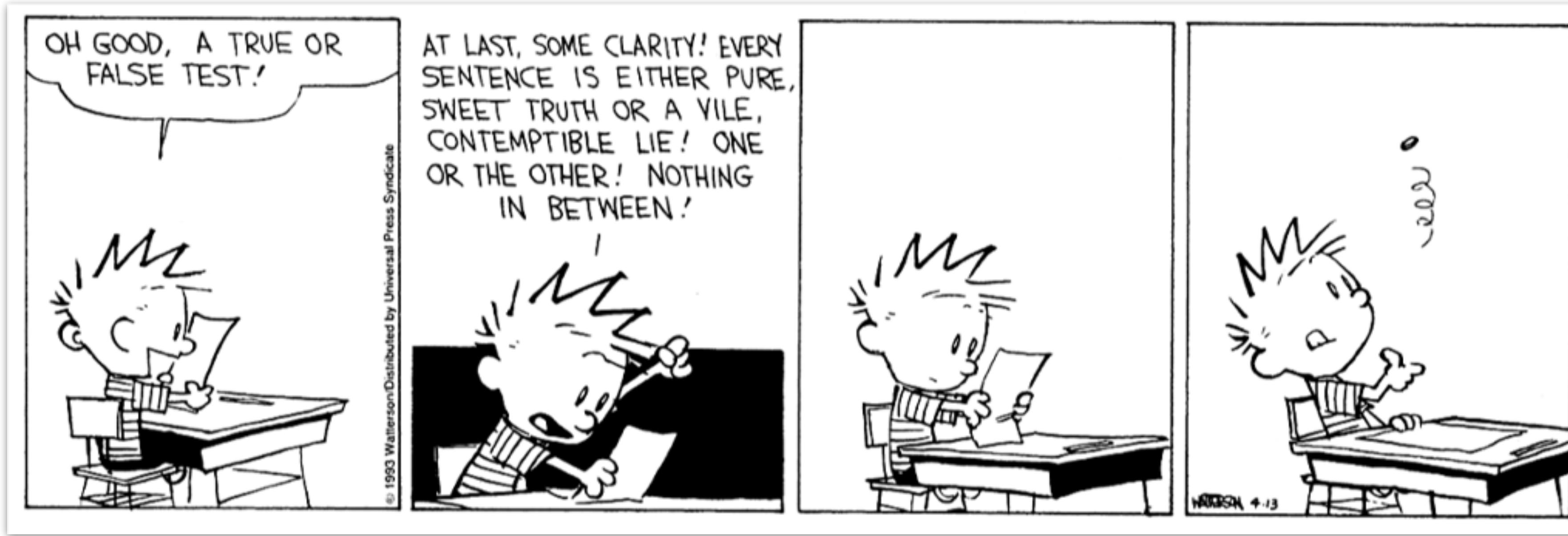


Model comparison



02/19/2025

Logistics

Project proposal

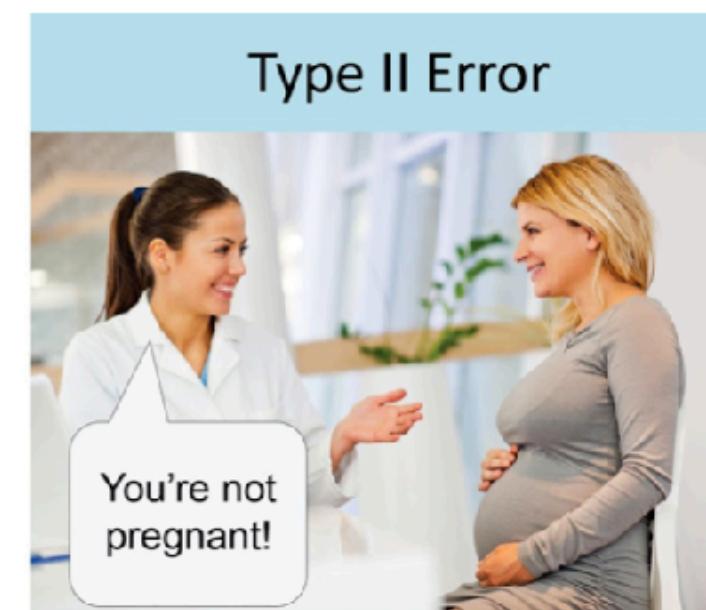
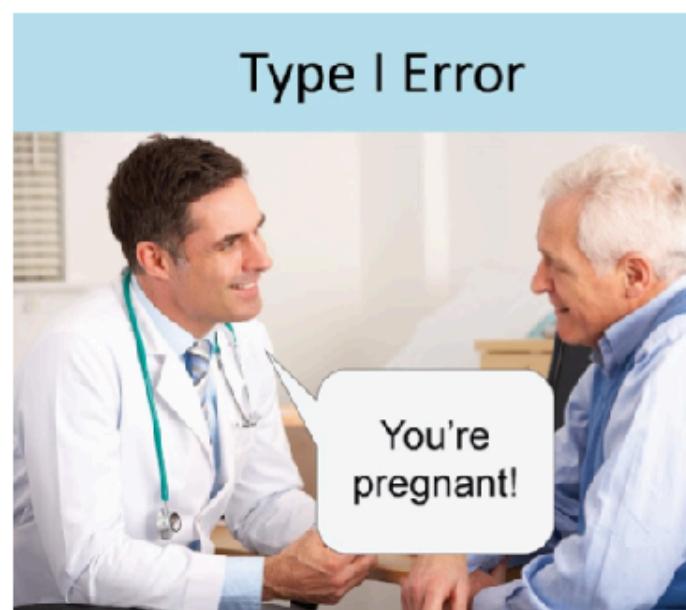
The project proposal is due on
Thursday, February 20th at 8pm

Plan for today

- Quick recap
- Model comparison
 - Cross-validation
 - AIC and BIC

Quick recap

Quick recap: Power analysis

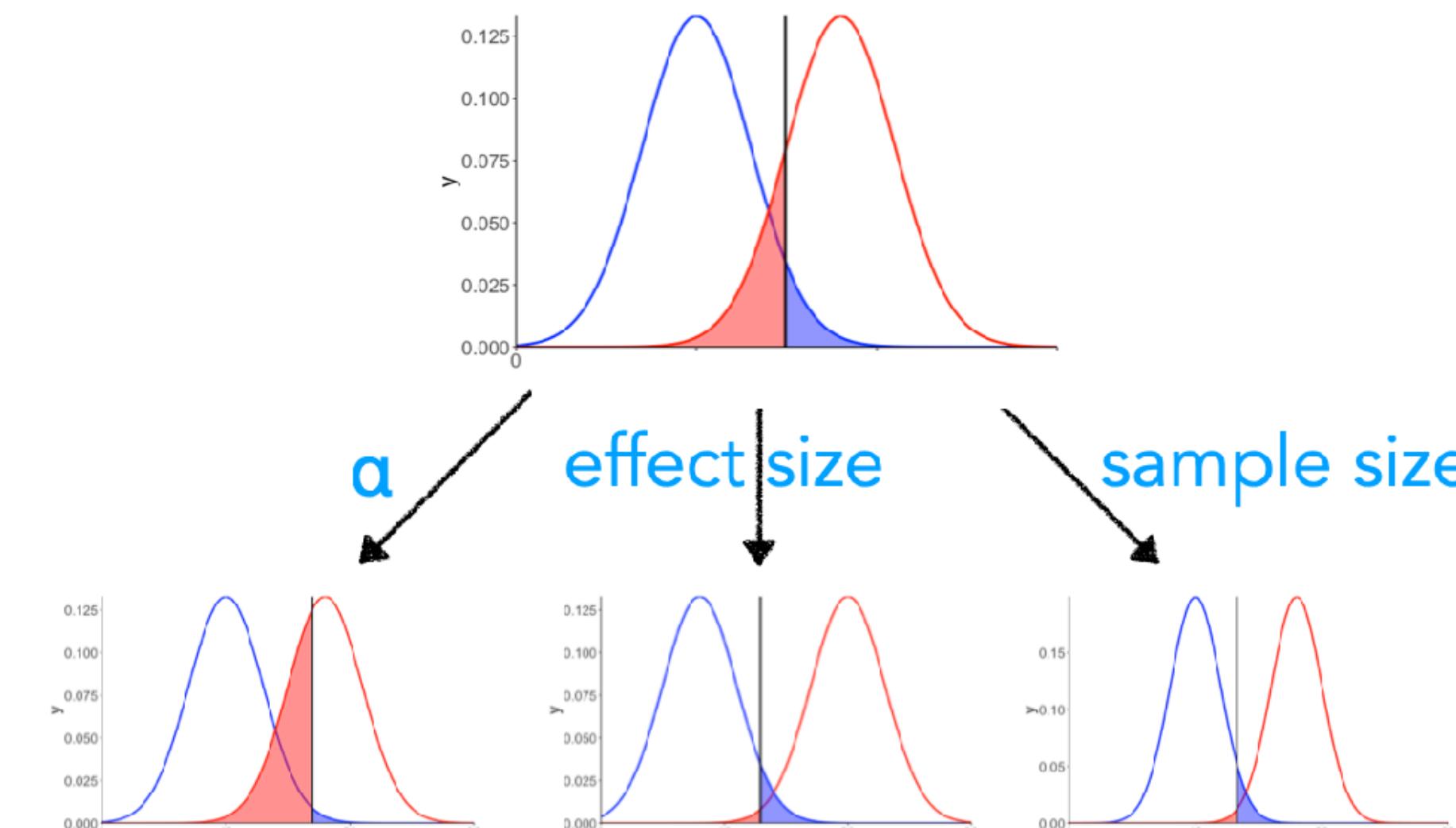


H_0 : Not pregnant. H_1 : Pregnant.

Type I Error: Falsely rejecting the null hypothesis (even though it is true).

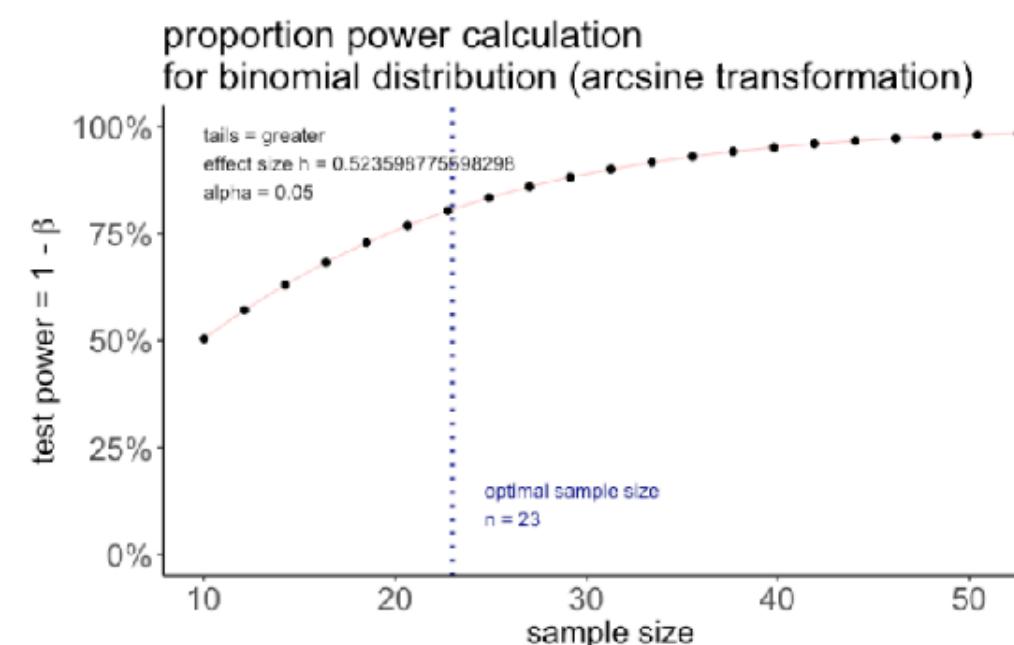
Type II Error: Failing to reject the null hypothesis (even though it is false).

The knobs we can turn to affect power



"`pwr`" package in R

```
1 library("pwr")
2 pwr.p.test(h = ES.h(p1 = 0.75, p2 = 0.50),
3             sig.level = 0.05,
4             power = 0.80,
5             alternative = "greater") %>%
6   plot()
```



Option 2

67

Power simulation recipe

- assume:
 - α , n , effect size
- simulate a large number of data sets of size n with the specified effect size
- for each data set, run a statistical test to calculate the p-value
- determine the probability of rejecting the H_0 (given that H_1 is true)

Option 3

54

Quick recap: Power analysis

Let's simulate ...

```
1 # make reproducible
2 set.seed(1)
3
4 # number of simulations
5 n_simulations = 5
6
7 # run simulation
8 expand_grid(n = seq(10, 40, 2),
9             simulation = 1:n_simulations,
10            p = 0.75) %>%
11 mutate(index = 1:n(),
12         .before = n) %>%
13 group_by(index, n, p, simulation) %>%
14 mutate(response = rbinom(n = 1,
15                         size = n,
16                         prob = p),
17         p.value = binom.test(x = response,
18                               n = n,
19                               p = 0.5,
20                               alternative = "two.sided")$p.value) %>%
21 group_by(n, p) %>%
22 summarize(power = sum(p.value < 0.05) / n())
```

Model comparison

The general procedure

1. Define H_0 as Model C (compact) and H_1 as Model A (augmented)
2. Fit model parameters to the data
3. Calculate the proportional reduction of error (PRE) in our sample
4. Decide whether the augmented model is **worth it** by comparing the observed PRE in our sample to the sampling distribution of PRE (assuming that H_0 is true)

Any problems with our approach?

sometimes it doesn't work ...

