Fig. 1.** Training and test error produced by fitting either a linear regression (left) or a 10th-order polynomial regression (right) when the true relationship in the population (red line) is linear. In both cases, the test data (green) deviate more from the model's predictions (blue line) than the training data (blue). However, the flexibility of the 10th-order polynomial model facilitates much greater overfitting, resulting in lower training error but much higher test error than the linear model. MSE = mean squared error.

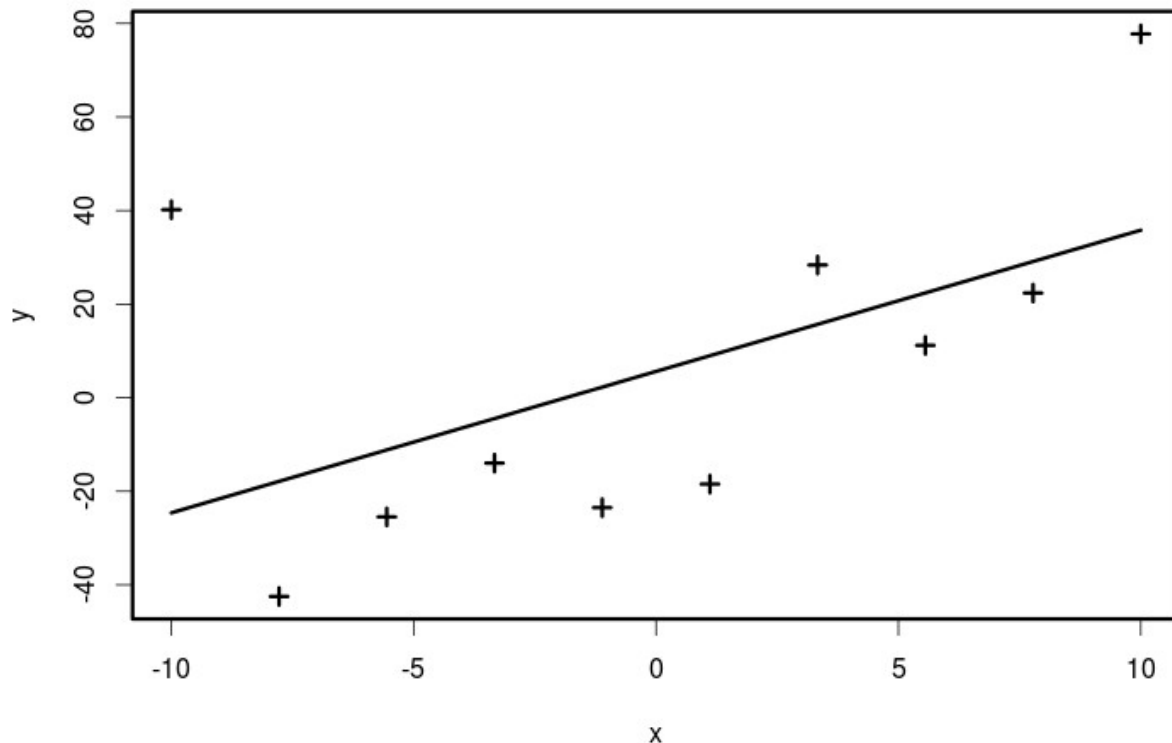
the test and training data is from the same distribution. instead of trying to evaluet how weel the first model based on the training data how waell it fits, iyt also have to fit the test data

(Yarkoni and Westfall, 2017)

