Computational Foundations for ML

10-607

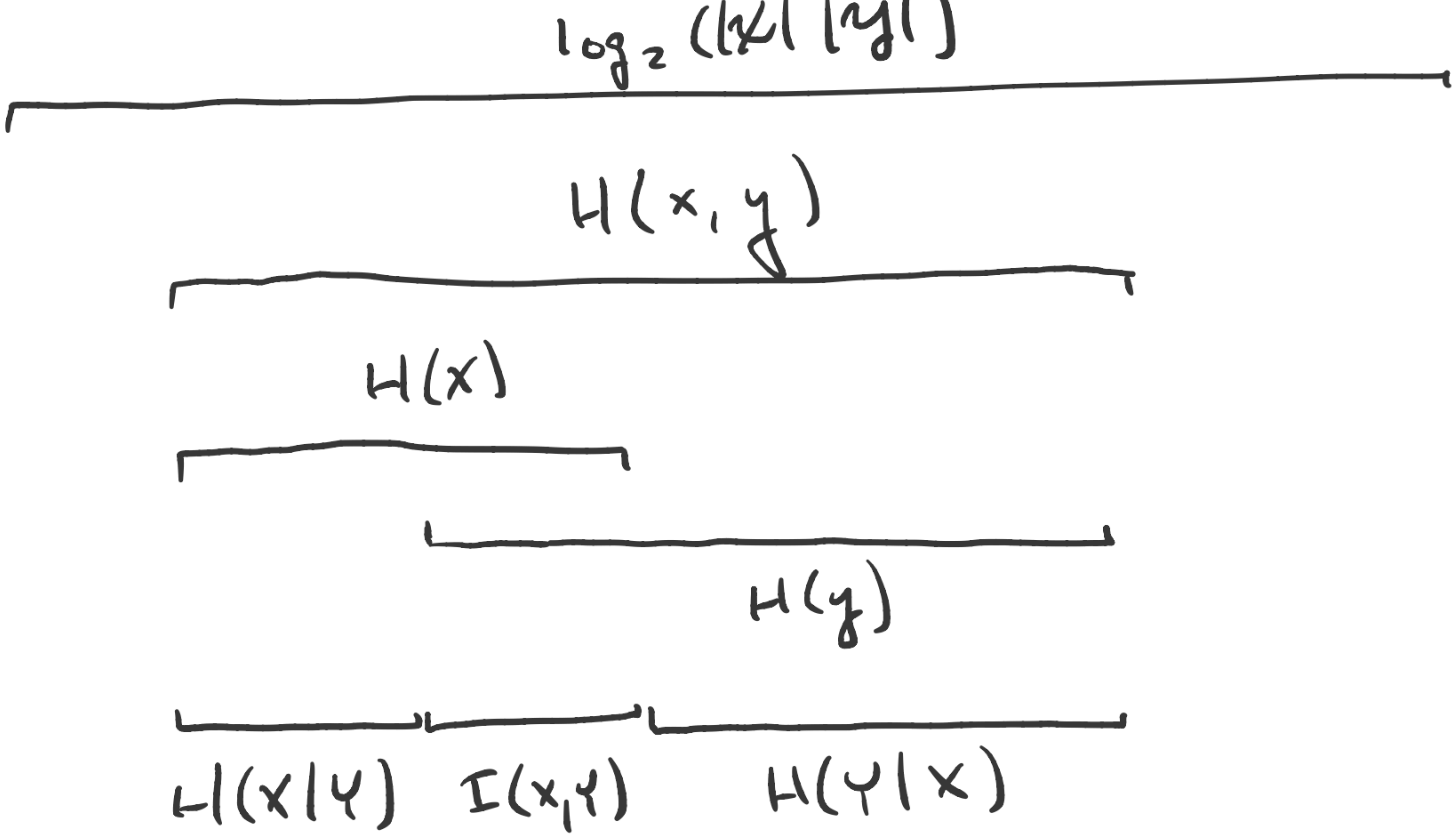
Geoff Gordon

Notes and reminders

- Change in my office hours