Model C

$$\text{balance}_i = \beta_0 + \epsilon_i$$

$$\text{balance}_i = \beta_0 + \beta_1 \cdot \text{student}_i + \epsilon_i$$

$$\text{balance}_i = \beta_0 + \beta_1 \cdot \text{student}_i + \epsilon_i$$

$$\text{balance}_i = \beta_0 + \beta_1 \cdot \text{student}_i + \epsilon_i$$

Model A

$$\text{balance}_i = \beta_0 + \beta_1 \cdot \text{student}_i + \beta_2 \cdot \text{age}_i + \epsilon_i$$

$$\text{balance}_i = \beta_0 + \beta_1 \cdot \text{student}_i + \beta_2 \cdot \text{age}_i + \epsilon_i$$

$$\text{balance}_i = \beta_0 + \beta_1 \cdot \text{age}_i + \epsilon_i$$

$$\text{balance}_i = \beta_0 + \beta_1 \cdot \text{age}_i + \beta_2 \cdot \text{degree}_i + \epsilon_i$$

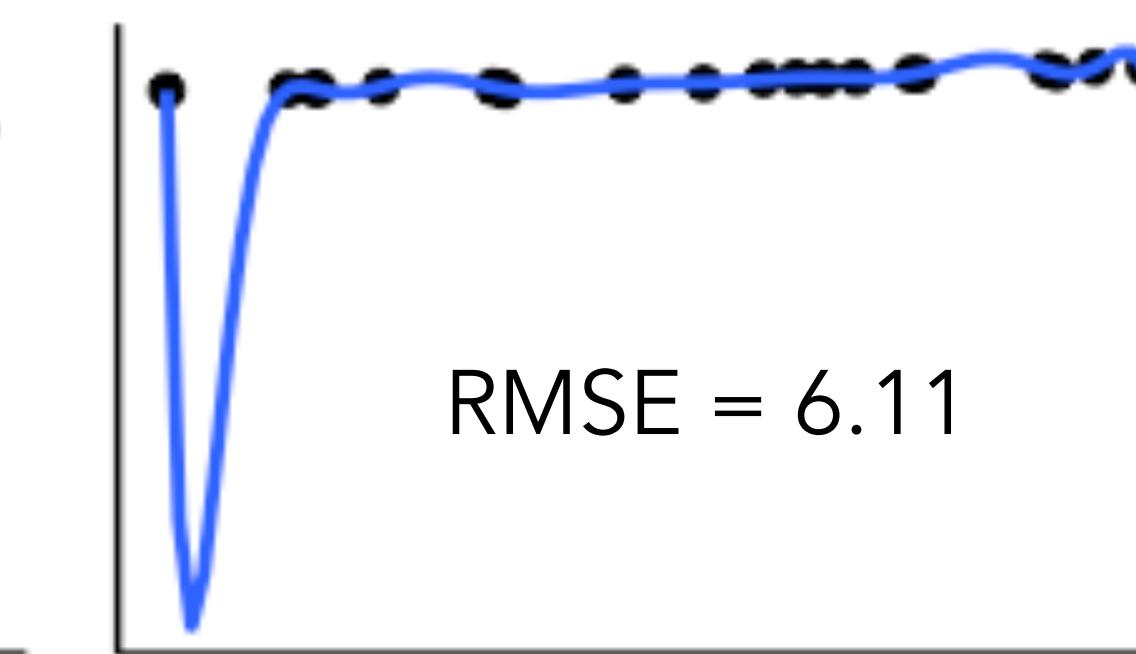
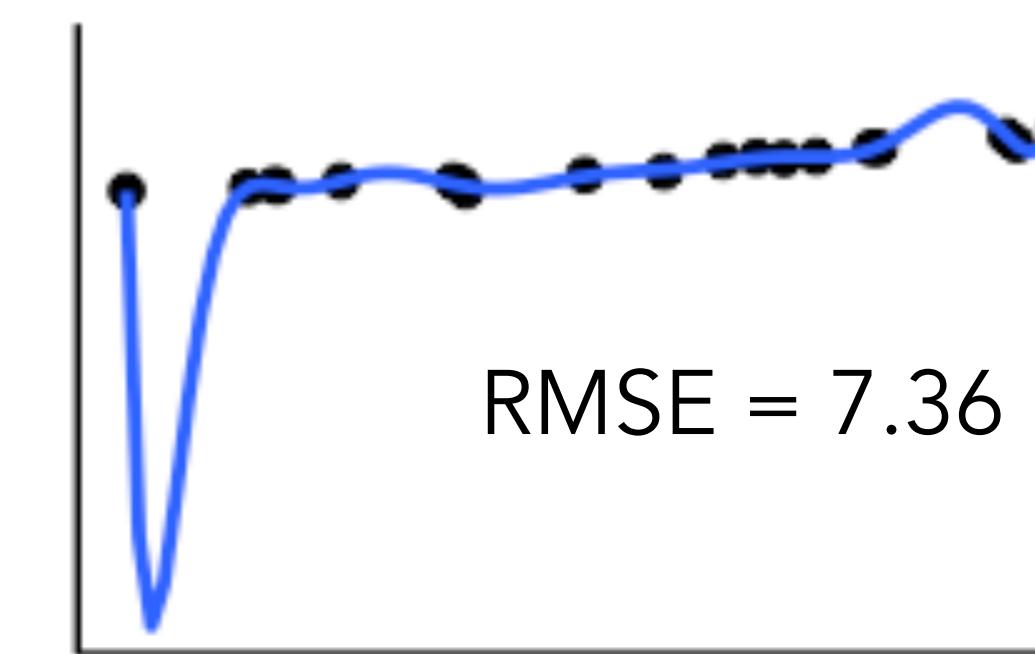
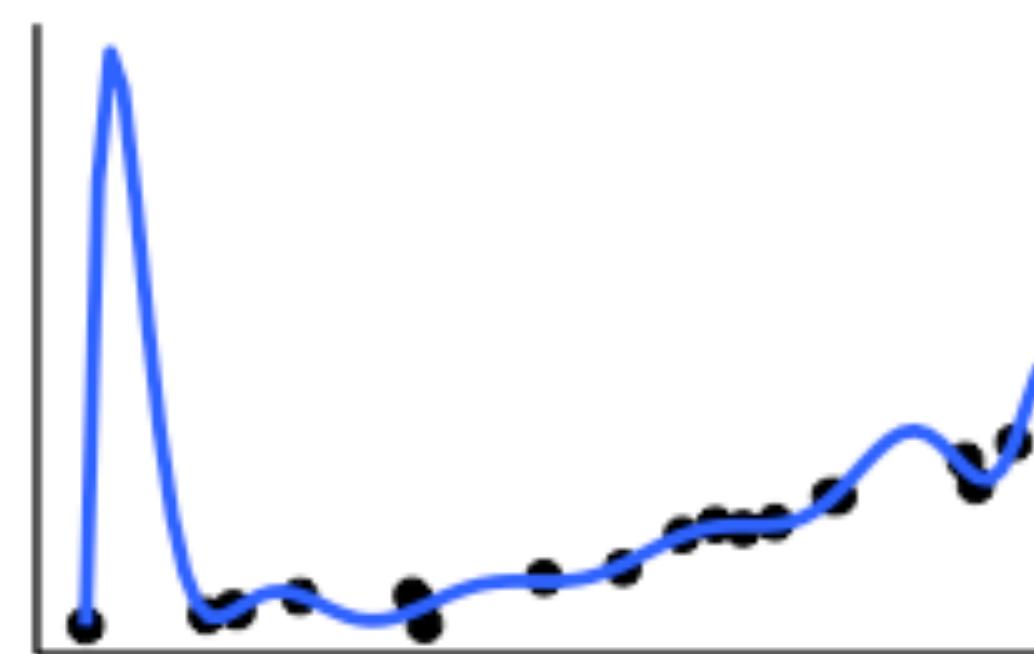
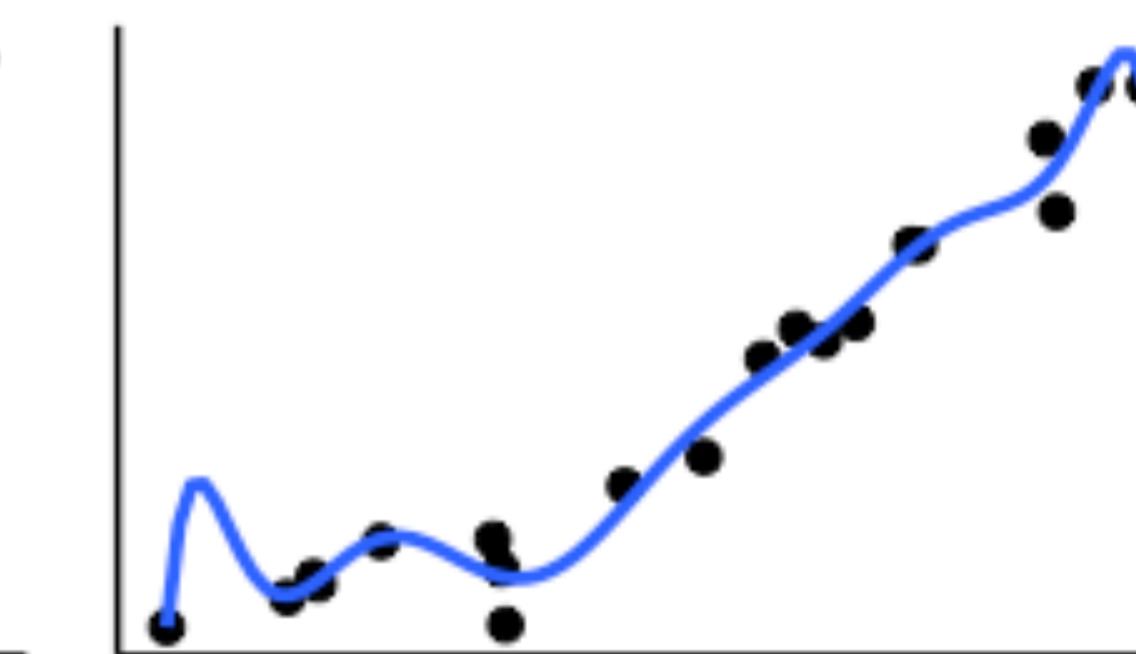
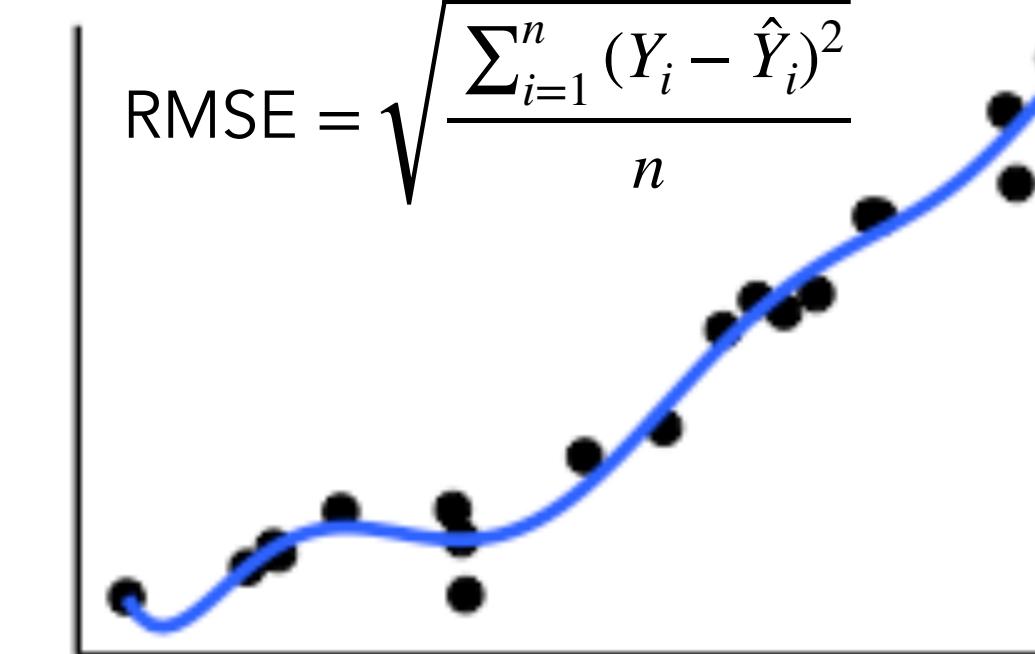
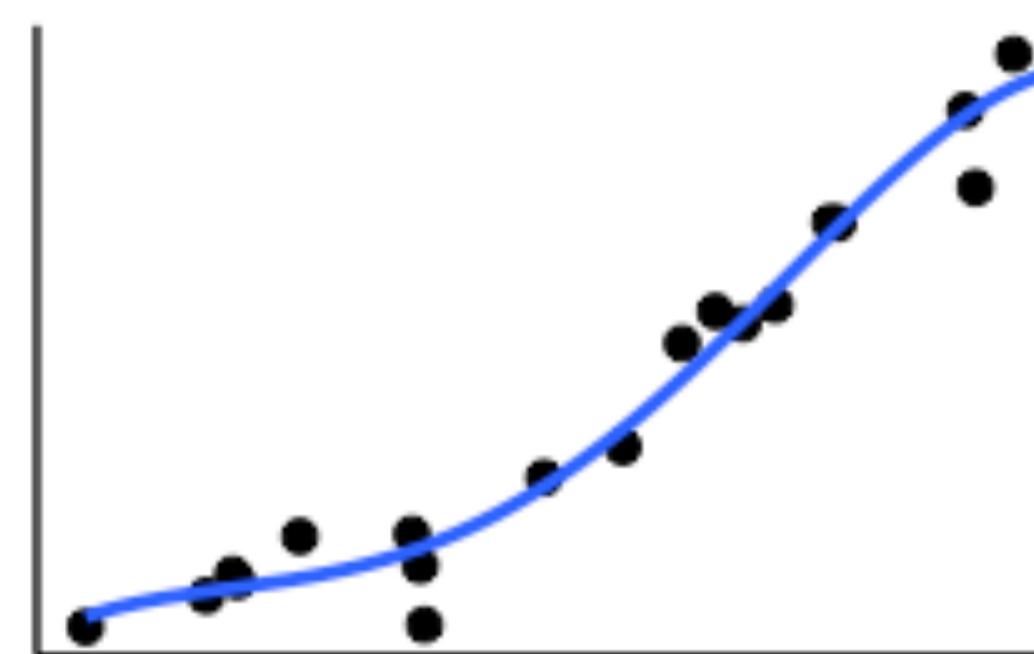
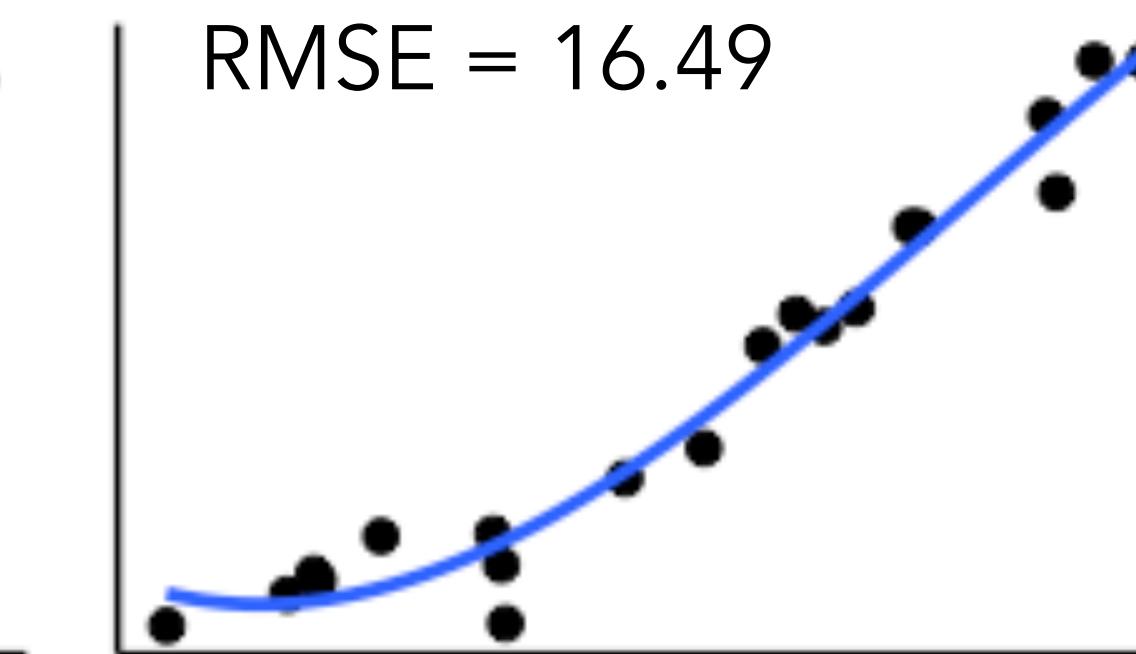
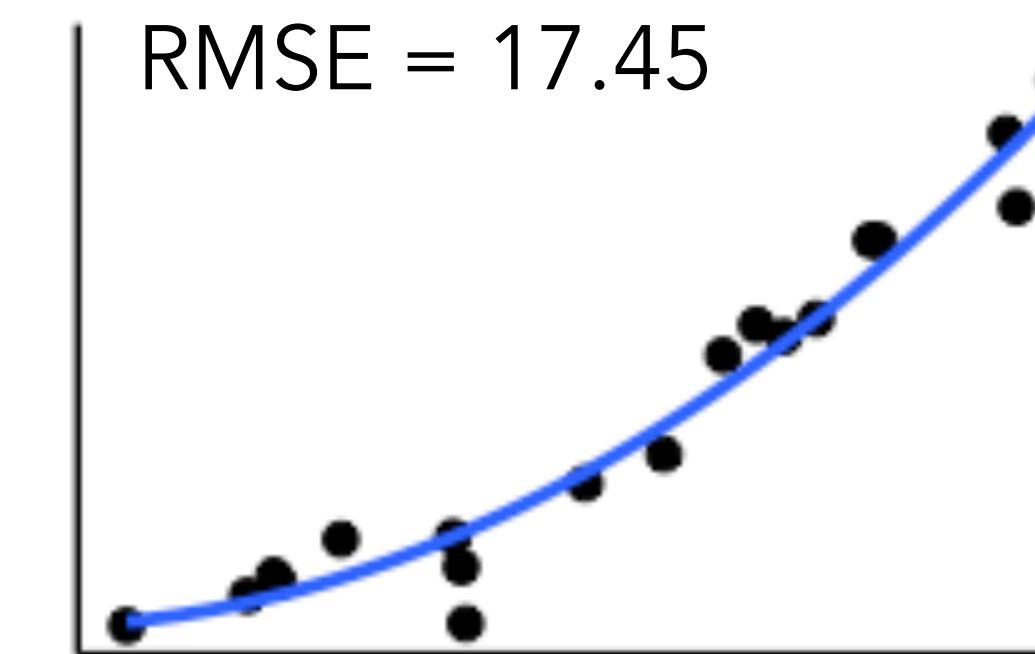
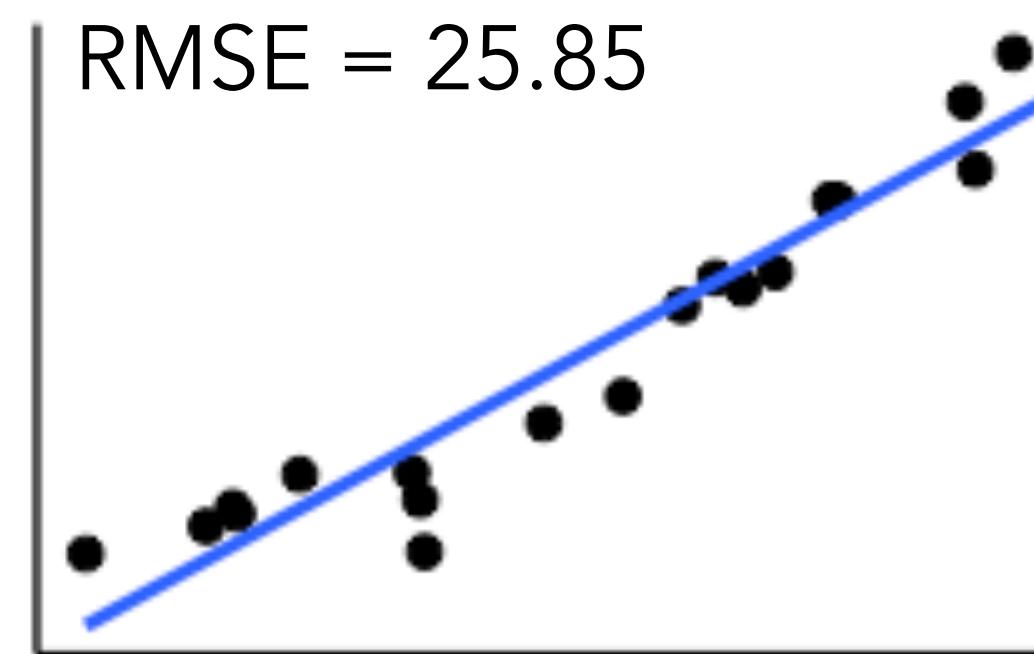