# A sample of 10 linear or quadratic?

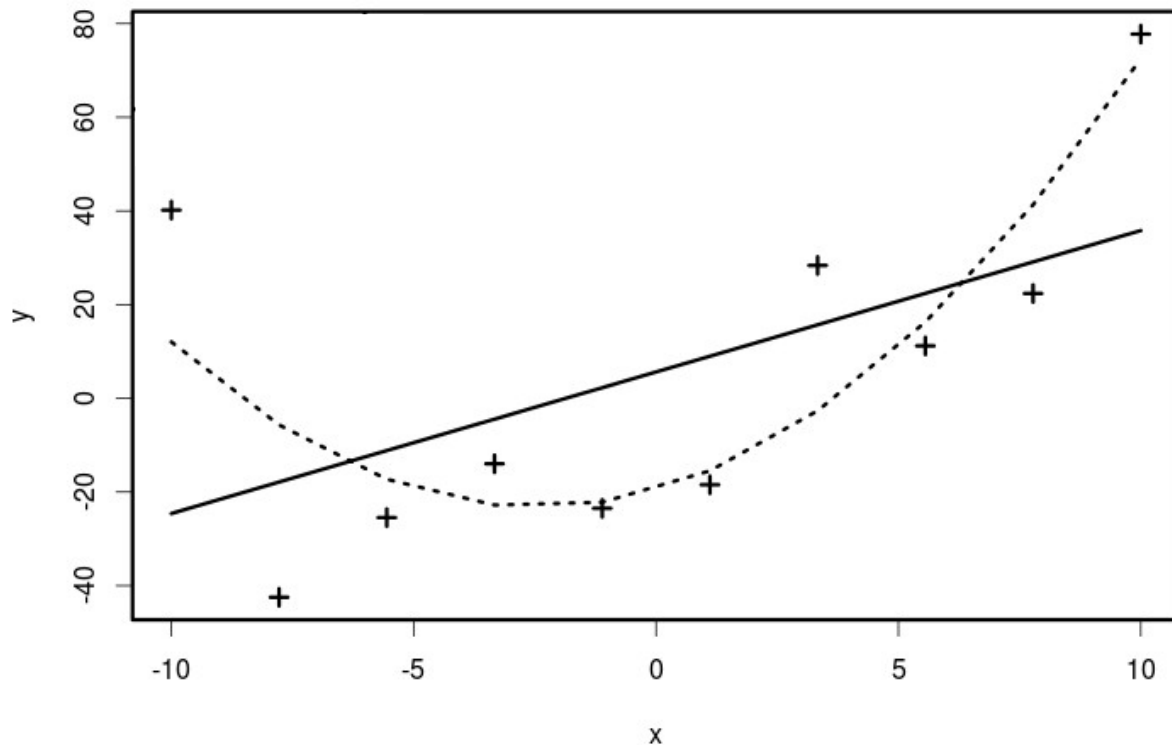


# A sample of 10 linear or quadratic?

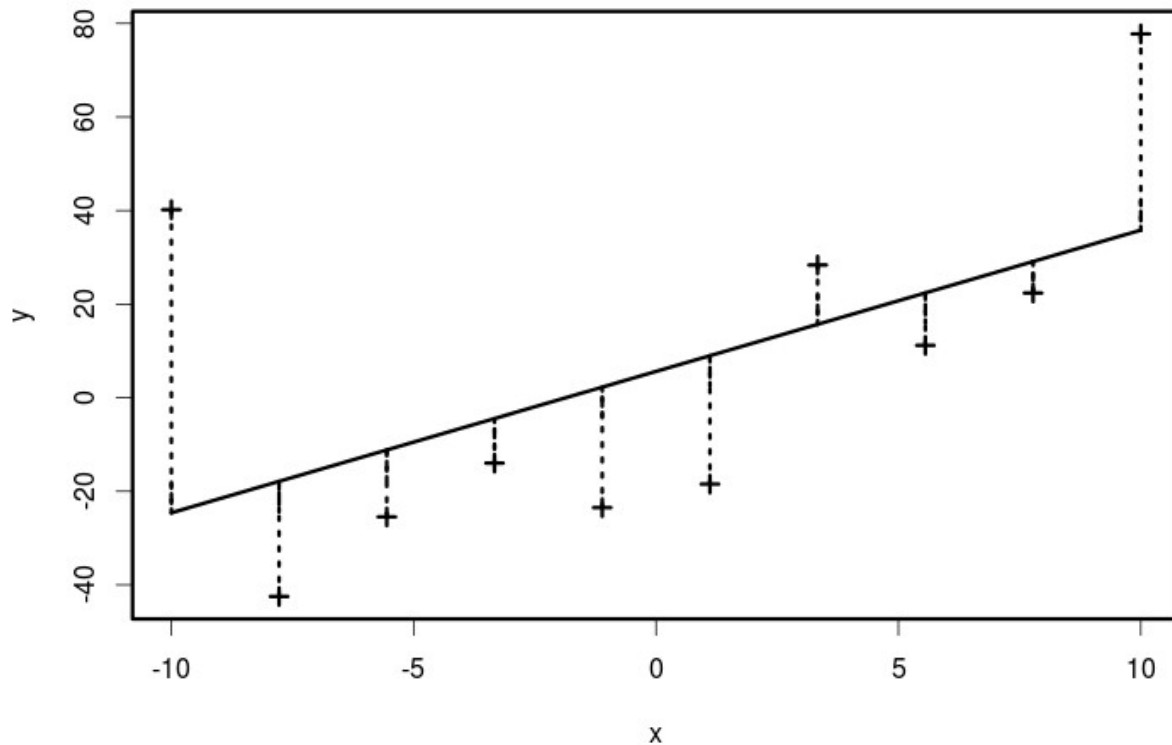




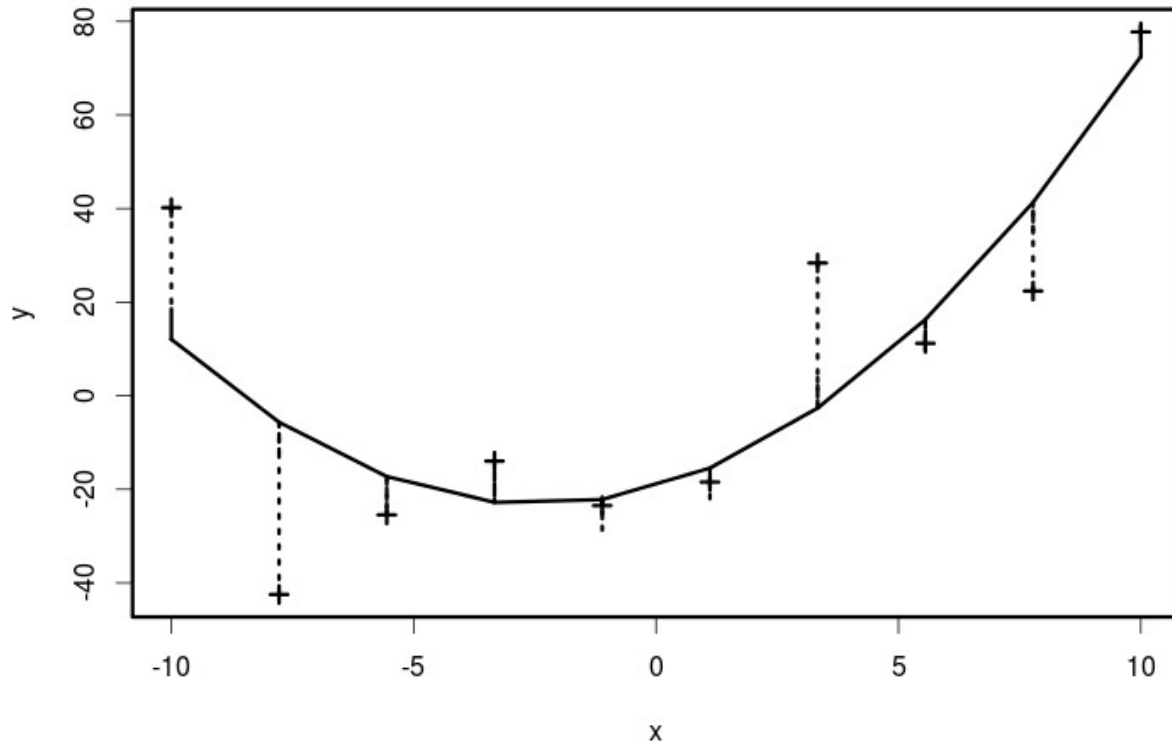
# A sample of 10 linear or quadratic?



# Residuals (linear)



# Residuals (quadratic)



Quadratic:

$$ax^2 + bx + c$$

Linear:

$$bx + c$$

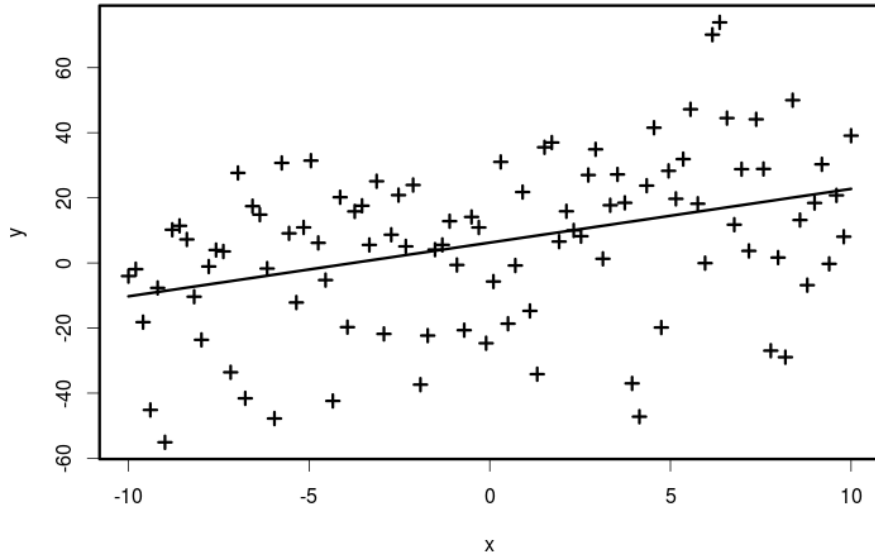
the quadratic is best

Estimates

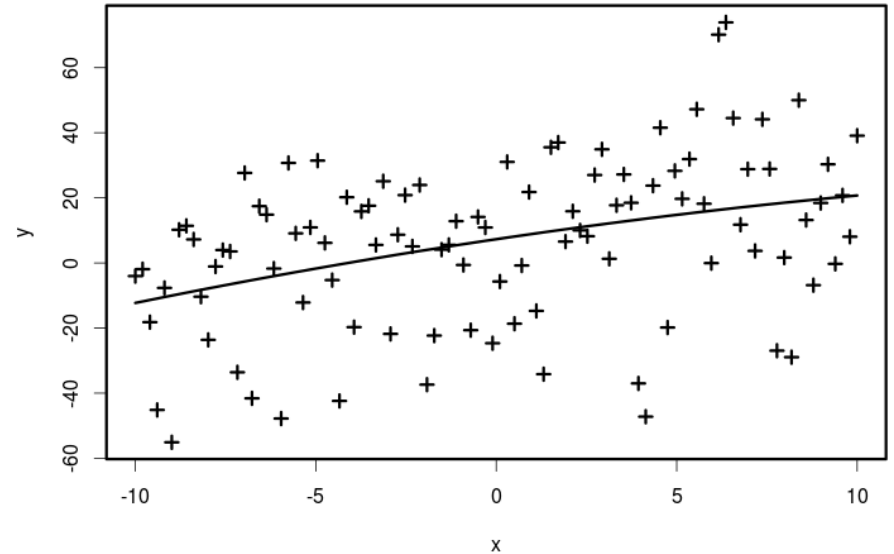
$$a = 0,6184; b = 3,0201$$

$$b = 3,020$$

# Now a sample of 100

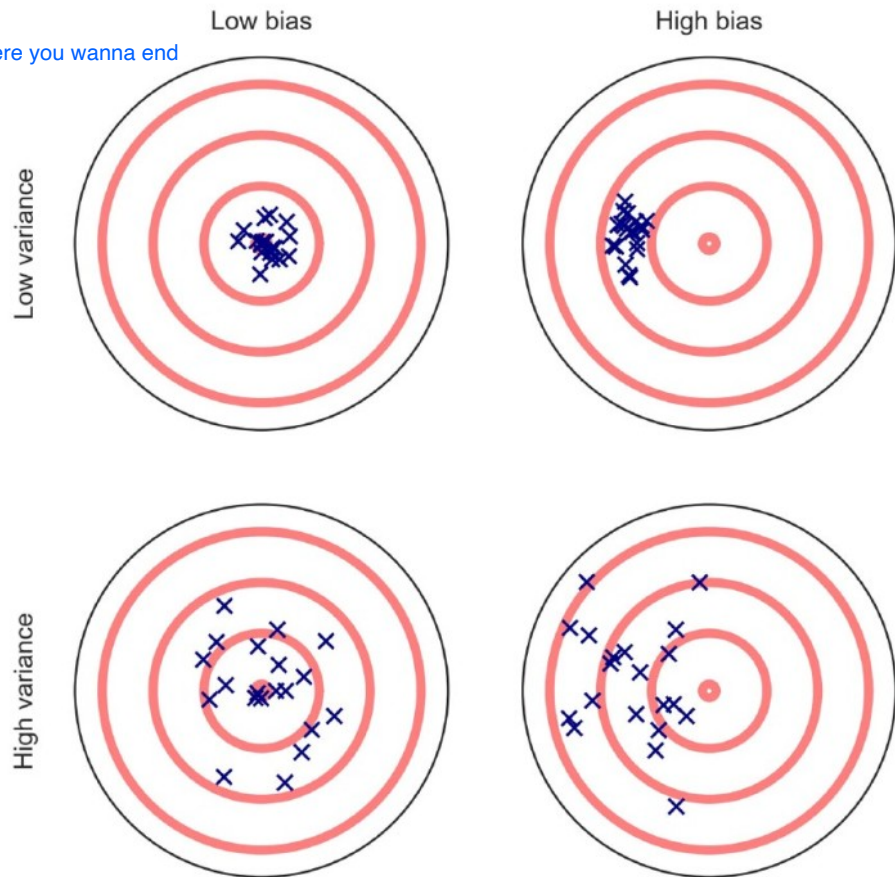


$$b = 1,650$$



$$a = -0,03074 \approx 0; b = 1,650$$

Here you wanna end

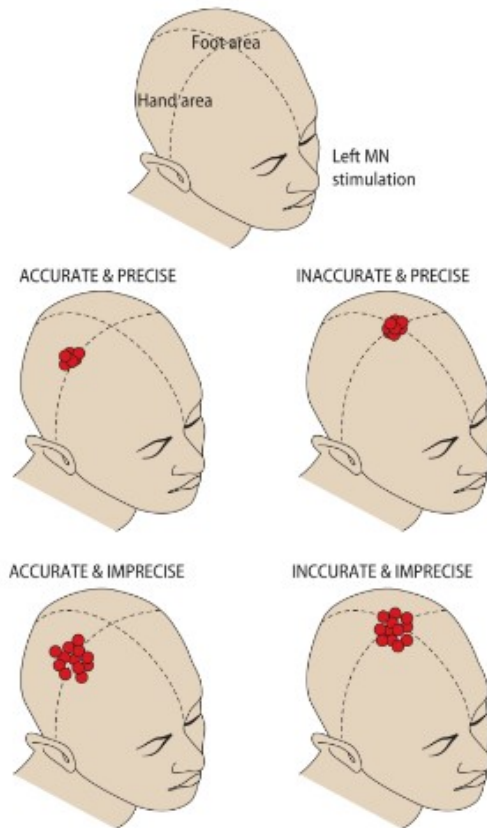


**Fig. 2.** An estimator's predictions can deviate from the desired outcome (or true scores) in two ways. First, the predictions may display a systematic tendency (or *bias*) to deviate from the central tendency of the true scores (compare right panels with left panels). Second, the predictions may show a high degree of *variance*, or imprecision (compare bottom panels with top panels).

(Yarkoni and Westfall, 2017)

Which two of these does our least-squares solution prefer?

The one with low bias  
low variance and high variance

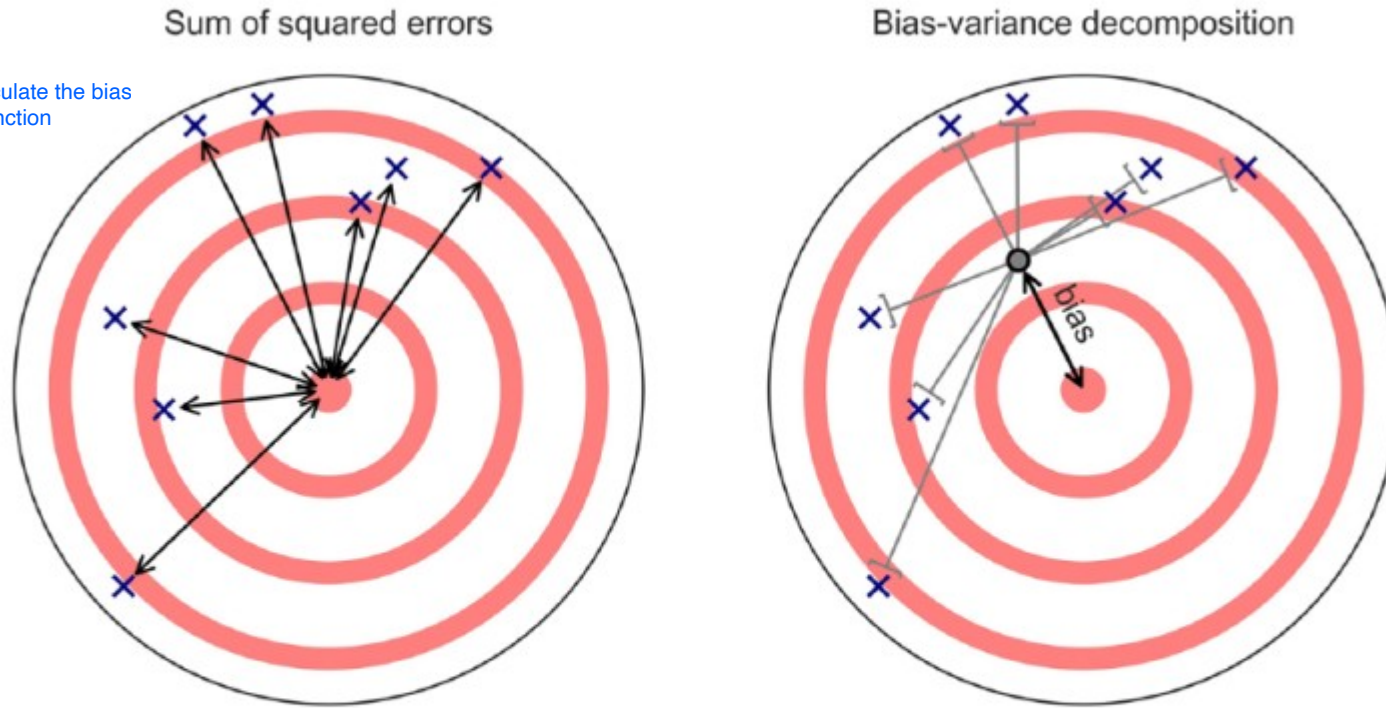


**FIGURE 3.7.** Accuracy versus precision. A schematic illustration of the differences between accuracy and precision of source localization. After left median-nerve stimulation, activations is expected in the right-hemisphere hand region of the primary somatosensory cortex. The foot area is shown at the top of the head. See text for further explanation.

(Hari and Puce, 2017)

(Yarkoni and Westfall, 2017)

In real life its not really possible to calculate the bias  
Because you need the true function



**Fig. 3.** Schematic illustration of the bias-variance decomposition. (Left) Under the classical error model, prediction error is defined as the sum of squared differences between true scores and observed scores (black lines). (Right) The bias-variance decomposition partitions the total sum of squared errors into two separate components: a bias term that captures a model's systematic tendency to deviate from the true scores in a predictable way (black line) and a variance term that represents the deviations of the individual observations from the model's expected prediction (gray lines).

# Bias variance decomposition

$$E[(y_0 - \hat{f}(x_0))^2] = \text{bias}(\hat{f}(x_0))^2 + \text{var}(\hat{f}(x_0)) + \sigma^2$$

E = expectation is the value it will take if you do it an infinite number of times

unexplained error

$$\text{bias}(\hat{f}(x_0)) = E[(y_0 - \hat{f}(x_0))^2]$$

This is wrong it should be instead:

$$E(\hat{f}(x_0)) - f(x_0)$$

the bias: how much do i shot off the target

is the *expected* squared error between the true value  $y_0$  and its estimates based on fits  $\hat{f}(x_0)$

**We'll look more into this during tomorrow's exercise**



# Multilevel modelling as a *bias* introducer

*“For example, some readers may be surprised to learn that multilevel modeling approaches to analyzing clustered data—which have recently seen a dramatic increase in adoption in psychology—improve on ordinary least squares (OLS) approaches to estimating individual cluster effects by deliberately biasing (through “shrinking” or “pooling”) the cluster estimates toward the estimated population average”*

(Yarkoni and Westfall, 2017)

# Introducing *bias*

*“In a widely used form of penalized regression called lasso regression (Tibshirani, 1996, 2011), this leastsquares criterion is retained, but the overall cost function that the estimation seeks to minimize now includes an additional penalty term that is proportional to the sum of the absolute values of the coefficients.”*

(Yarkoni and Westfall, 2017)

# Penalised regression

$$\text{RSS} = \sum (y_i - \hat{y}_i)^2 \text{ (minimise to obtain least squares solution)}$$

$$\text{lasso regression: } \text{RSS} + \lambda \sum_{j=1}^p |\beta_j| \text{ (minimise this sum)}$$

The higher the beta value the more it explains?

here you put some variables to 0, the do not explain any variance

$$\text{ridge regression: } \text{RSS} + \lambda \sum_{j=1}^p (\beta_j^2) \text{ (minimise this sum)}$$

$i$ : observations

$p$ : predictor variables

$\lambda$ : a constant

# Penalised regression

$RSS = \sum (y_i - \hat{y}_i)^2$  (minimise to obtain least squares solution)

lasso regression :  $RSS + \lambda \sum_{j=1}^p |\beta_j|$  (minimise this sum)

ridge regression :  $RSS + \lambda \sum_{j=1}^p (\beta_j^2)$  (minimise this sum)

$i$ : observations

$p$ : predictor variables

$\lambda$ : a constant

## Group discussion

In each case: what happens when?

1.  $\lambda$  increases? it shrinks the b-value to zero
2.  $\lambda$  decreases?
3.  $\lambda$  is 0? then its just an Ordinary least square regression
4.  $\lambda$  goes towards infinity?

# Penalised regression

$$\text{RSS} = \sum (y_i - \hat{y}_i)^2 \text{ (minimise to obtain least squares solution )}$$

$$\underset{\lambda}{\text{argmin}} = \text{RSS} + \lambda \sum_{j=1}^p |\beta_j|$$

$$\underset{\lambda}{\text{argmin}} = \text{RSS} + \lambda \sum_{j=1}^p (\beta_j^2)$$

$i$ : observations

$p$ : predictor variables

$\lambda$ : a constant

# How to choose $\lambda$ ?

```
##  
## Call:  
## lm(formula = hp ~ mpg + wt + drat + qsec, data = mtcars)  
##  
## Coefficients:  
## (Intercept)          mpg          wt          drat          qsec  
##      473.779       -2.877       26.037        4.819       -20.751
```

What is  $\lambda$  equal to here?