Tools for model comparison

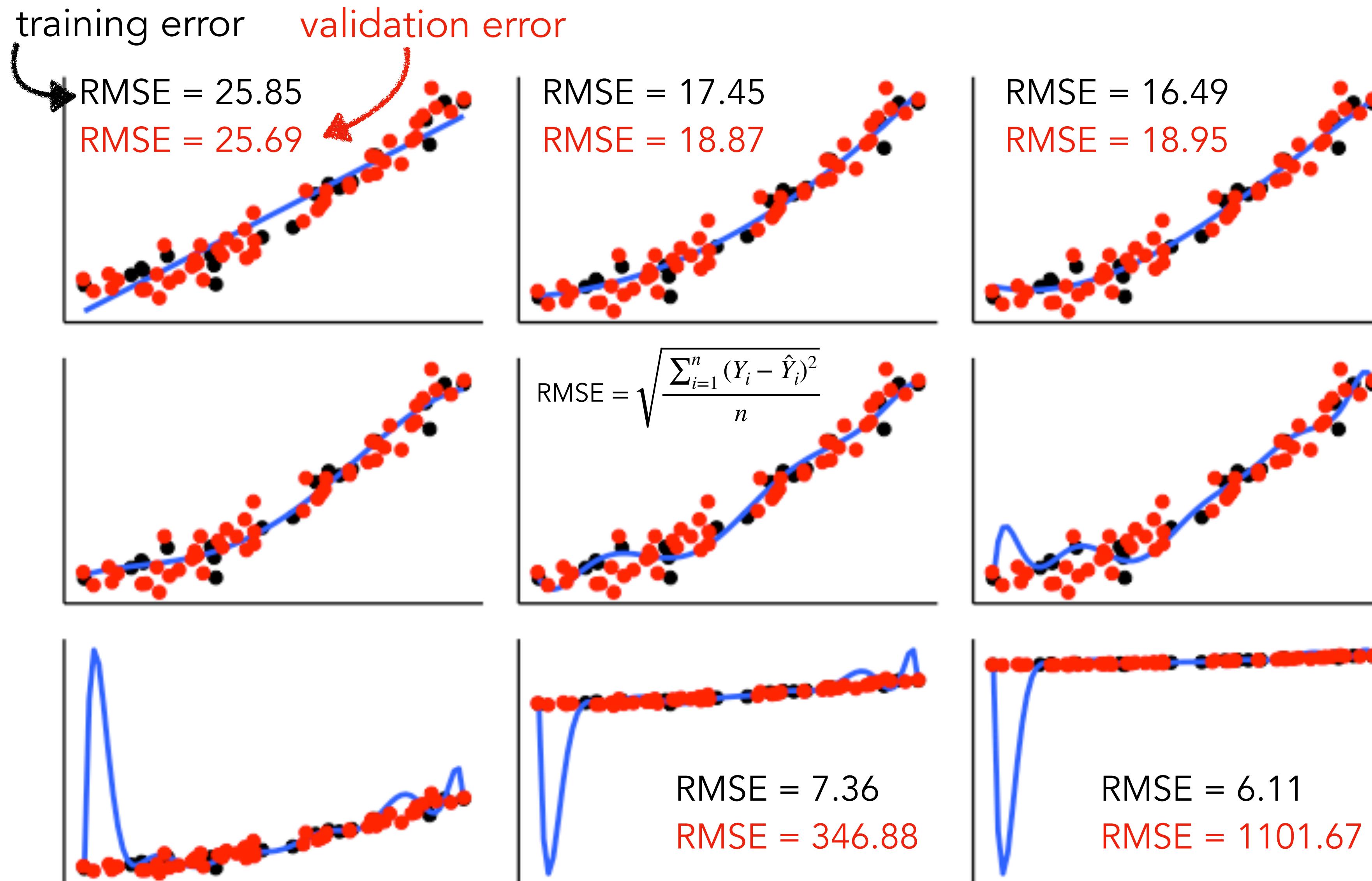
- **anova()** : compare a compact model with an augmented model via the F-test (or likelihood ratio test)
 - problem: only works for **nested models** (where the augmented model contains all the predictors of the compact model and more)
- **What if we want to compare models that are not nested?**
 - Cross-validation
 - AIC and BIC
 - Bayesian data analysis (we'll get there soon)

Cross-validation

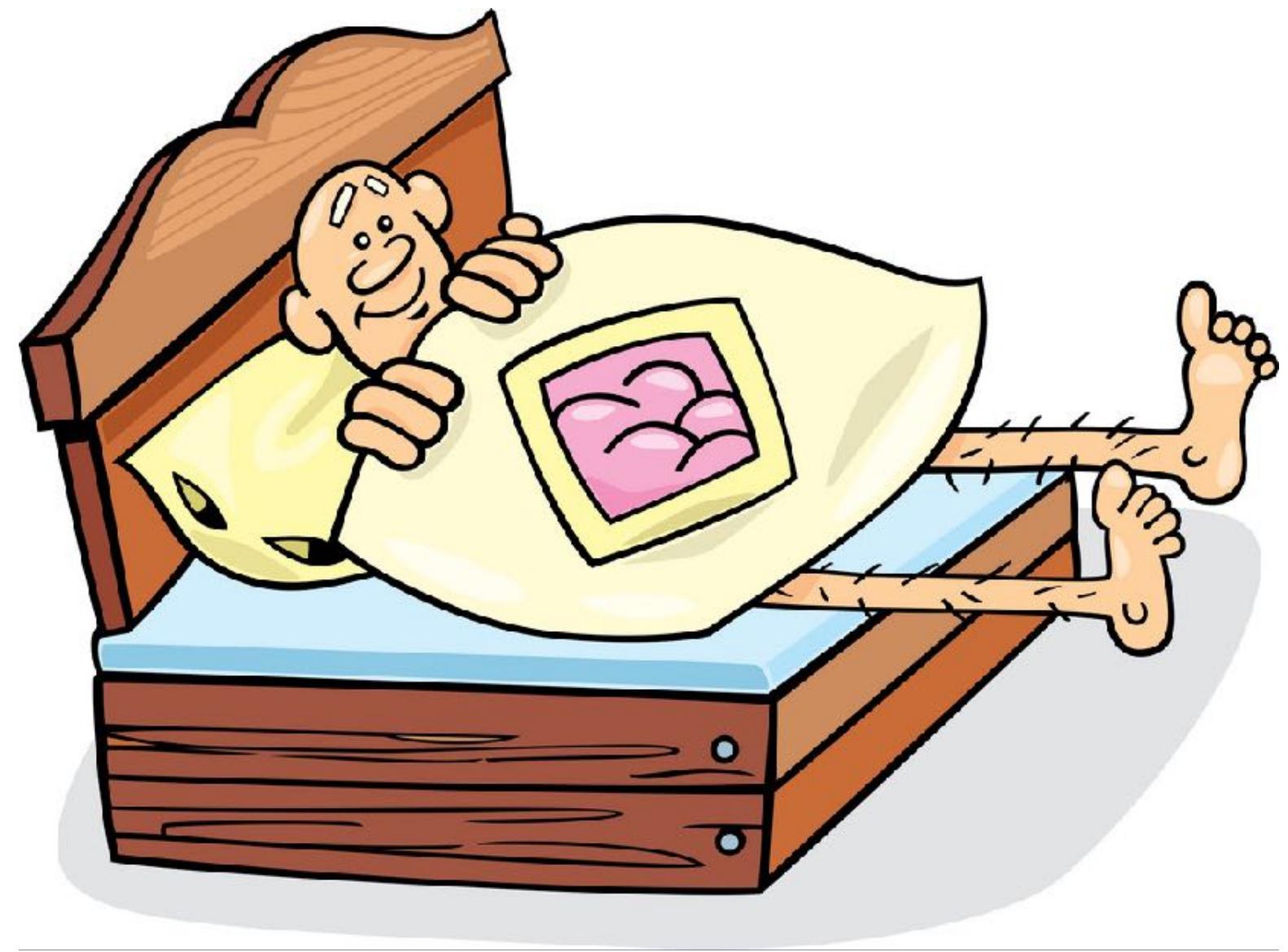
Which model describes the data best?



Which model describes the data best?