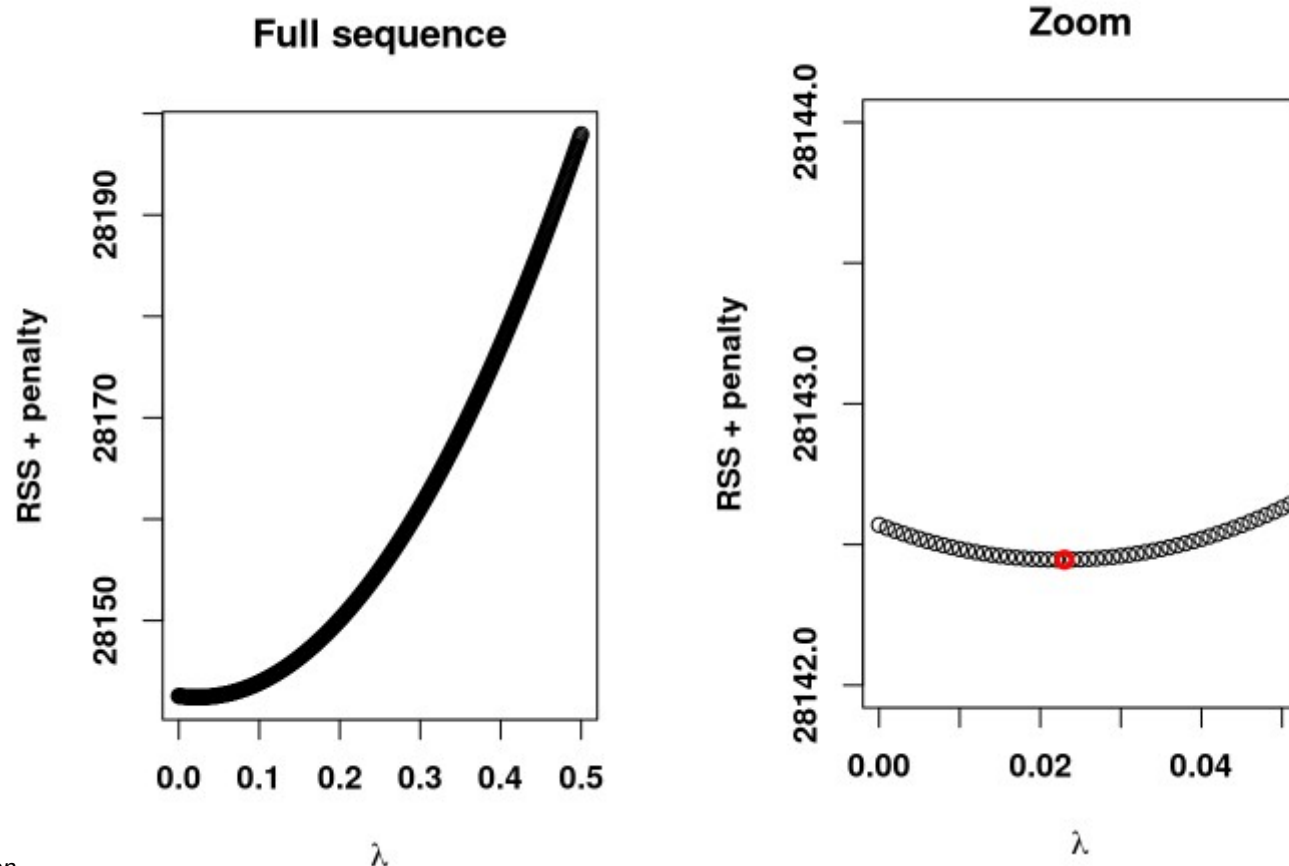
```
sum.of.squares.total <- sum((y - mean(y))^2)  
sum.of.squared.errors.lm <- sum(residuals(linear_model)^2)  
print(r.squared.lm <- 1 - sum.of.squared.errors.lm/sum.of.squares.total)
```

```
## [1] 0.8072553
```

# How to choose $\lambda_{(lasso)}$ ?

```
##  
## Call:  glmnet(x = x, y = y, alpha = 1, lambda = c(0, 0.2, 2, 4, 20,  
100))  
##  
##          lambda          RSS      penalty          sum  
## Df    %Dev Lambda          lambda          RSS      penalty          sum  
## 1    0 0.0000  100.0          100.0 145726.9          0.0          145726.9  
## 2    2 0.6567   20.0        20.00000 50025.64218        14.05496 50039.69714  
## 3    3 0.8004    4.0         4.00000 29082.92003        41.34363 29124.26366  
## 4    3 0.8051    2.0         2.00000 28408.50683        44.99057 28453.49740  
## 5    4 0.8072    0.2         0.20000 28097.60764        52.47741 28150.08505  
## 6    4 0.8073    0.0         0.00000 28088.09951        54.46997 28142.56948
```

# How to choose $\lambda$ ?





# What does $\lambda$ do (ridge)?

$$\hat{\beta}_{ridge} = \overset{\text{covariance matrix}}{(X^T X + \lambda I)^{-1}} X^T y$$

$I$ : an identity matrix with  $p$  predictor variables

$\lambda$ : a constant (small)

# What is $X^T X$ ?

```
head(X)
```

```
##                (Intercept)  mpg    wt drat  qsec
## Mazda RX4                1 21.0 2.620 3.90 16.46
## Mazda RX4 Wag            1 21.0 2.875 3.90 17.02
## Datsun 710                 1 22.8 2.320 3.85 18.61
## Hornet 4 Drive             1 21.4 3.215 3.08 19.44
## Hornet Sportabout          1 18.7 3.440 3.15 17.02
## Valiant                    1 18.1 3.460 2.76 20.22
```

```
print(cov.X)
```

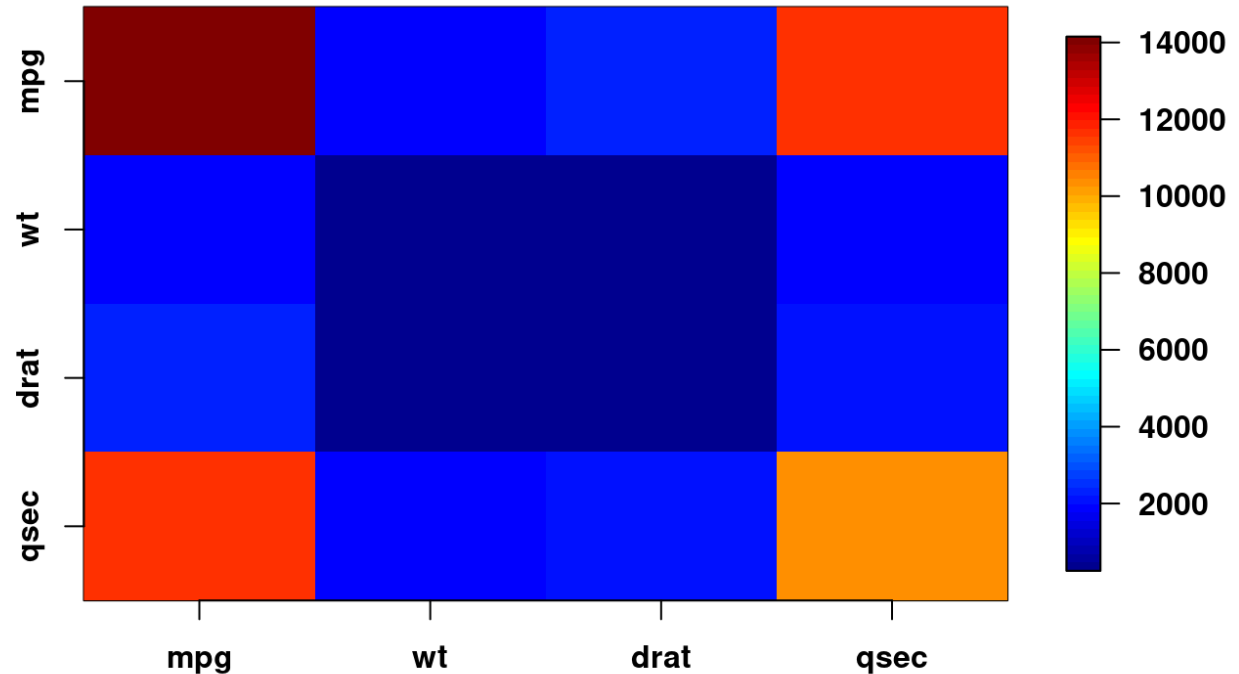
$$X_{COV} = X^T X$$

```
##                mpg          wt          drat          qsec
## mpg  14042.310  1909.7528  2380.2770  11614.745
## wt   1909.753   360.9011   358.7190   1828.095
## drat 2380.277   358.7190   422.7907   2056.914
## qsec 11614.745  1828.0946  2056.9140  10293.480
```