Underfitting vs. Overfitting



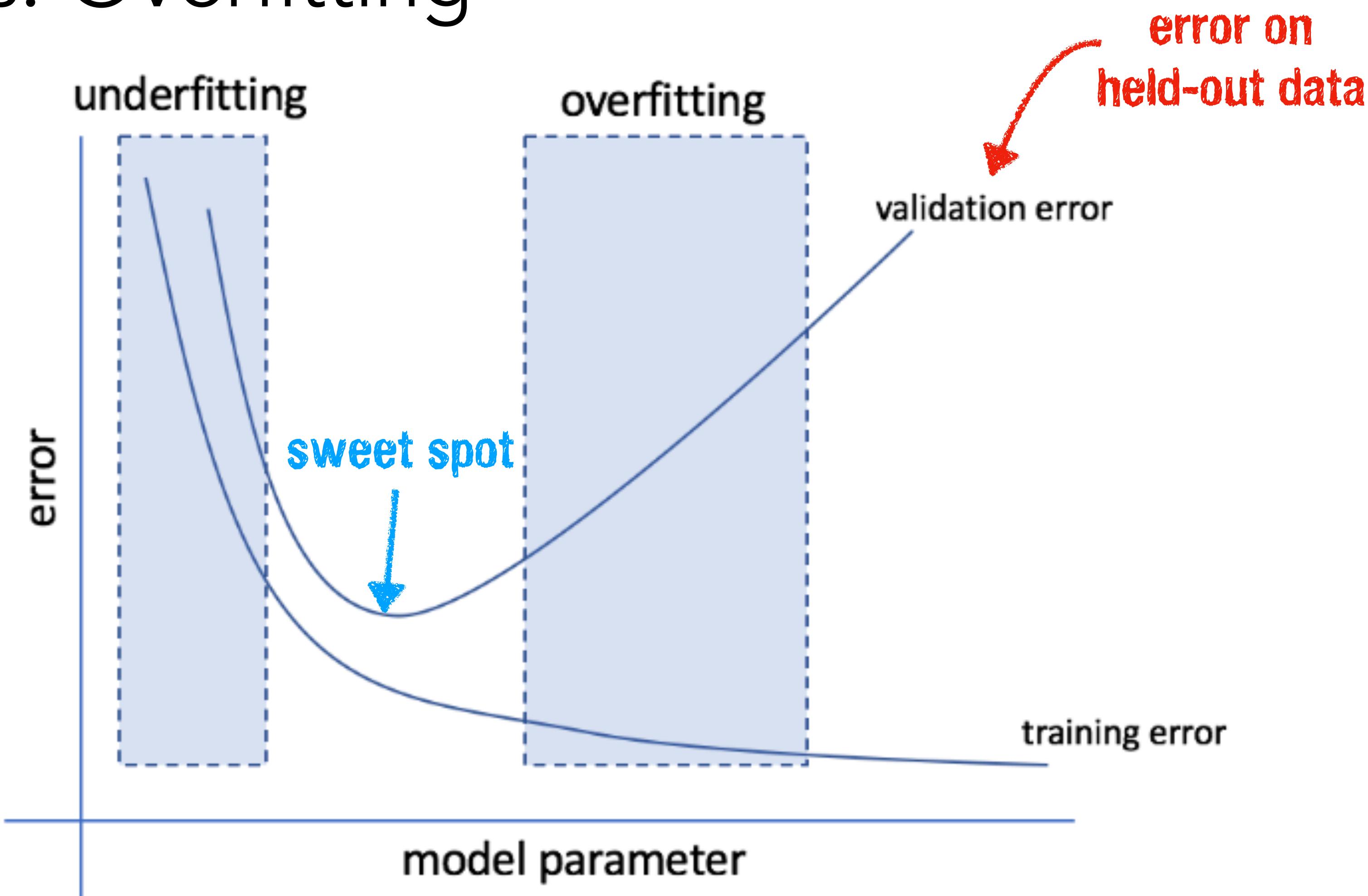
**ONE WAY TO EXPLAIN
UNDERFITTING**



Underfitting vs. Overfitting

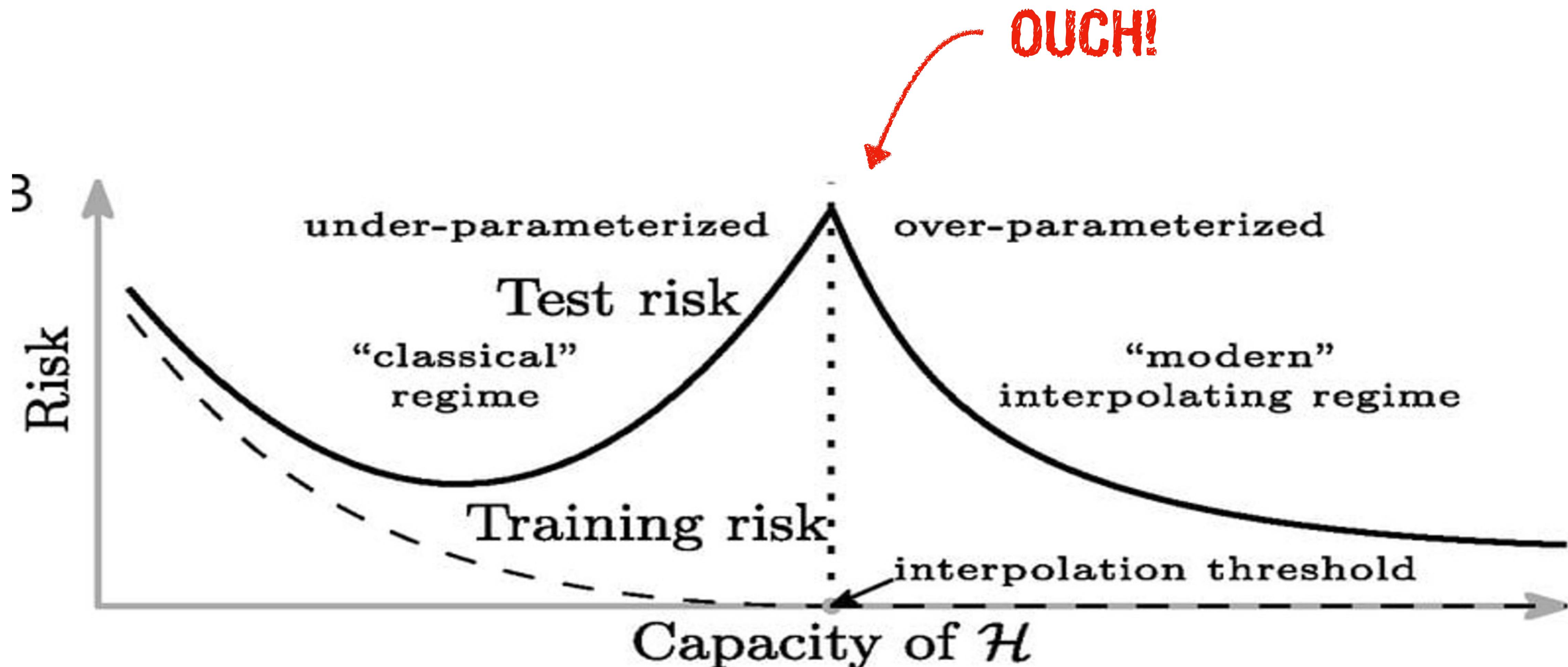
- a good model should:
 - explain the actual data well
 - predict future data well
- bias-variance tradeoff:
 - **bias** = error from erroneous assumptions in the model, high bias can cause a model to miss the relevant relations **underfitting** between predictors and outcome
 - **variance** = error from sensitivity to small fluctuations in the data, high variance can cause a model to fit the random **noise** in the data **overfitting**

Underfitting vs. Overfitting



the goal is to find the **sweet spot** between underfitting and overfitting

Discovery of Double-Descent and Over-parameterized Models



Leave-one-out crossvalidation



LOO

Leave One
Out

non-inspirational quote

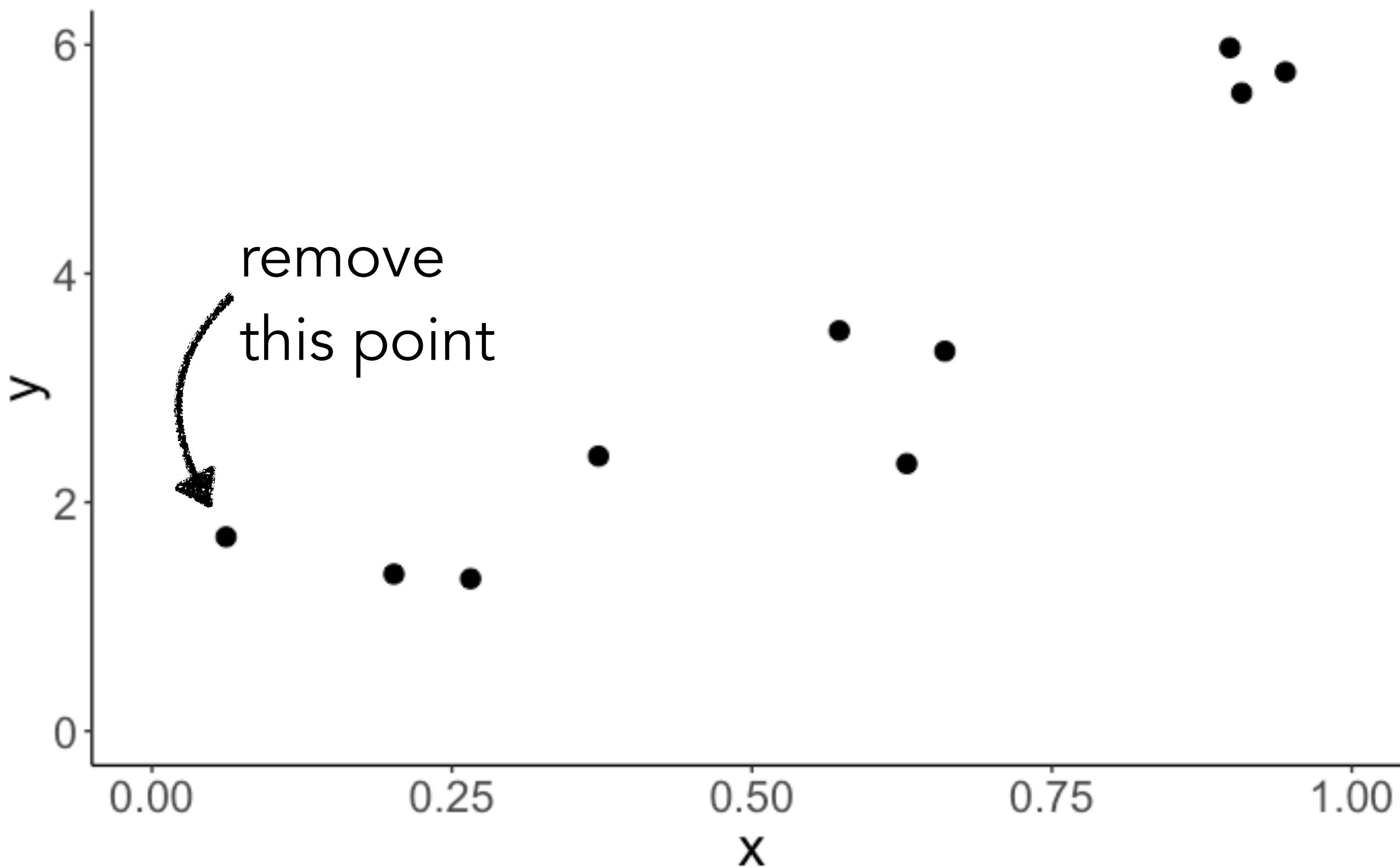
Leave one out cross-validation

- train the model on all the data points except for one
- calculate the prediction error for the held-out data point

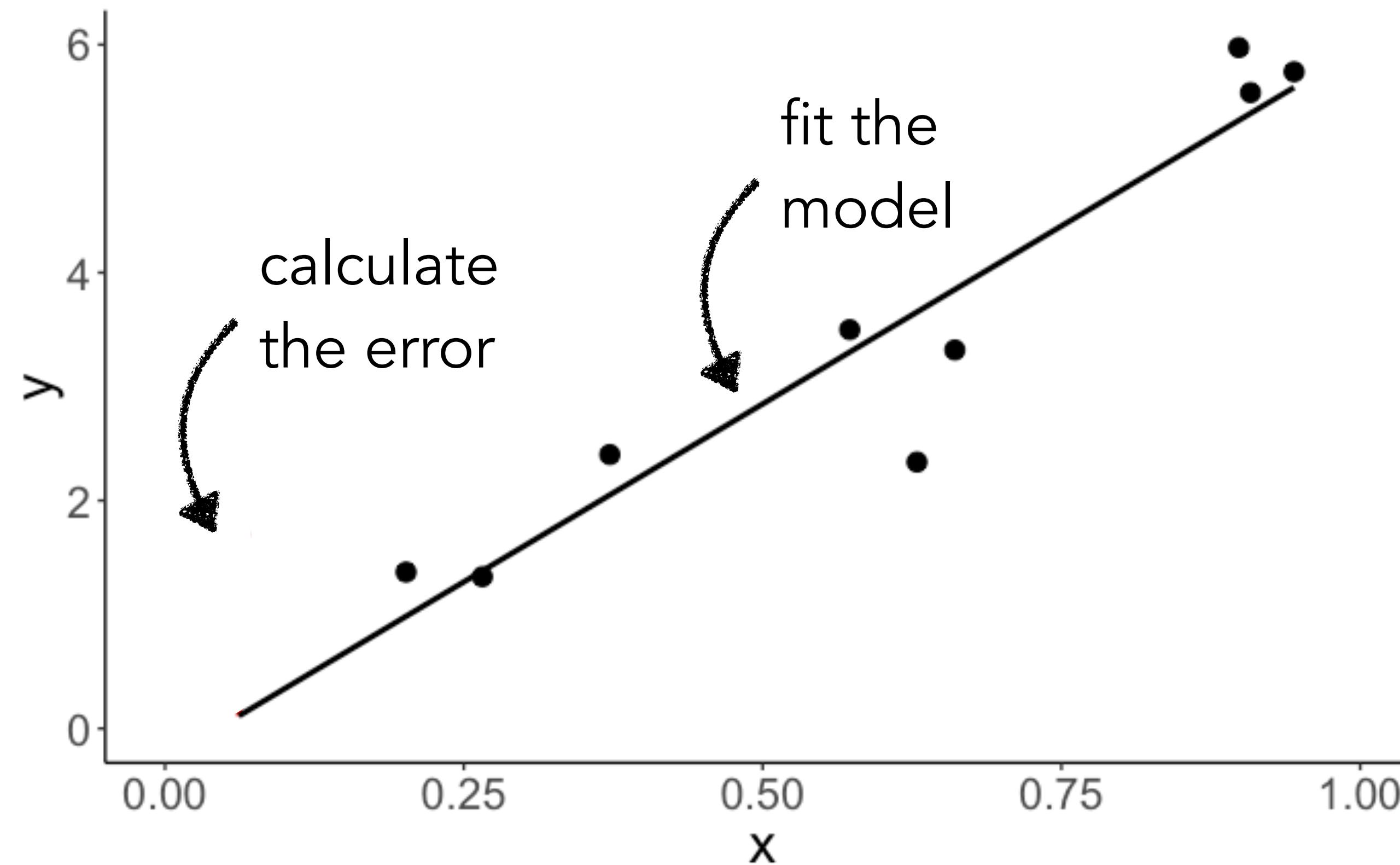


repeat for all
data points

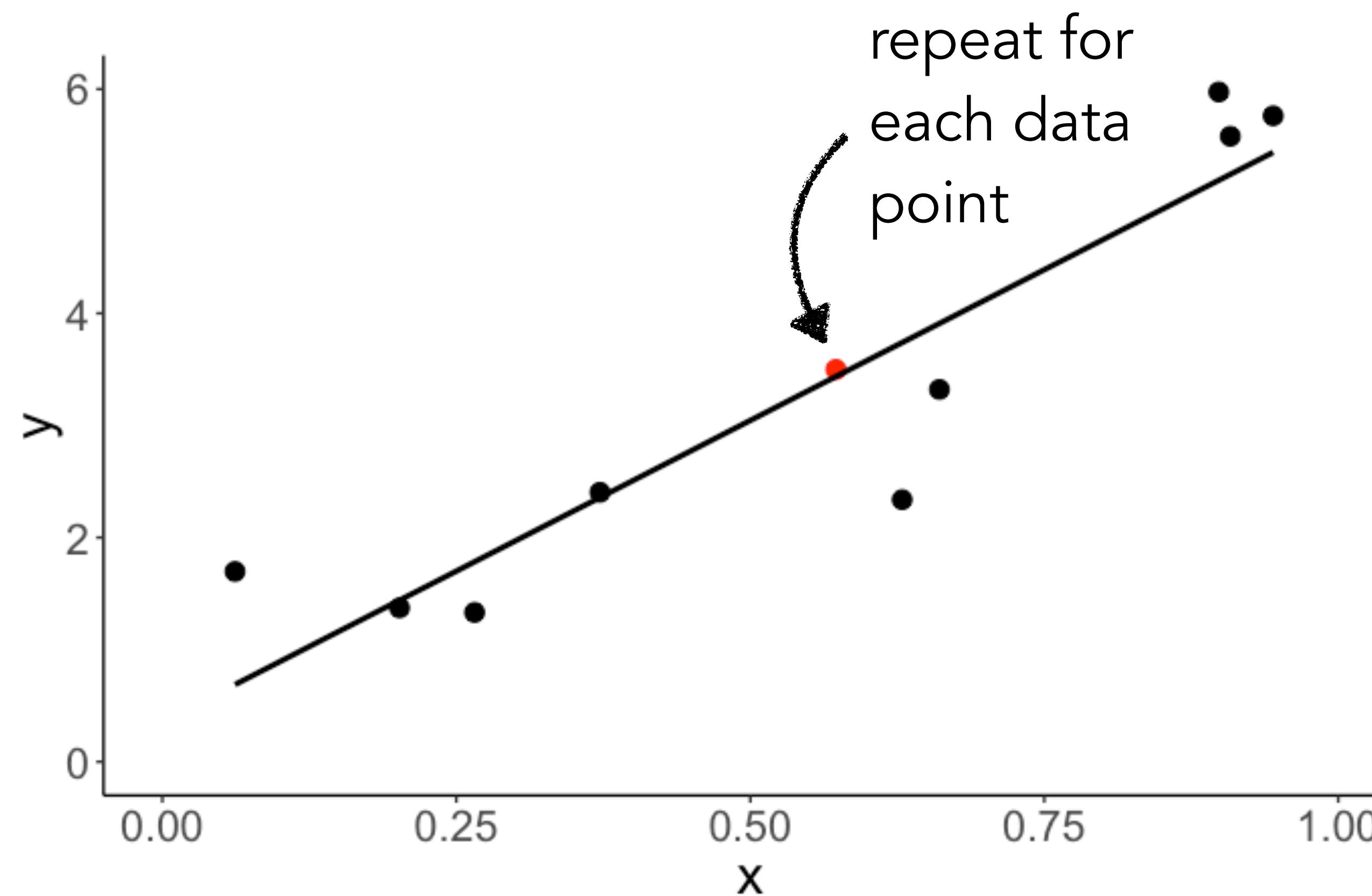
Leave one out cross-validation



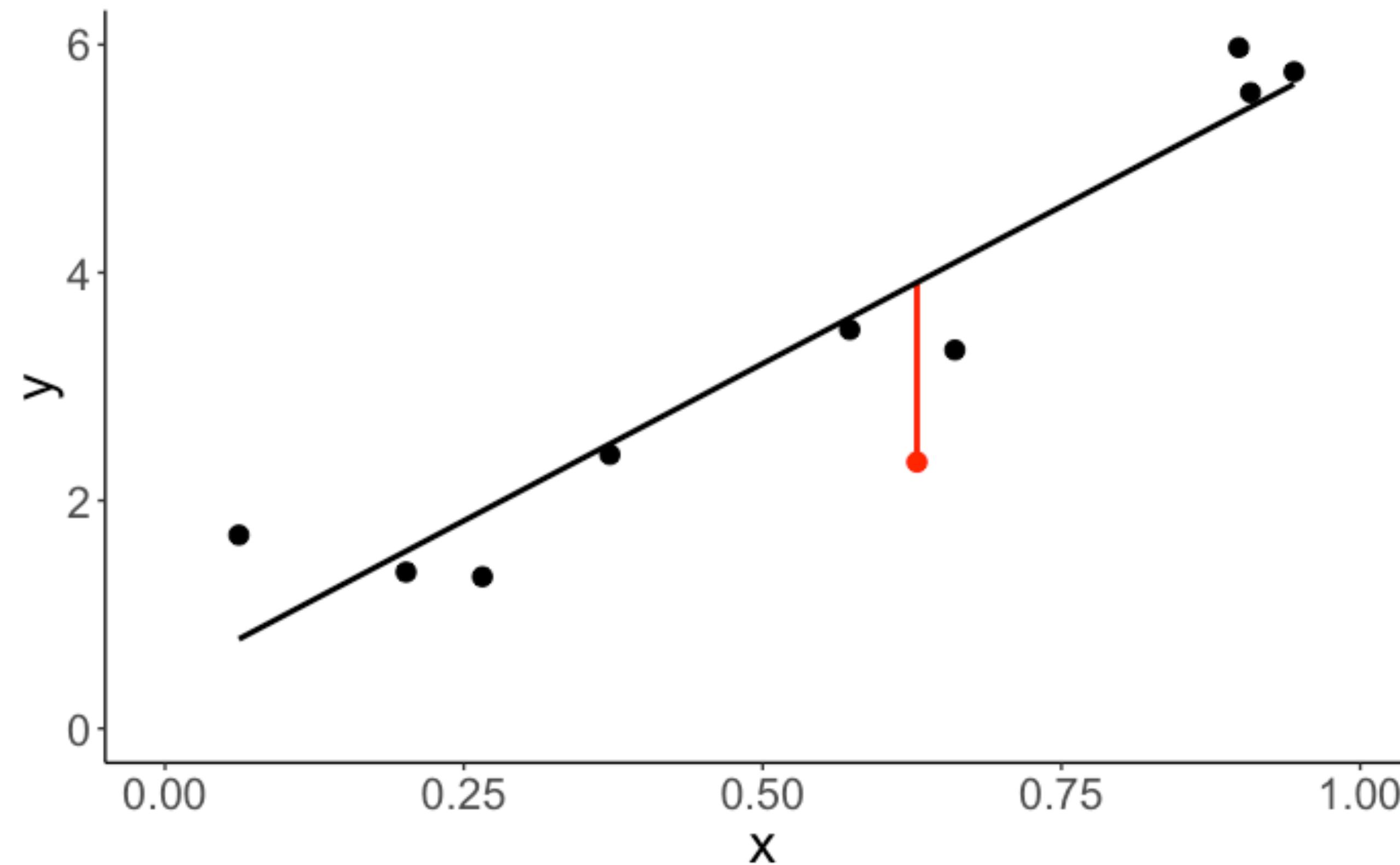
Leave one out cross-validation



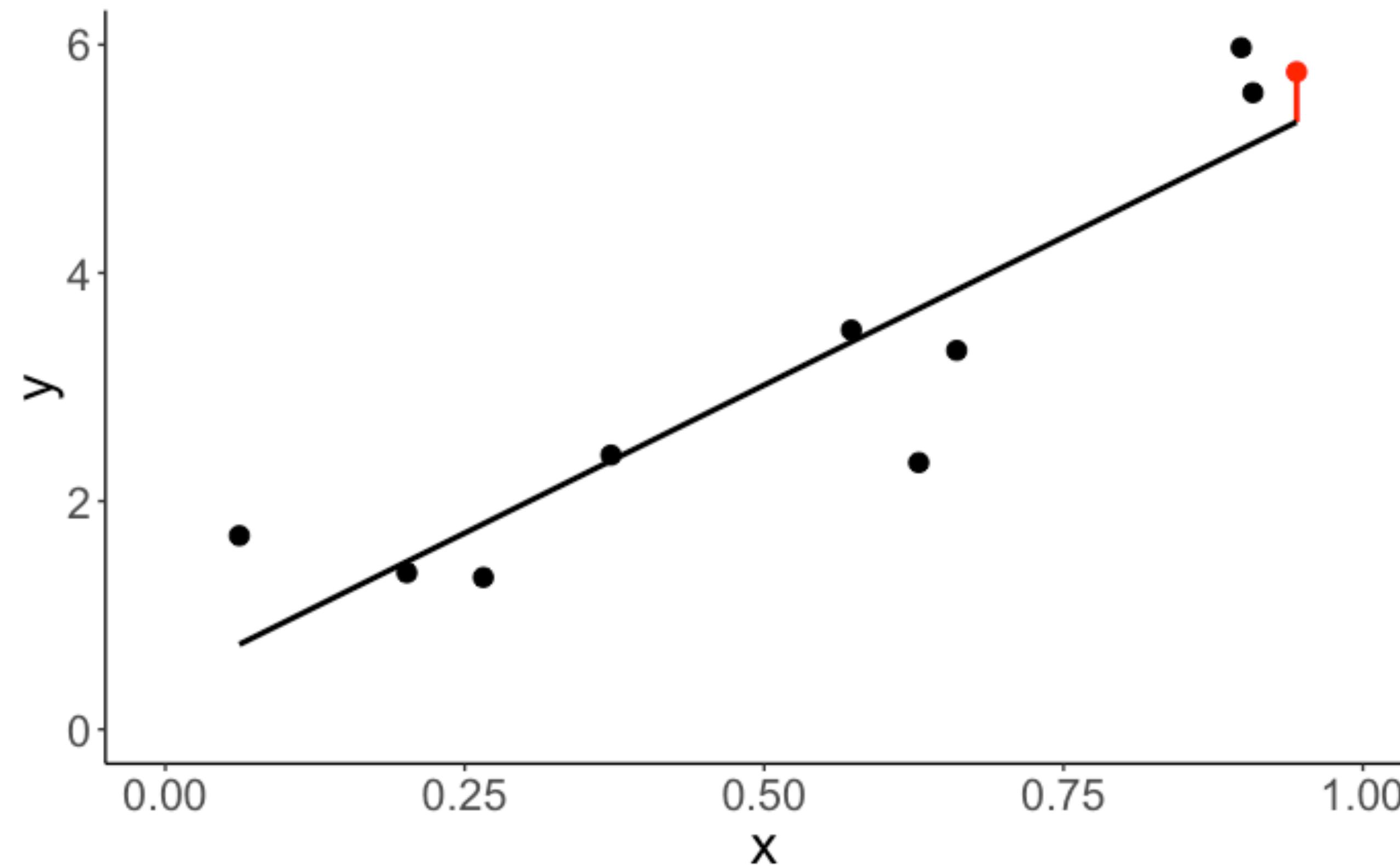
Leave one out cross-validation



Leave one out cross-validation

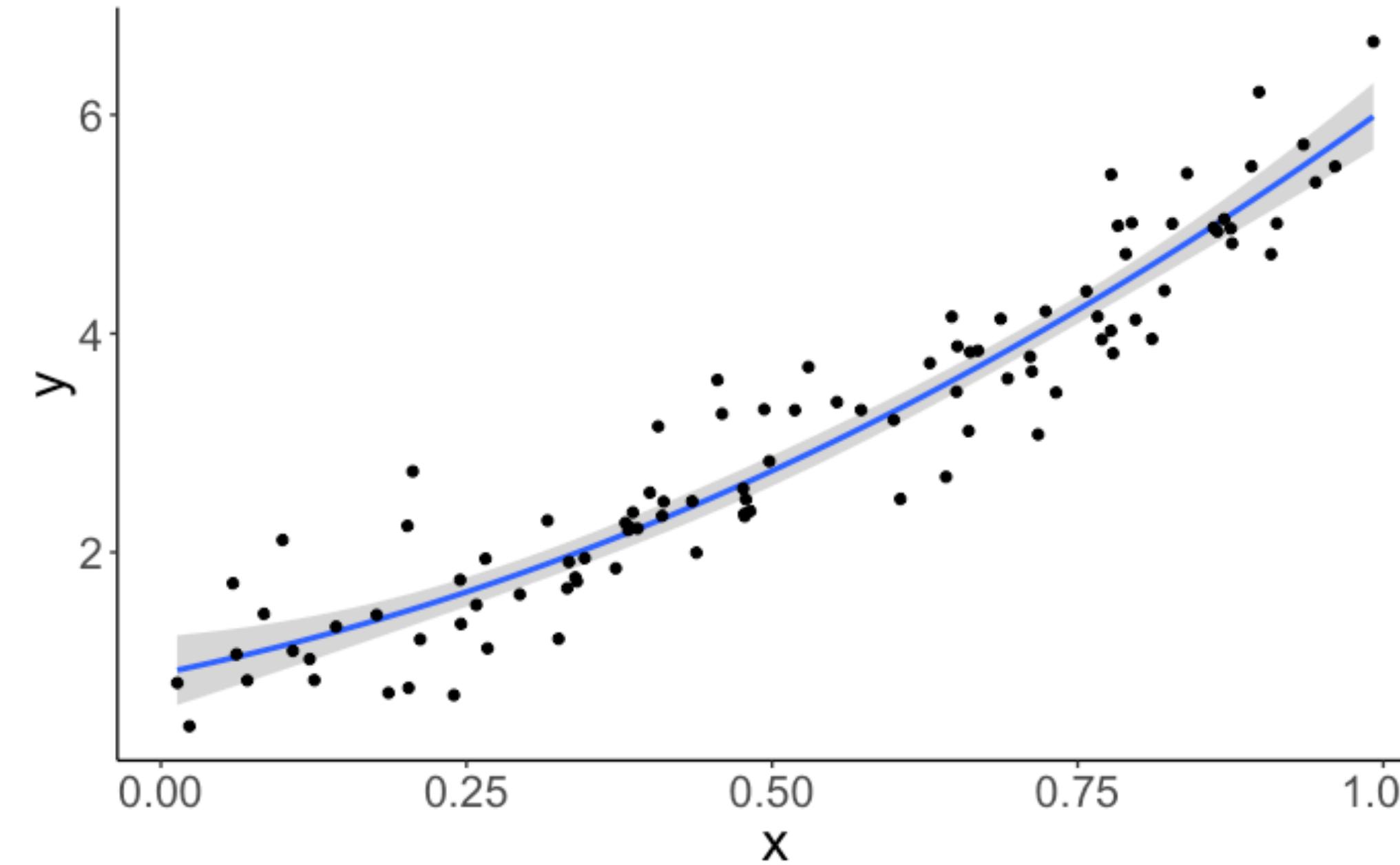


Leave one out cross-validation



Leave-one out crossvalidation

```
1 # make example reproducible
2 set.seed(1)
3
4 # parameters
5 sample_size = 100
6 b0 = 1
7 b1 = 2
8 b2 = 3
9 sd = 0.5
10
11 # sample
12 df.data = tibble(
13   participant = 1:sample_size,
14   x = runif(sample_size, min = 0, max = 1),
15   y = b0 + b1*x + b2*x^2 + rnorm(sample_size, sd = sd)
16 )
```



Leave-one out crossvalidation

ground truth

$$y_i = 1 + 2 \cdot x_i + 3 \cdot x_i^2 + e$$

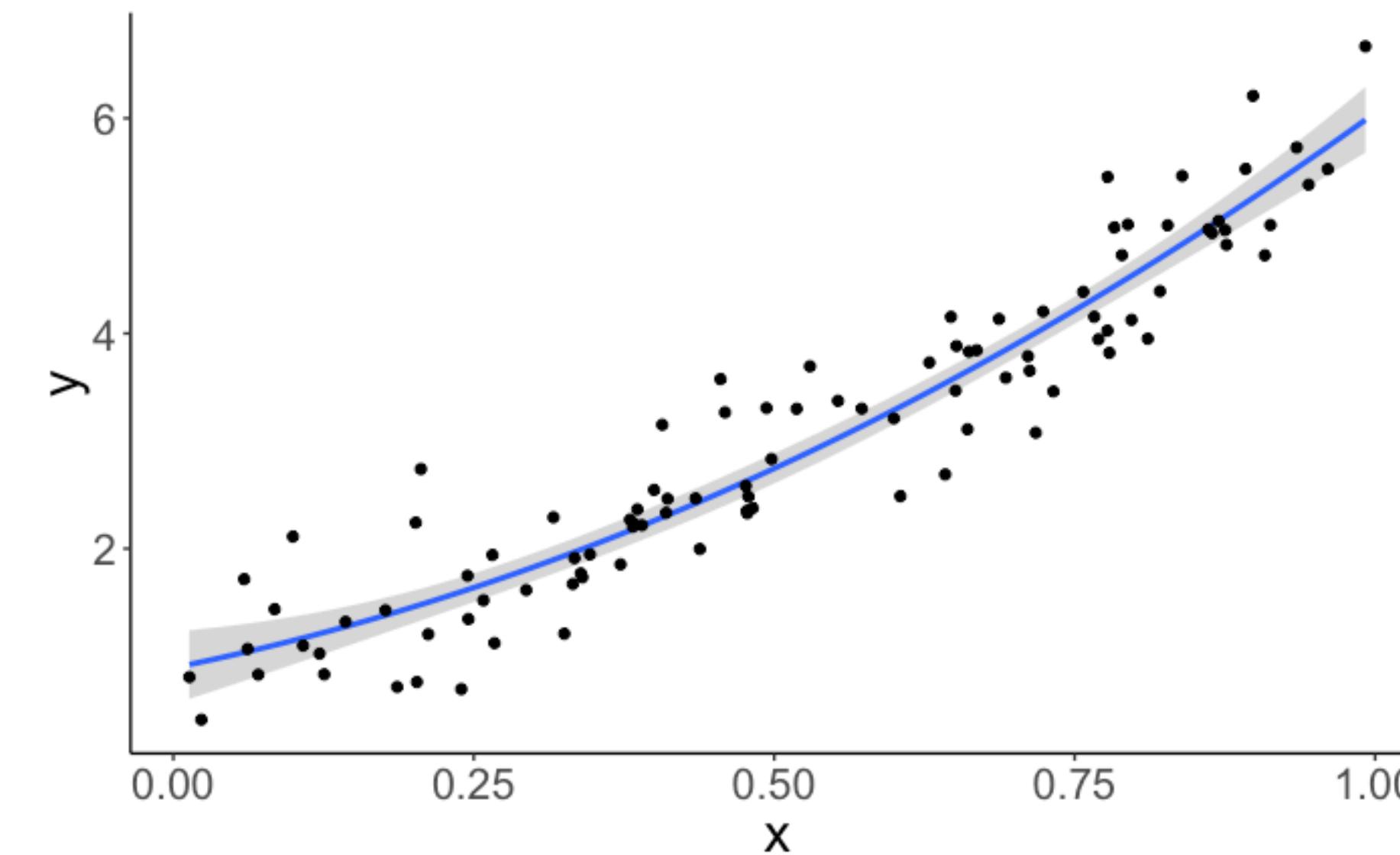
$$e \sim \mathcal{N}(\text{mean} = 0, \text{sd} = 0.5)$$

candidate models

simple $\hat{y}_i = b_0 + b_1 \cdot x_i$

correct $\hat{y}_i = b_0 + b_1 \cdot x_i + b_2 \cdot x_i^2$

complex $\hat{y}_i = b_0 + b_1 \cdot x_i + b_2 \cdot x_i^2 + b_3 \cdot x_i^3$



we could do an F-test
here since the models
are nested ...