in the diagonal - you see the variance

$$X_{COV} = X^T X$$

Covariance matrix



The fact that the off-diagonal  $> 0$ , indicates that there is **collinearity**

# Collinearity can be bad

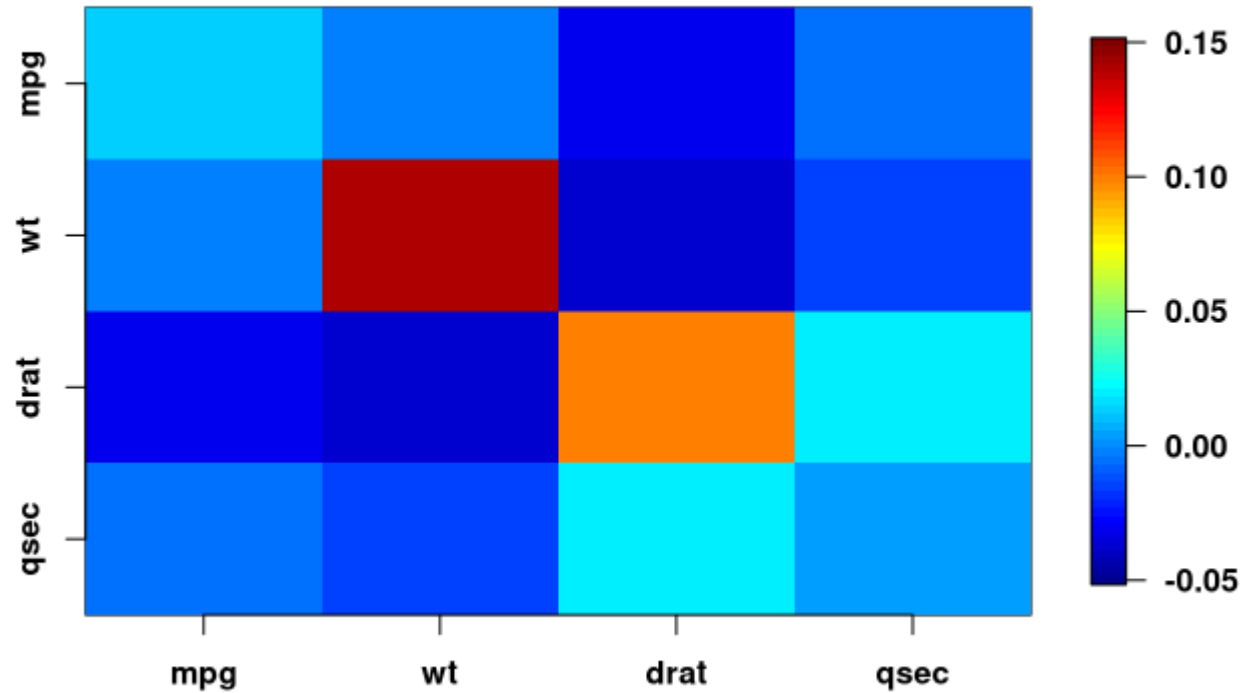
```
##  
## Call:  
## lm(formula = hp ~ mpg + wt + drat + qsec, data = mtcars)  
##  
## Coefficients:  
## (Intercept)          mpg          wt          drat          qsec  
##      473.779      -2.877      26.037       4.819     -20.751
```

Assuming no collinearity, what is the interpretation of the coefficients?  
With collinearity, is that interpretation possible?

then when going 1 it will change -2.877 in the mpg — but they are correlated and therefore you apparently cant interpret it in that way?

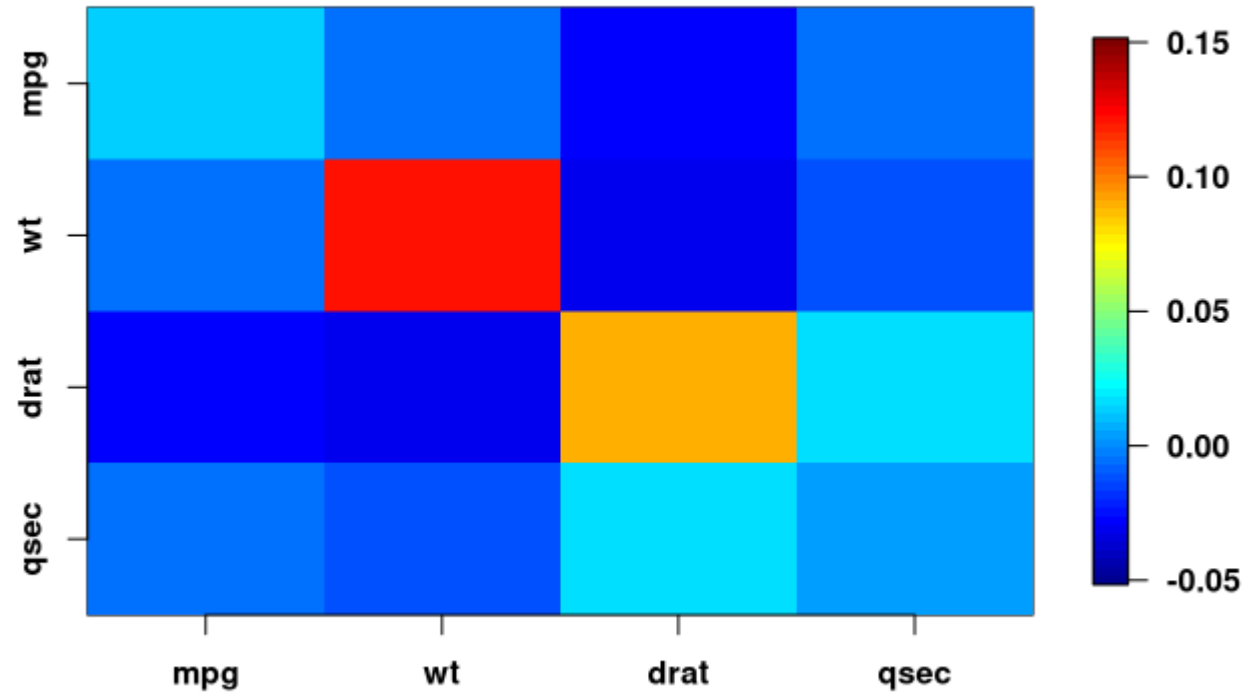
$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

Inverted Covariance matrix, regularized:(lambda= 0 )



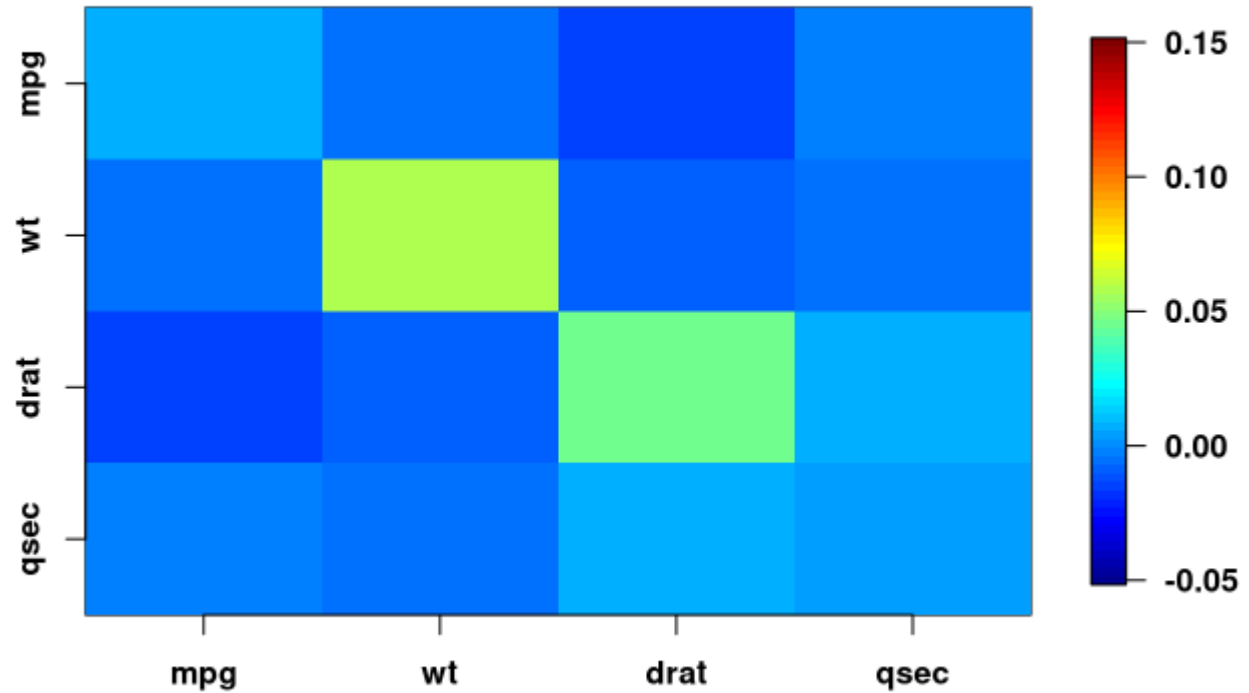
$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

Inverted Covariance matrix, regularized:(lambda= 1 )



$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

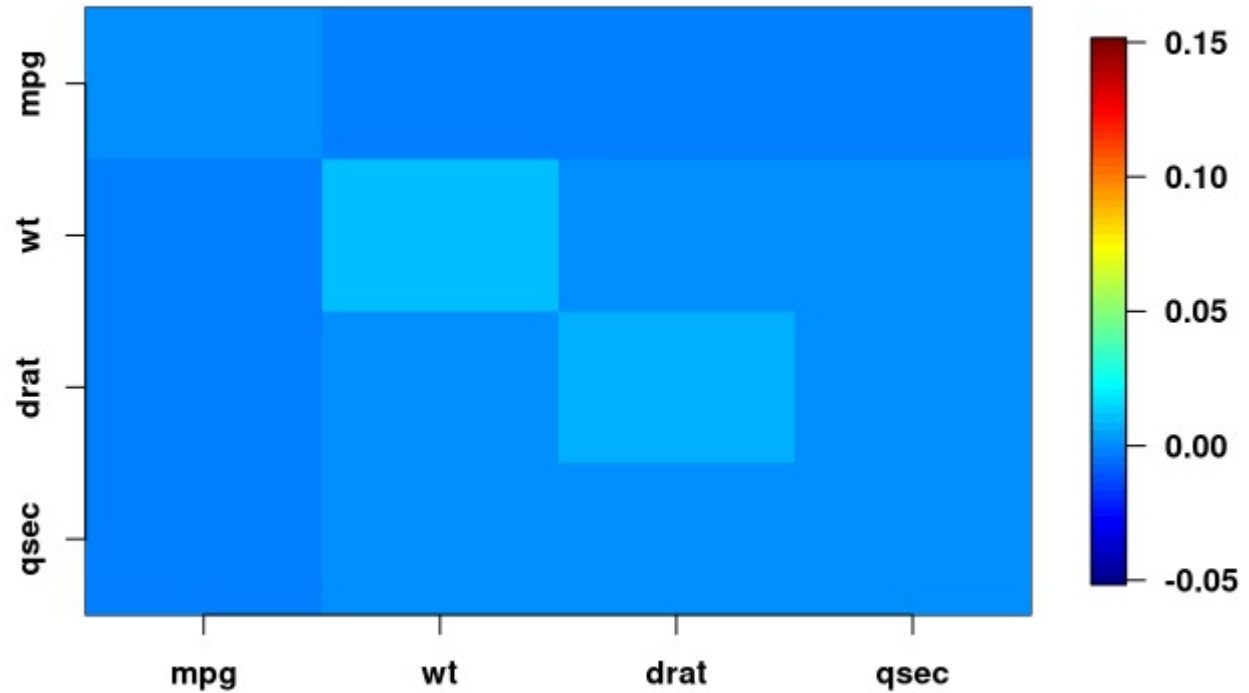
Inverted Covariance matrix, regularized:(lambda= 10 )