Leave-one out crossvalidation

```
1 library("modelr") ← nice package for basic cross-validation  
2  
3 df.cross = df.data %>%  
4   crossv_loo() %>% # function which generates training and test data sets  
5   mutate(model_simple = map(train, ~ lm(y ~ 1 + x, data = .)),  
6         model_correct = map(train, ~ lm(y ~ 1 + x + I(x^2), data = .)),  
7         model_complex = map(train, ~ lm(y ~ 1 + x + I(x^2) + I(x^3), data = .))) %>%  
8   pivot_longer(cols = contains("model"),  
9                 names_to = "index",  
10                values_to = "model") %>%  
11  mutate(rmse = map2_dbl(.x = model, .y = test, .f = ~ rmse(.x, .y)))
```

index	mean_rmse
simple	0.65
correct	0.42
complex	0.41

complex model has the lowest error on the training data

Leave-one out crossvalidation

```
1 library("modelr")
2
3 df.cross = df.data %>%
4   crossv_loo() %>%
```

splits the data set into training and test

	train	test	.id
1	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	1
2	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	2
3	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	3
4	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	4
5	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	5
6	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	6
7	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	7
8	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	8
9	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	9
10	list(data = list(participant = 1:10, x = c(0.265508663...))	list(data = list(participant = 1:10, x = c(0.265508663...))	10

each entry is simply a pointer to the data set plus some indices .idx for the filtered data points

Leave-one out crossvalidation

```
1 library("modelr")
2
3 df.cross = df.data %>%
4   crossv_loo() %>%
5   mutate(model_simple = map(train, ~ lm(y ~ 1 + x, data = .)),
6         model_correct = map(train, ~ lm(y ~ 1 + x + I(x^2), data = .)),
7         model_complex = map(train, ~ lm(y ~ 1 + x + I(x^2) + I(x^3), data = .))) %>%
```

fit three different models to the training data set

Leave-one out crossvalidation

```
1 library("modelr")
2
3 df.cross = df.data %>%
4   crossv_loo() %>%
5   mutate(model_simple = map(train, ~ lm(y ~ 1 + x, data = .)),
6         model_correct = map(train, ~ lm(y ~ 1 + x + I(x^2), data = .)),
7         model_complex = map(train, ~ lm(y ~ 1 + x + I(x^2) + I(x^3), data = .))) %>%
8   pivot_longer(cols = contains("model"),
9                 names_to = "index",
10                values_to = "model") %>%
11   mutate(rmse = map2_dbl(.x = model, .y = test, .f = ~ rmse(.x, .y)))
```

calculate the root mean squared error for each model on the test data set

```
1 df.cross %>%
2   group_by(index) %>%
3   summarize(mean_rmse = mean(rmse))
```

index	mean_rmse
simple	0.65
correct	0.48
complex	0.70

the correct model has the lowest prediction error

Leave-one out crossvalidation

Any potential problems with the LOO?

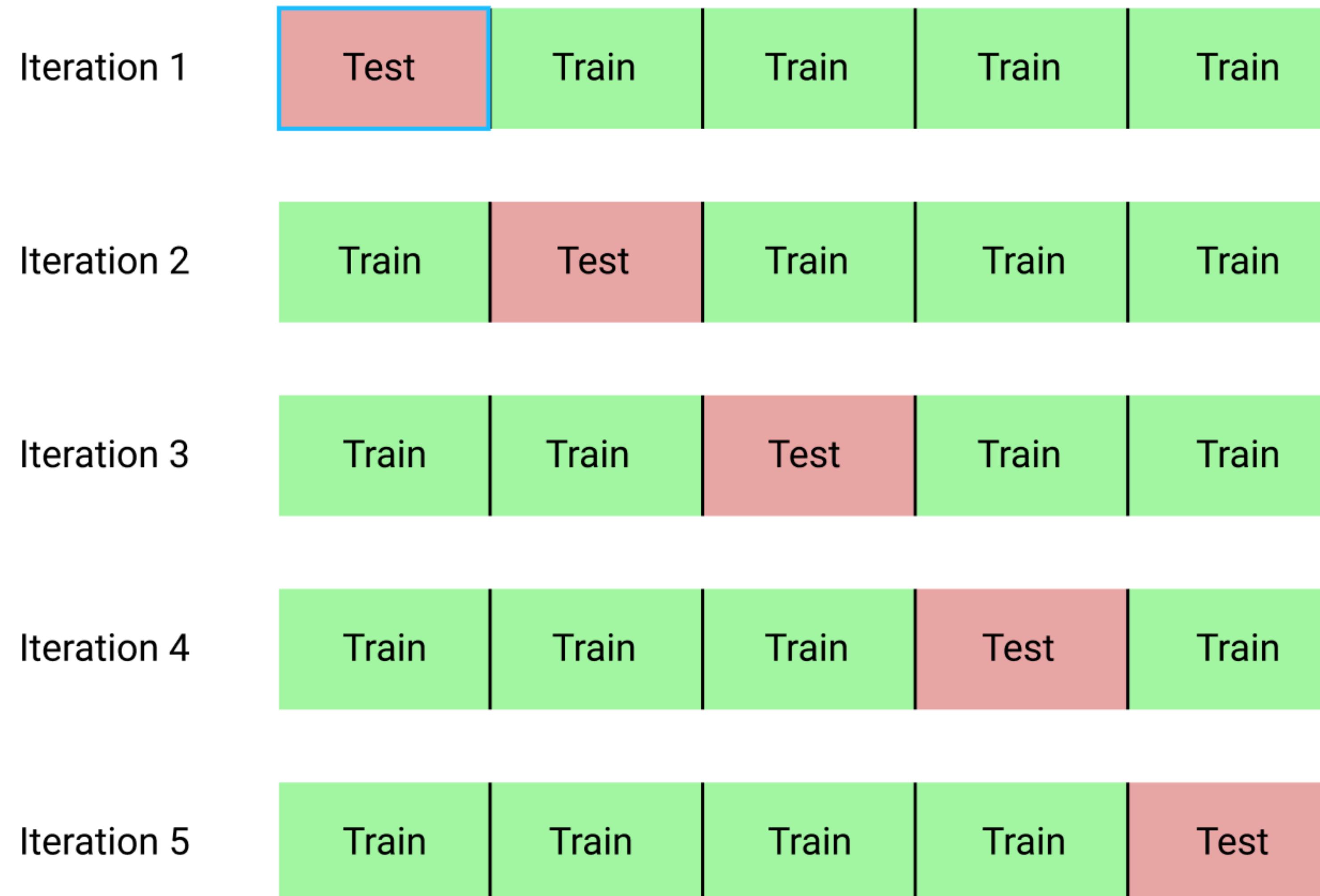
can be computationally expensive since it requires fitting
the model n times (once for each data point) ...



k-fold cross validation

k-fold crossvalidation

Full data set



k-fold crossvalidation

k-fold crossvalidation

```
1 df.cross = df.data %>%
2   crossv_kfold(k = 10) %>%
3   mutate(model_simple = map(.x = train,
4     .f = ~ lm(y ~ 1 + x, data = .)),
5   model_correct = map(.x = train,
6     .f = ~ lm(y ~ 1 + x + I(x^2), data = .)),
7   model_complex = map(.x = train,
8     .f = ~ lm(y ~ 1 + x + I(x^2) + I(x^3), data = .)) %>%
9   pivot_longer(cols = contains("model"),
10    names_to = "model",
11    values_to = "fit") %>%
12   mutate(rsquare = map2_dbl(.x = fit,
13     .y = test,
14     .f = ~ rsquare(.x, .y)))
```

using R² as a measure

why didn't we use R² for LOO?

this wouldn't work for LOO
since we only have one data
point in the test data...

index	median_rsquare
simple	0.839
correct	0.865
complex	0.860

the correct model accounts for
the most variance in the test data

k-fold vs. leave-one-out crossvalidation

- LOO:

- trained on **more** data
- more variance
- less bias

more overfitting
less underfitting

- k-fold:

- trained on **less** data
- less variance
- more bias

less overfitting
more underfitting

- **bias** = error from erroneous model assumptions, miss the relevant relations
- **variance** = error from small fluctuations in the data, fitting the random **noise**

Monte Carlo crossvalidation

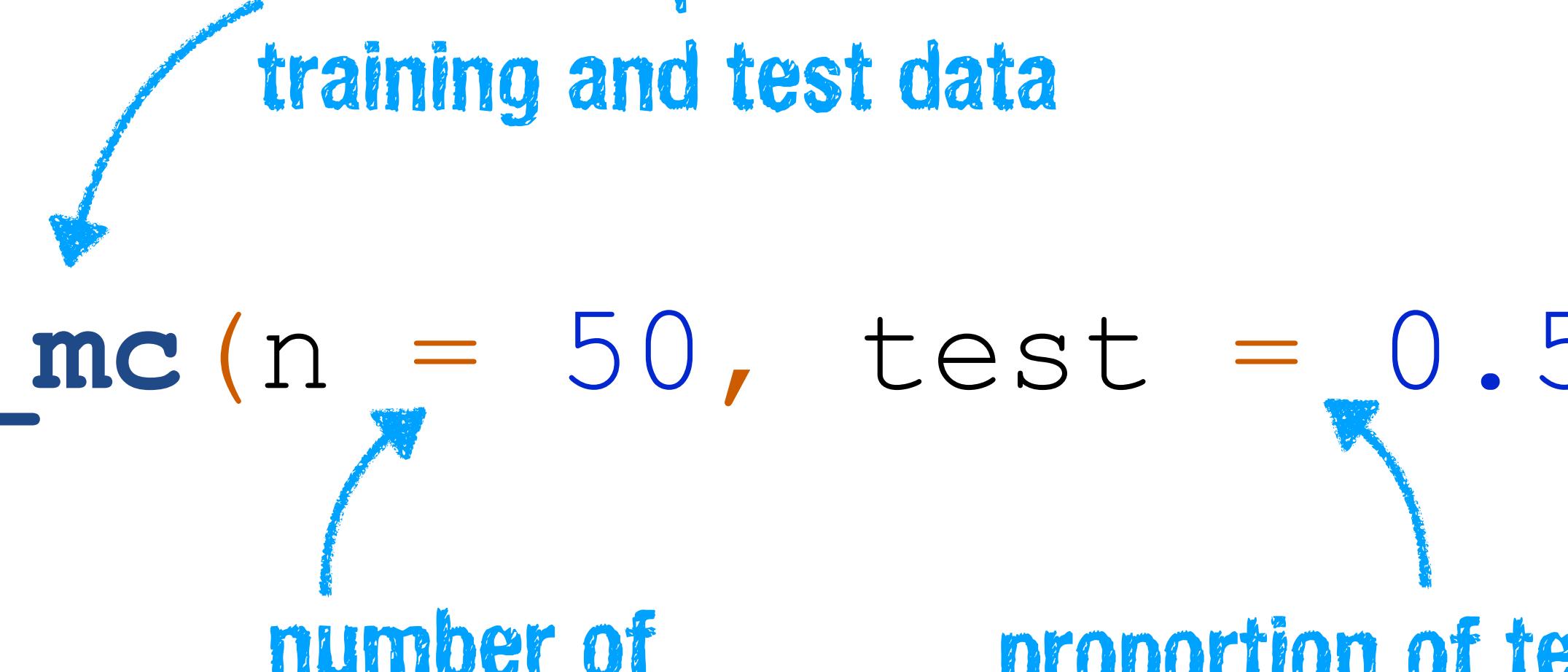
Monte Carlo crossvalidation

`crossv_mc(n = 50, test = 0.5)`

random splits into
training and test data

number of
training-test splits

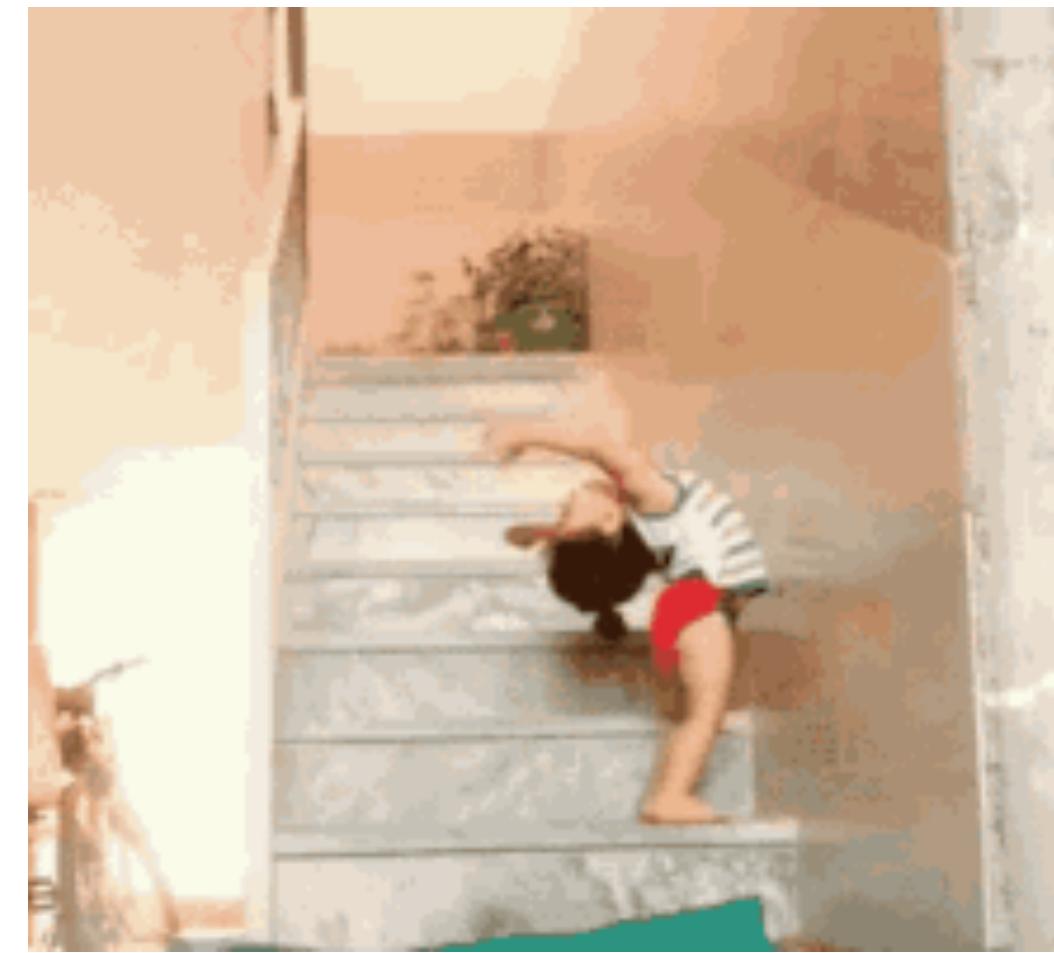
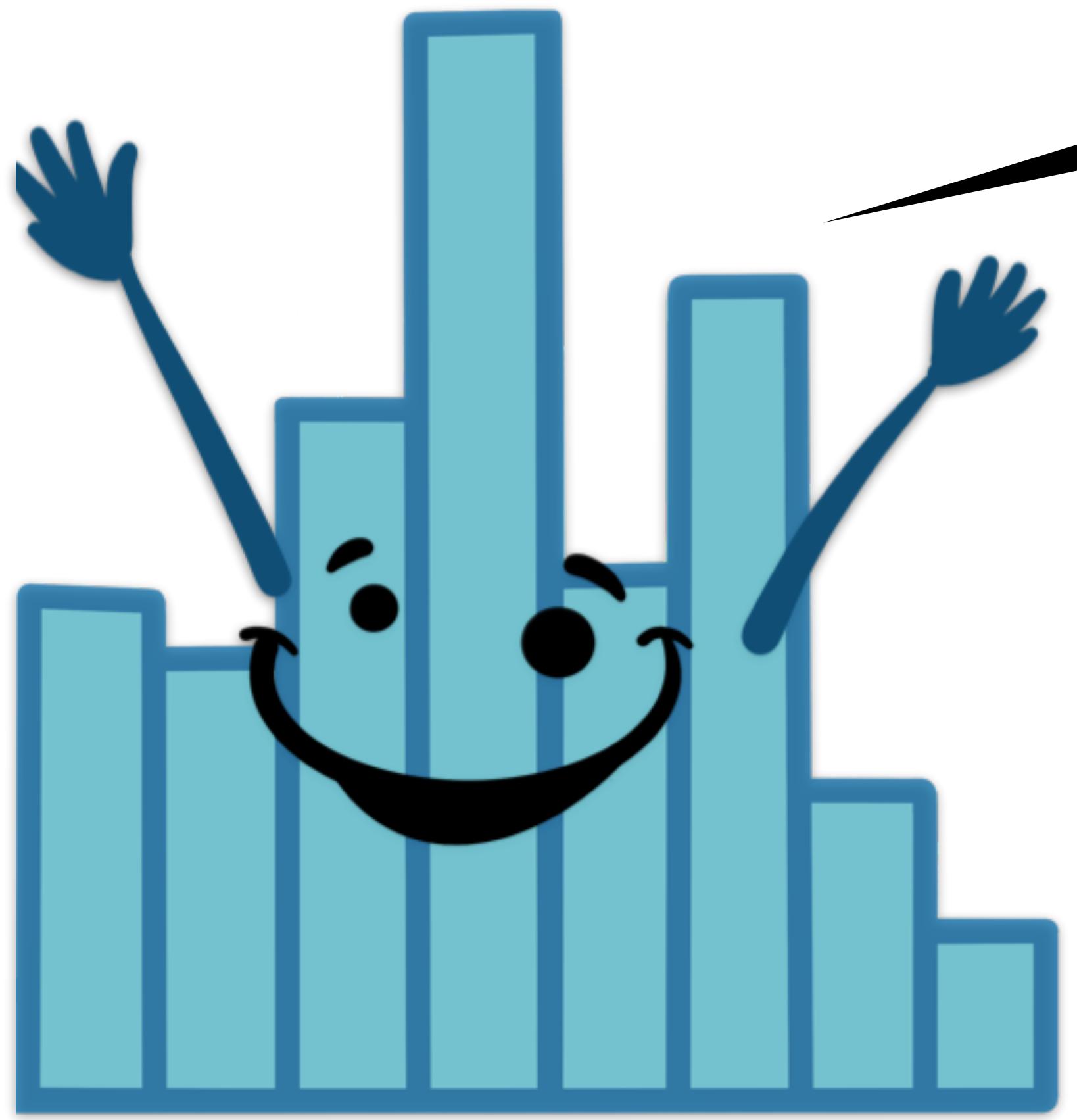
proportion of test data
in each split



- splits can also be done in a **stratified** way (check out the `tidymodels` package)
- for example, generate training data that has the same percentage of cases from each group
- fit data from some participants, test on data from other participants, ...

02:00

stretch break!





AIC and BIC

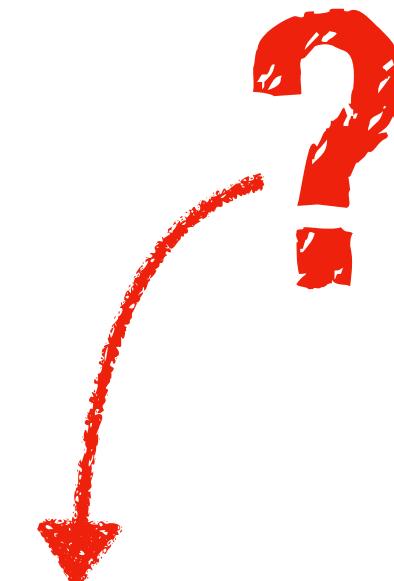
AIC and BIC

- AIC = Akaike Information Criterion
- BIC = Bayesian Information Criterion

not that much Bayesian about it ...

$$\text{AIC} = 2k - 2 \ln(\hat{L})$$

$$\text{BIC} = \ln(n)k - 2 \ln(\hat{L})$$



\hat{L} = maximized value of the likelihood function of the model

k = number of parameters in the model

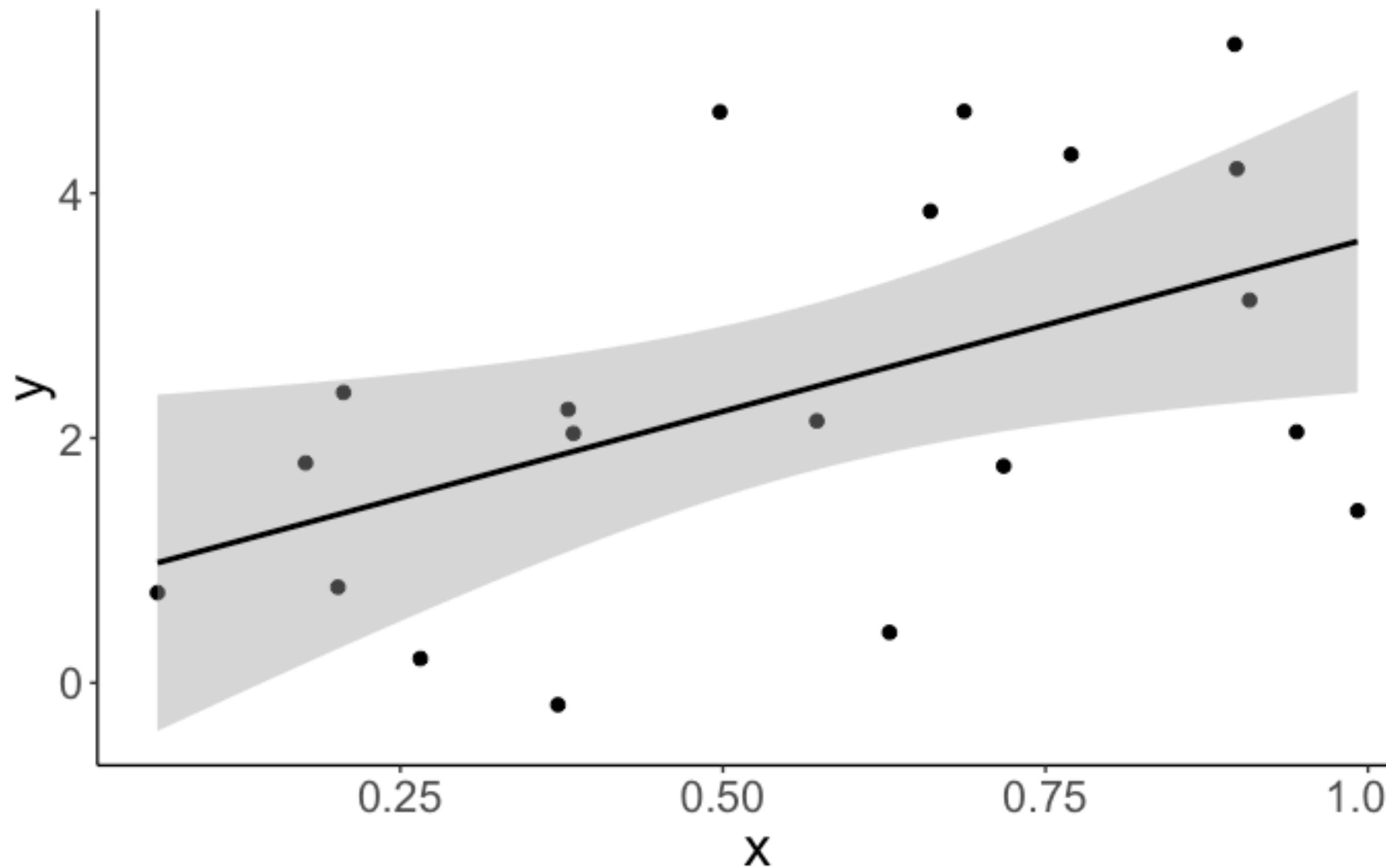
n = number of observations

AIC and BIC

- How do we get the likelihood of our model?
 - in a linear regression, minimizing least squares is equivalent to maximizing the likelihood of the data given the model
- Assumptions of the linear model:
 - residuals are normally distributed with:
 - mean = 0 and sd = sigma
 - calculate overall likelihood by computing the likelihood of each residual, and then multiplying

AIC and BIC

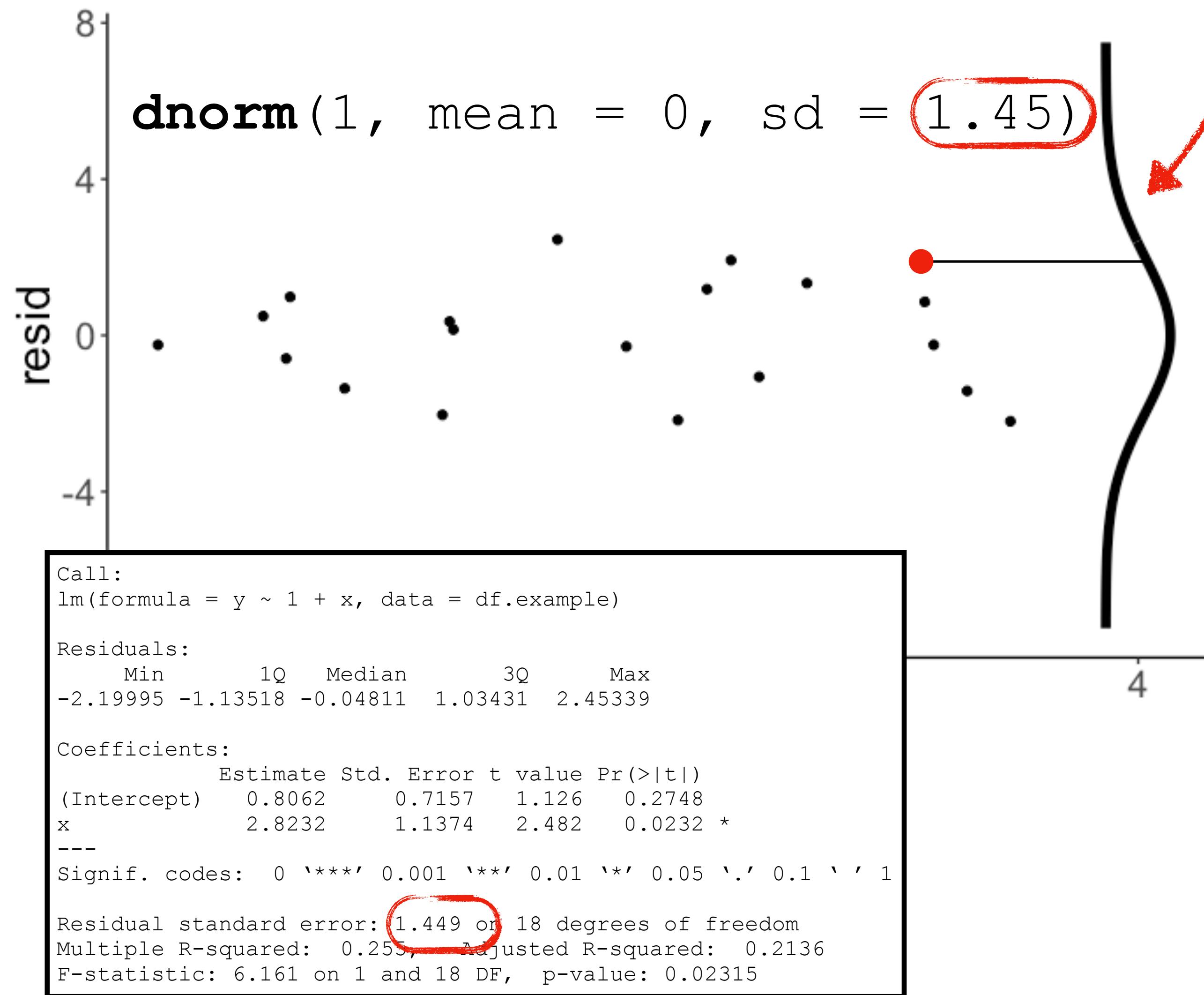
data with linear model fit



AIC and BIC

normal distribution
over residuals

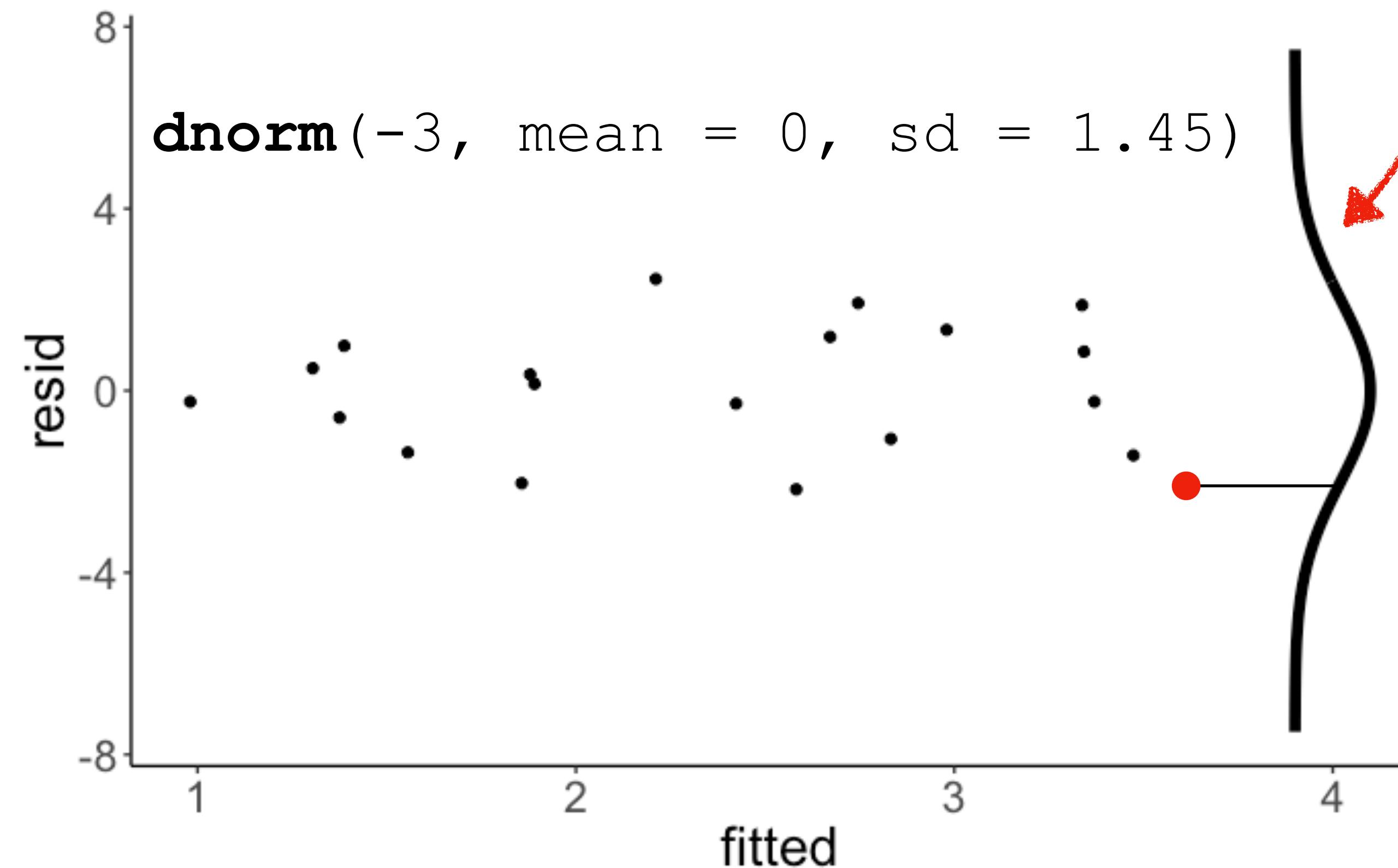
Residual plot



AIC and BIC

Residual plot

normal distribution
over residuals

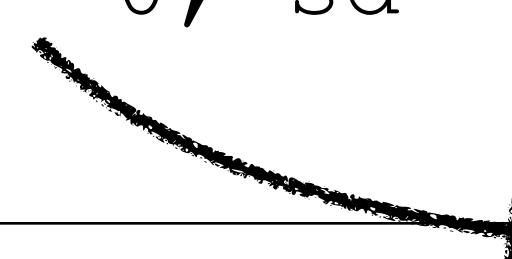


since the data points are independent, we can calculate the overall likelihood by multiplying the likelihood of each observation