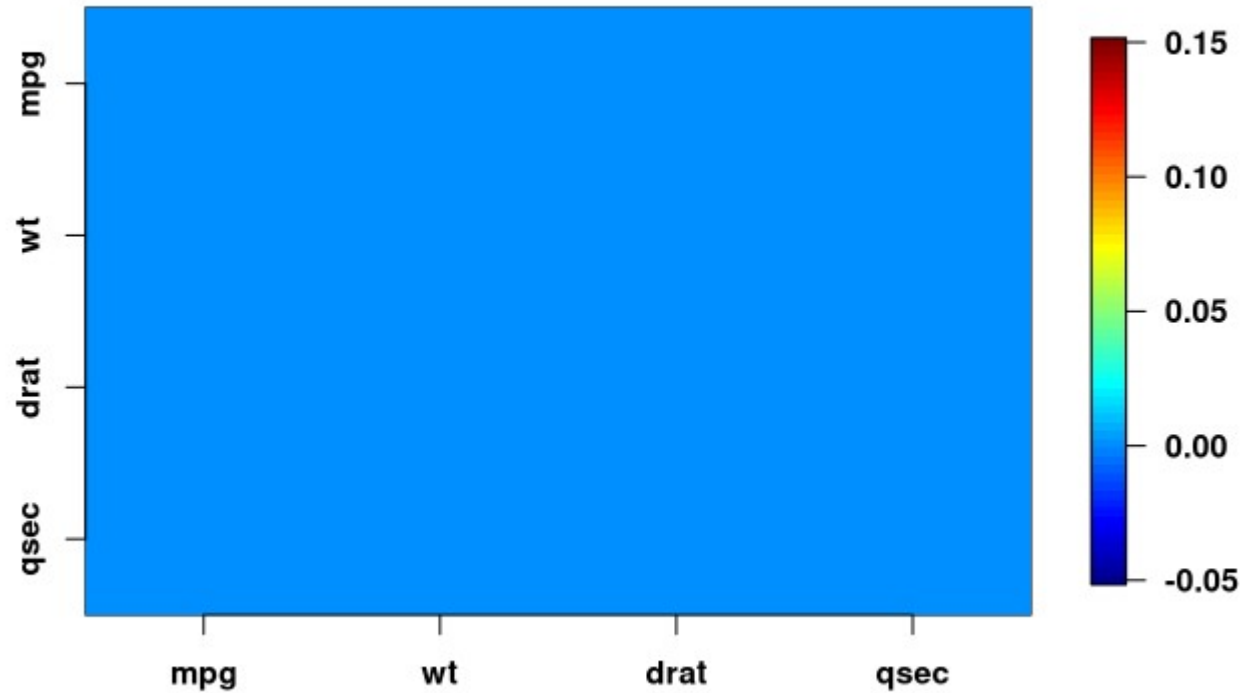
$$\hat{\beta}_{ridge} = \left( X^T X + \lambda I \right)^{-1} X^T y$$

Inverted Covariance matrix, regularized:(lambda= 100 )



$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

Inverted Covariance matrix, regularized:(lambda= 1000 )



**So why is it called  
regularisation?**

# Two notes about the inverted matrix:

with increase of  $\lambda$

1. Diagonal shrinks (bias is added)
2. Off-diagonal shrinks (collinearity is reduced, which improves the stability of the model)

## **In a stable model:**

Feeding new data or adding new predictor variables will not change the parameter estimates a lot

We have succeeded in finding a  $\lambda$  making our model more stable (improved **in-sample** validity), but we haven't found a  $\lambda$  that optimises predictive power – (**out-of-sample**)

# Out-of-sample as validity check



```
mtcars.1 <- mtcars[1:10, ]
```



# Out-of-sample as validity check

Call:

```
lm(formula = hp ~ mpg + wt + drat + qsec, data = mtcars.1)
```

Coefficients:

(Intercept)	mpg	wt	drat	qsec
414.541	-13.638	12.753	11.263	-5.042

Call:

```
lm(formula = hp ~ mpg + wt + drat + qsec, data = mtcars)
```

Coefficients:

(Intercept)	mpg	wt	drat	qsec
473.779	-2.877	26.037	4.819	-20.751



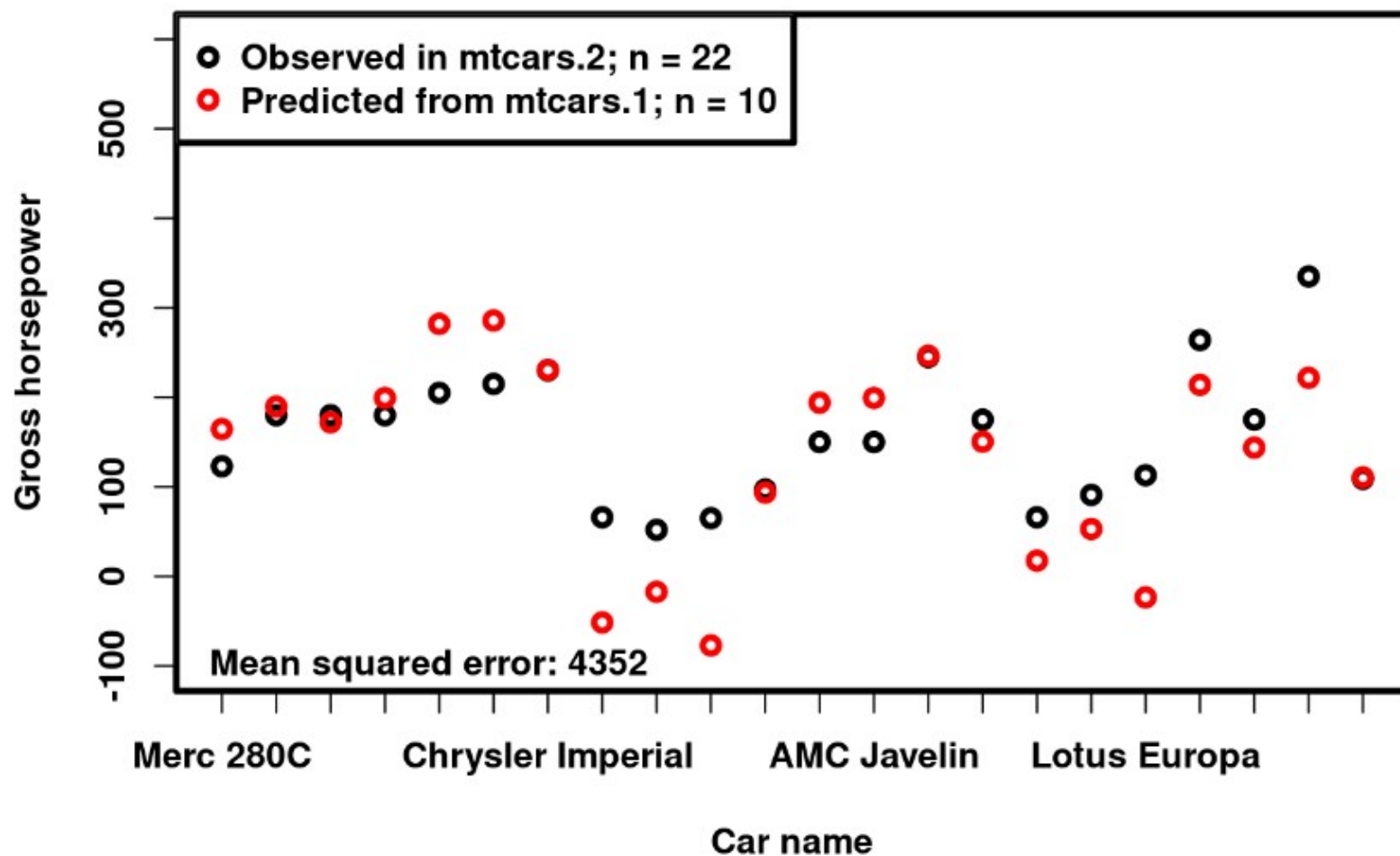
Suddenly, someone shows up with



Let's check  
our model

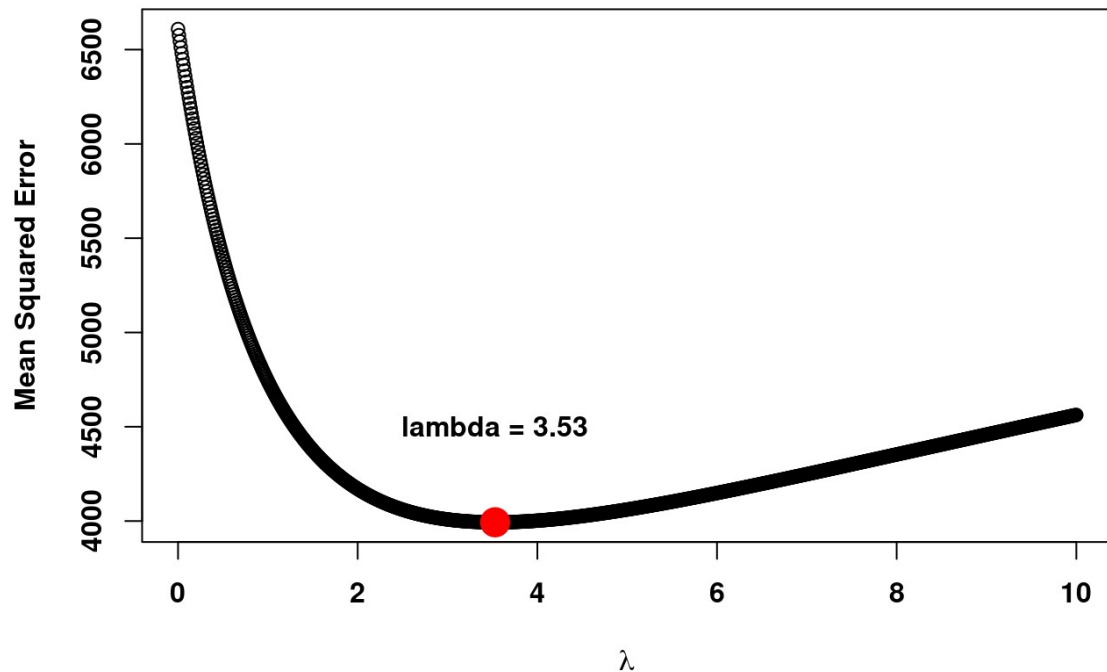
```
mtcars.2 <- mtcars[11:32, ]
```

## Predictions based on mtcars.1



# Finding optimal lambda

## Ridge Regression



$$MSE = \text{mean}((y - \hat{y})^2)$$

Which dataset is the MSE calculated on?

Call:

```
lm(formula = hp ~ mpg + wt + drat + qsec + 0, data = mtcars.1)
```

Coefficients:

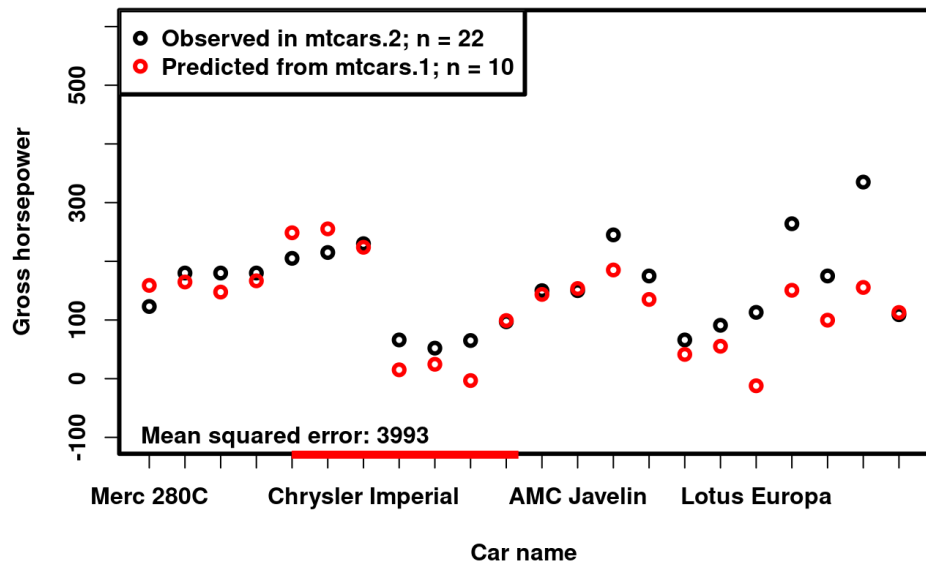
mpg	wt	drat	qsec
-8.379	76.588	45.705	-5.850

```
print(beta.hat.ridge <- ridge.regression(X, y, mtcars.1, min.lambda))
```

```
##           mpg      wt      drat      qsec  
## [1,] -7.421887 33.637 22.0378 4.70691
```

## What has happened to the coefficients?

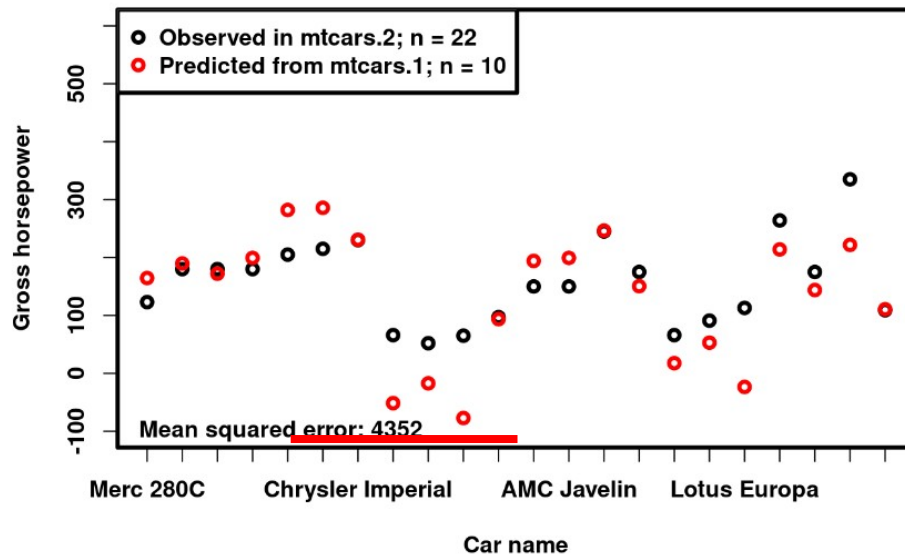
Predictions based on mtcars.1;  $\lambda=3.53$



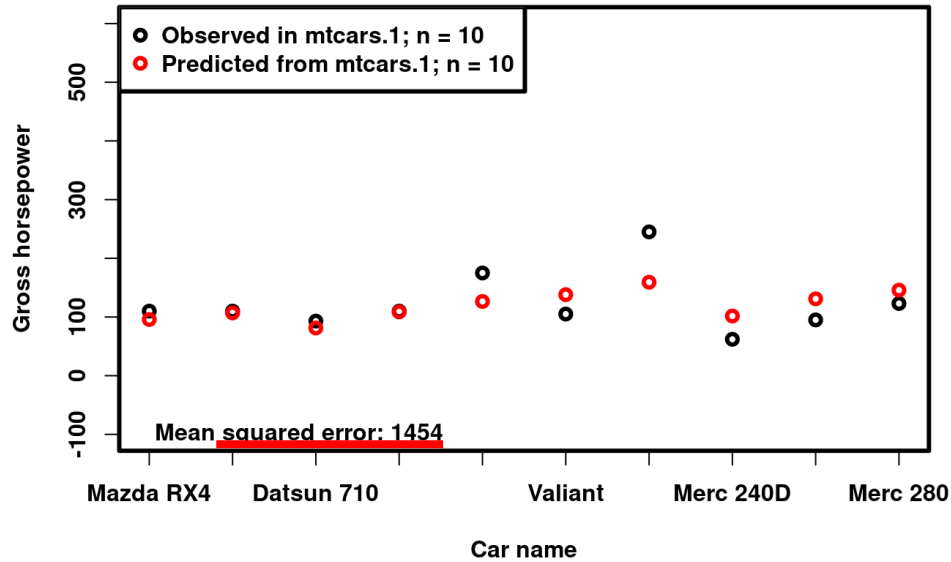
## Prediction on mtcars.2

$$\lambda=0$$

Predictions based on mtcars.1



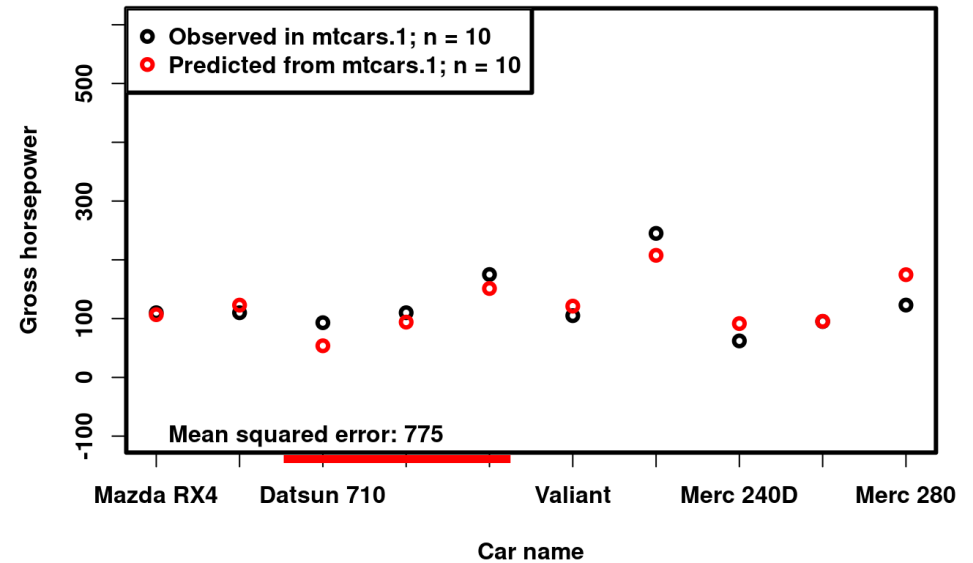
Predictions based on mtcars.1; lambda=3,53



# “Prediction” on mtcars.1

$$\lambda = 0$$

Predictions based on mtcars.1; lambda=0



# Nomenclature

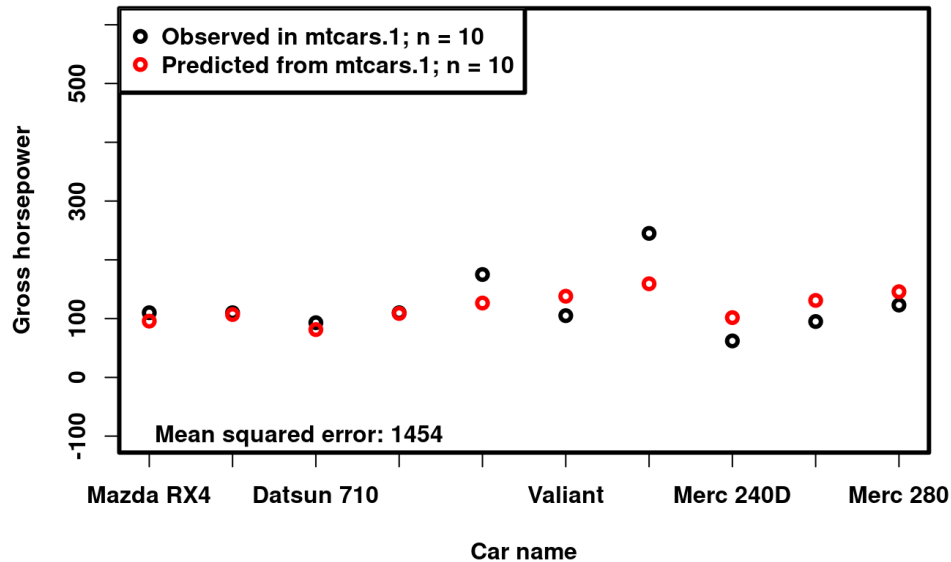
- mtcars.1 -> training set
- mtcars.2 -> test set
  - NB! Normally, we prefer that our training set is bigger than our test set
- By introducing *bias* in our training set, we at the same time reduce the *variance* of our training set, increasing the reliability of our predictions on a test set

# “Testing” on training set

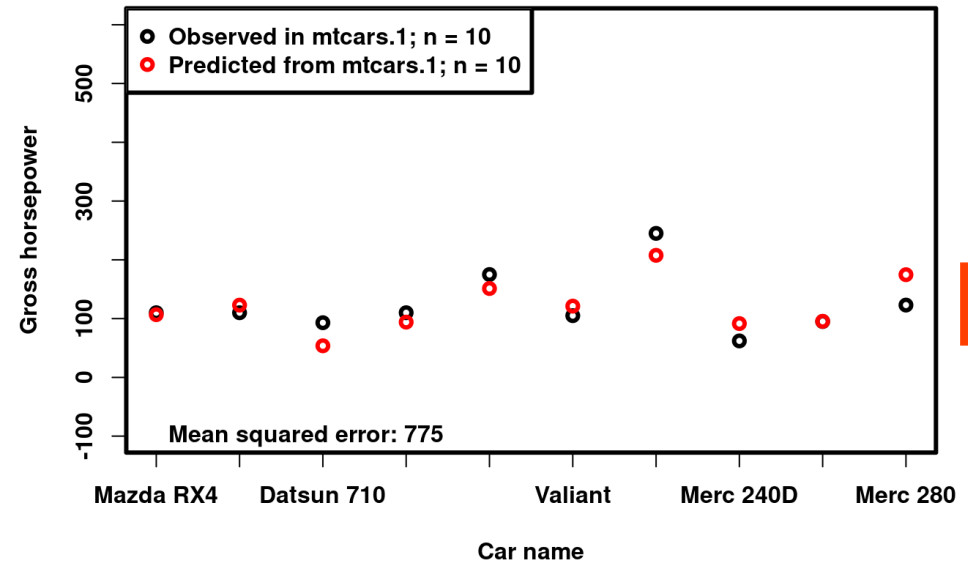
greater bias  
lesser variance

smallest bias  
greater variance (of  $\hat{y}$ )

Predictions based on mtcars.1; lambda=3,53



Predictions based on mtcars.1; lambda=0



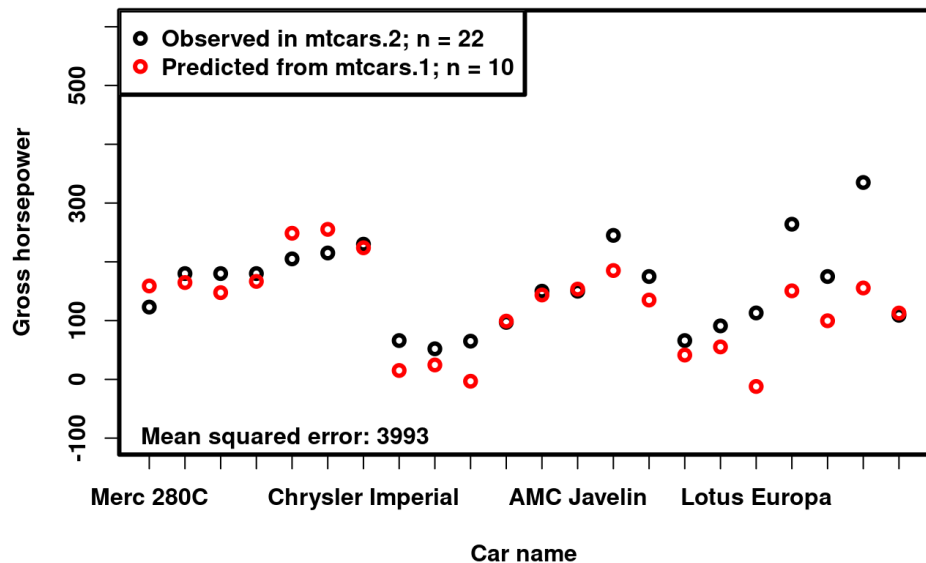


# Optimal $\lambda$

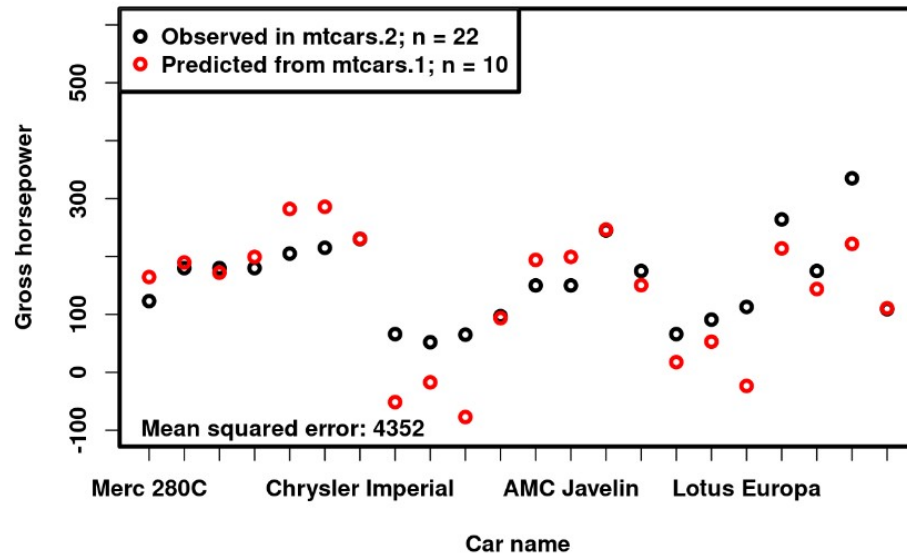
lesser bias  
lesser variance

more bias  
greater variance (of  $\hat{y}$ )

Predictions based on mtcars.1; lambda=3,53



Predictions based on mtcars.1



# Did you learn?

*Explanation and prediction*

- 1) Understanding that fitting (explaining) often leads to overfitting
- 2) Learning methods to prevent overfitting by introducing *bias*
- 3) Understanding that the error can be decomposed into *bias* and *variance*

# Next time

- The Perceptron
- Adaline
- Linear regression

# References

- Bolker, B.M., Brooks, M.E., Clark, C.J., Geange, S.W., Poulsen, J.R., Stevens, M.H.H., White, J.-S.S., 2009. Generalized linear mixed models: a practical guide for ecology and evolution. *Trends in Ecology & Evolution* 24, 127–135.  
<https://doi.org/10.1016/j.tree.2008.10.008>
- Gelman, A., Hill, J., 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.
- Hari, R., Puce, A., 2017. *MEG-EEG Primer*. Oxford University Press, New York, NY, US.
- Yarkoni, T., Westfall, J., 2017. Choosing Prediction Over Explanation in Psychology: Lessons From Machine Learning. *Perspect Psychol Sci* 12, 1100–1122. <https://doi.org/10.1177/1745691617693393>