AIC and BIC

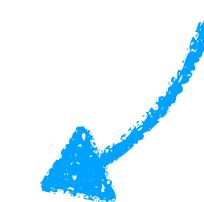
```
1 # generate some data
2 df.like = tibble(
3   x = runif(20, min = 0, max = 1),
4   y = 1 + 3 * x + rnorm(20, sd = 2)
5 )
6
7 # fit the model
8 fit = lm(formula = y ~ x,
9           data = df.like)
10
11 # model summary
12 fit %>%
13   glance()
```

`dnorm(1.88, mean = 0, sd = 1.45) = 0.12`



x	y	fitted	resid	likelihood
0.90	5.22	3.34	1.88	0.12
0.27	0.20	1.56	-1.36	0.18
0.37	-0.18	1.86	-2.04	0.10
0.57	2.14	2.42	-0.28	0.27
0.91	3.13	3.37	-0.24	0.27
0.20	0.78	1.38	-0.59	0.25
0.90	4.20	3.34	0.86	0.23
0.94	2.05	3.47	-1.42	0.17
0.66	3.85	2.67	1.18	0.20
0.63	0.41	2.58	-2.17	0.09

inferred standard
deviation of the error

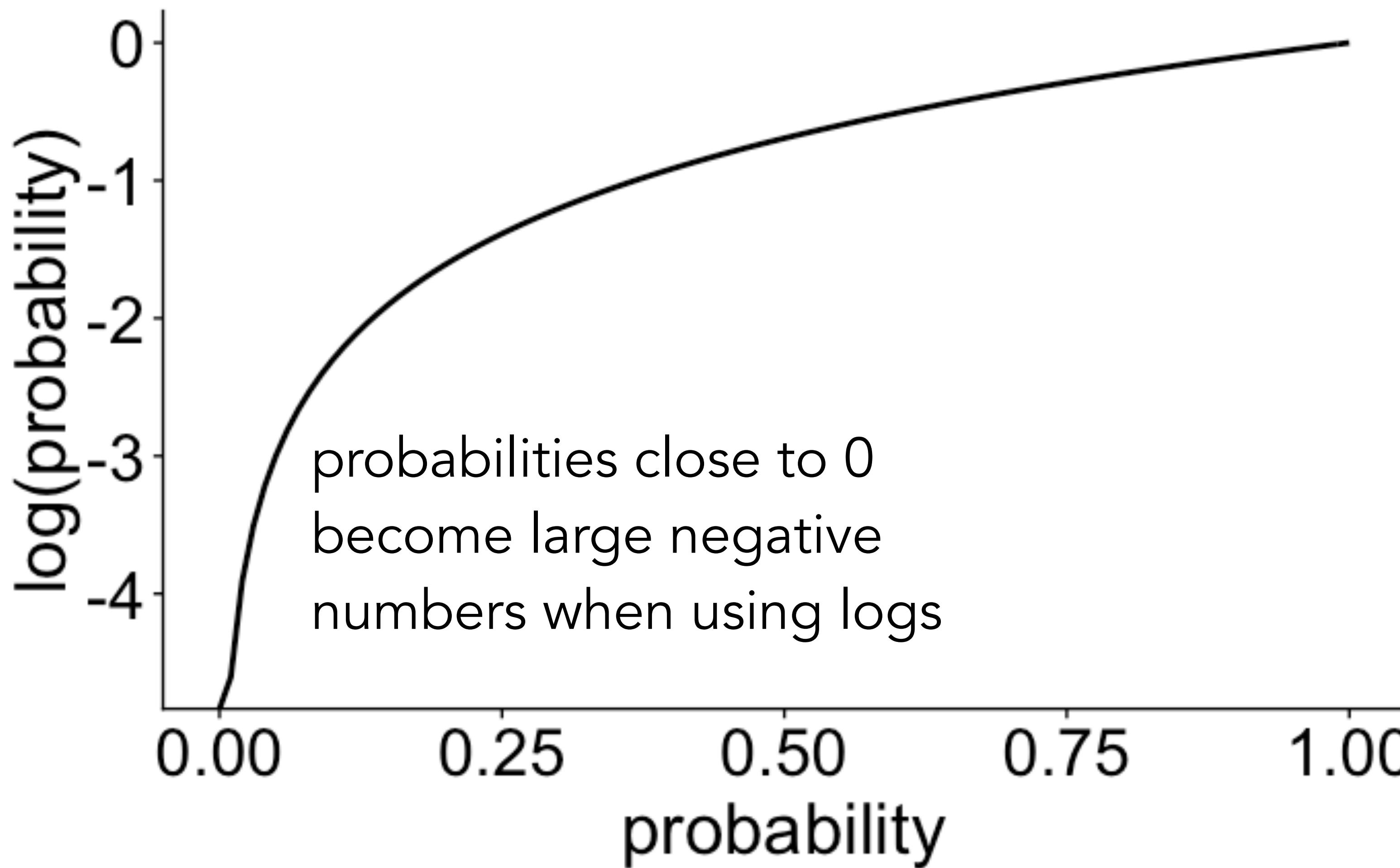


r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.25	0.21	1.45	6.16	0.02	2	-34.74	75.47	78.46	37.77	18

$e \sim \mathcal{N}(\text{mean} = 0, \text{sd} = 1.45)$


$$\sum_{i=1}^n \ln(\text{likelihood})$$

`log()` is your friend!



Clue guide to probability

- *joint probability:*
- if A and B are independent then
- **Definition:** $p(A, B) = p(A) \cdot p(B)$



- $p(\text{Prof Plum, candle stick}) =$
 $p(\text{Prof Plum}) \cdot p(\text{candle stick}) =$

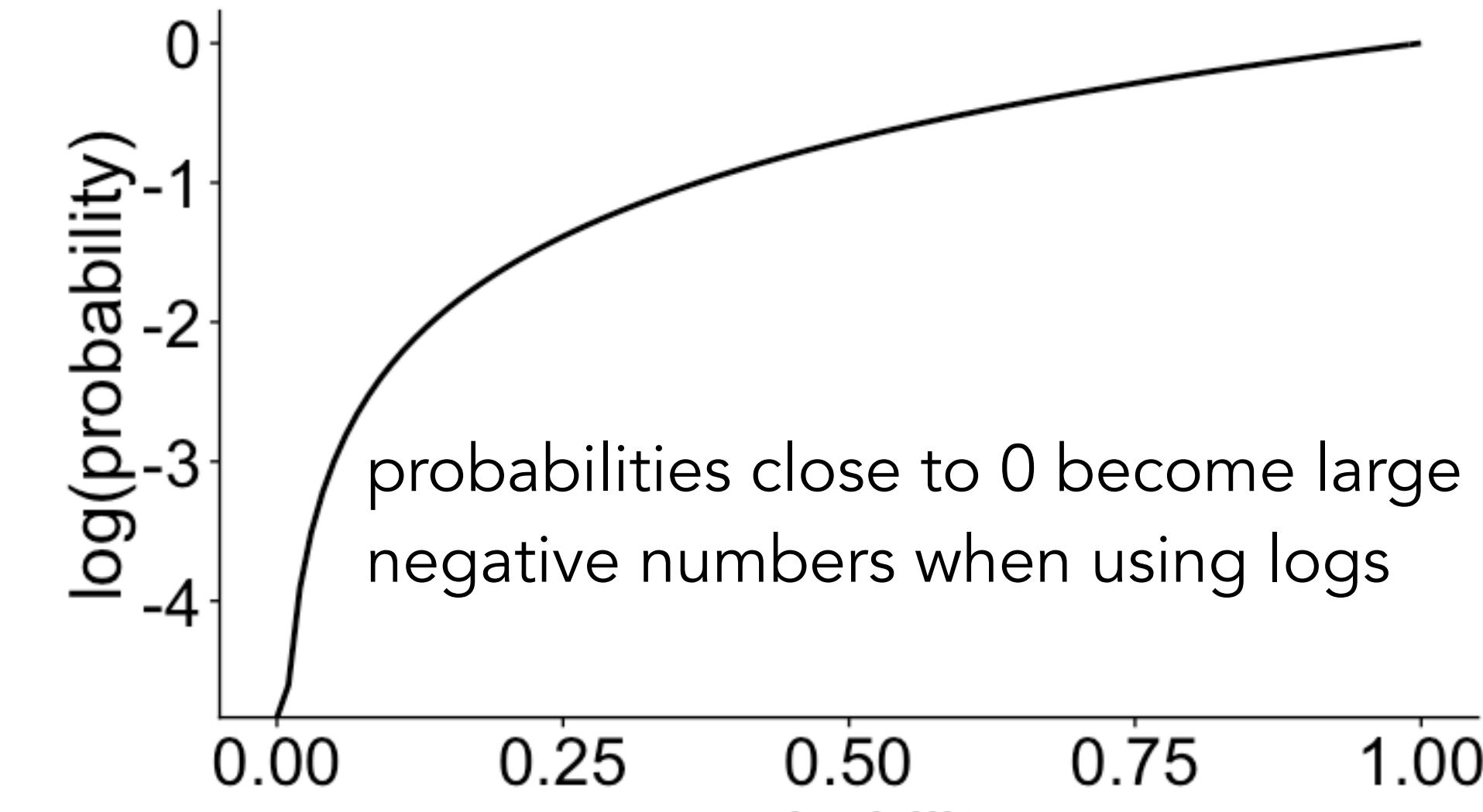
$$\frac{1}{6} \cdot \frac{1}{6} = \frac{1}{36}$$

Who?



What?

`log()` is your friend!



multiplying probabilities

$$0.01 \cdot 0.01 \cdot 0.01 \cdot 0.01 = 0.00000001$$

number becomes extremely small quickly

take `log()`

$$\log(0.01) = -4.60517$$

number becomes large
but that's ok

summing logs

$$(-4.60517) + (-4.60517) + (-4.60517) + (-4.60517) = -18.42068$$

transform back into probability

$$\exp(-18.42068) = 0.00000001$$

often not necessary since
we just use `logLikelihood`

AIC and BIC

```

1 # generate some data
2 df.like = tibble(
3   x = runif(20, min = 0, max = 1),
4   y = 1 + 3 * x + rnorm(20, sd = 2)
5 )
6
7 # fit the model
8 fit = lm(formula = y ~ x,
9           data = df.like)
10
11 # model summary
12 fit %>%
13   glance()

```

`dnorm(1.88, mean = 0, sd = 1.45) = 0.12`



x	y	fitted	resid	likelihood
0.90	5.22	3.34	1.88	0.12
0.27	0.20	1.56	-1.36	0.18
0.37	-0.18	1.86	-2.04	0.10
0.57	2.14	2.42	-0.28	0.27
0.91	3.13	3.37	-0.24	0.27
0.20	0.78	1.38	-0.59	0.25
0.90	4.20	3.34	0.86	0.23
0.94	2.05	3.47	-1.42	0.17
0.66	3.85	2.67	1.18	0.20
0.63	0.41	2.58	-2.17	0.09

inferred standard deviation of the error




r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.25	0.21	1.45	6.16	0.02	2	-34.74	75.47	78.46	37.77	18

$e \sim \mathcal{N}(\text{mean} = 0, \text{sd} = 1.45)$

AIC and BIC

$$AIC = 2k - 2 \ln(\hat{L})$$

$$BIC = \ln(n)k - 2 \ln(\hat{L})$$

$\ln(\hat{L})$ = maximized value of the likelihood function of the model **-34.74**

k = number of parameters in the model **3**

n = number of observations **20**

the sd of the normal distribution modeling the residuals counts as a parameter

```
lm(formula = y ~ 1 + x, data = df.example)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.25	0.21	1.45	6.16	0.02	2	-34.74	75.47	78.46	37.77	18

AIC and BIC

$$AIC = 2k - 2 \ln(\hat{L}) = 2 \cdot 3 - 2 \cdot (-34.74) = 75.47$$

$$BIC = \ln(n)k - 2 \ln(\hat{L}) = \ln(20) \cdot 3 - 2 \cdot (-34.74) = 78.46$$

$\ln(\hat{L})$ = maximized value of the likelihood function of the model **-34.74**

k = number of parameters in the model **3**

n = number of observations **20**

the sd of the normal distribution modeling the residuals counts as a parameter

```
lm(formula = y ~ 1 + x, data = df.example)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.25	0.21	1.45	6.16	0.02	2	-34.74	75.47	78.46	37.77	18

AIC and BIC

$$\text{AIC} = 2k - 2 \ln(\hat{L})$$

$$\text{BIC} = \ln(n)k - 2 \ln(\hat{L})$$

ΔBIC	Evidence against higher BIC
0 to 2	Not worth more than a bare mention
2 to 6	Positive
6 to 10	Strong
>10	Very Strong

- for both AIC and BIC, *lower* is better!
- neither provide a test of a model in the sense of testing a null hypothesis
 - AIC or BIC tell us nothing about the absolute quality of a model, only the quality relative to other models
- BIC generally penalizes free parameters more strongly than AIC (though it depends on the size of n)

What shall I use when?

- Use it all!
- ideally, the different measures provide converging evidence

Table 2

Summary of the model results. Values for r and RMSE indicate means (with 5% and 95% quantiles in parentheses) based on 100 split-half cross-validation runs. BIC scores are based on running the models on the full data set.

Model	r	RMSE	BIC
Difference & pivotality	.86 (.66, .95)	10.56 (6.17, 17.21)	158.59
Difference	.70 (.30, .90)	26.92 (16.4, 40.6)	209.74
Pivotality	.63 (.41, .77)	14.23 (11.39, 17.54)	199.53
Optimality	.66 (.42, .84)	14.55 (10.54, 17.91)	199.47

Note: BIC = Bayesian Information Criterion (lower values indicate better model performance).

Plan for today

- Quick recap
- Model comparison
 - Cross-validation
 - AIC and BIC



0%

much too slow

0%

a little too slow

0%

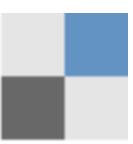
just right

0%

a little too fast

0%

much too fast



Start the presentation to see live content. For screen share software, share the entire screen. Get help at pollev.com/app



Thank you!