- Review period on W
- Lab 4 on F

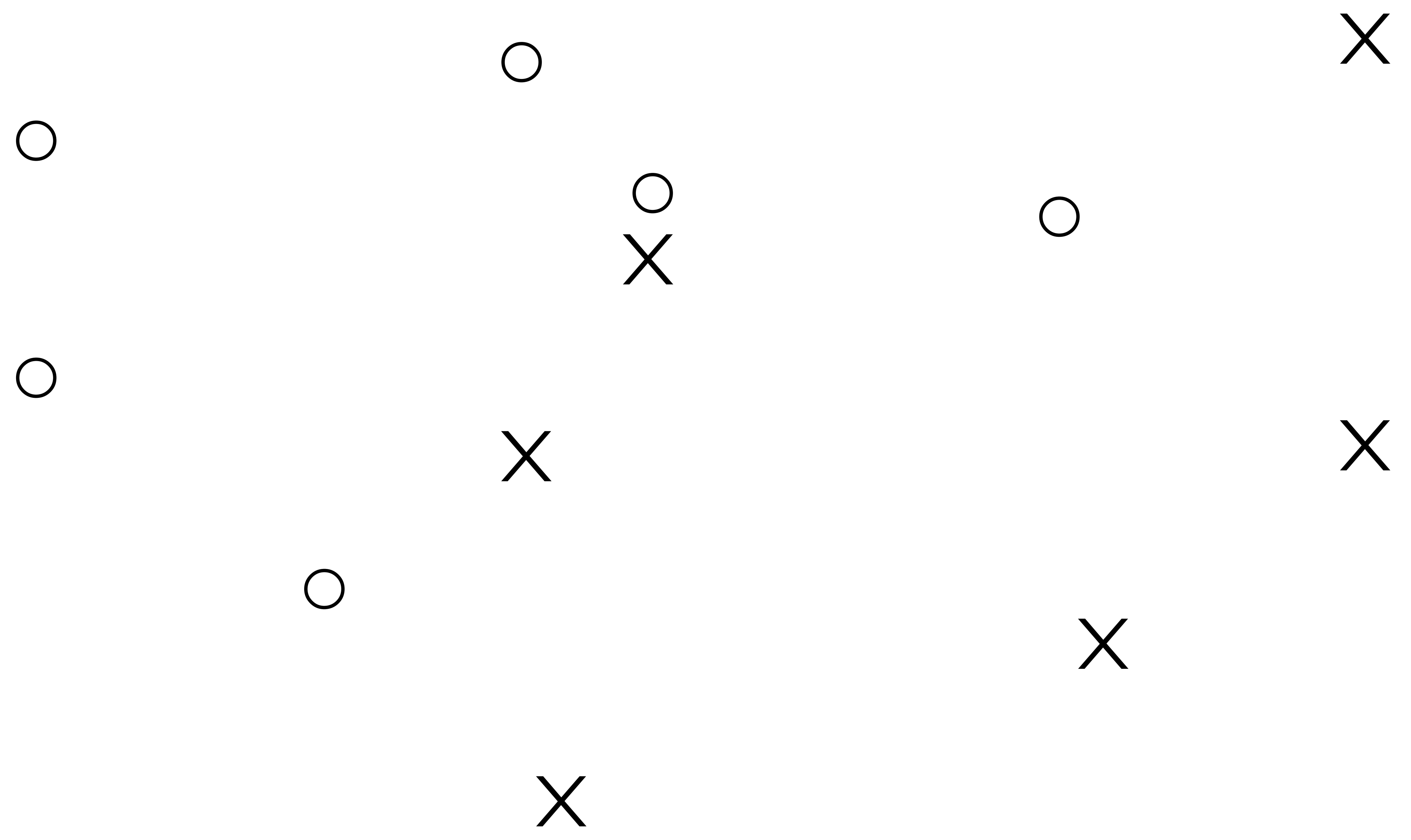
Information inequalities

for joint P(X, Y)



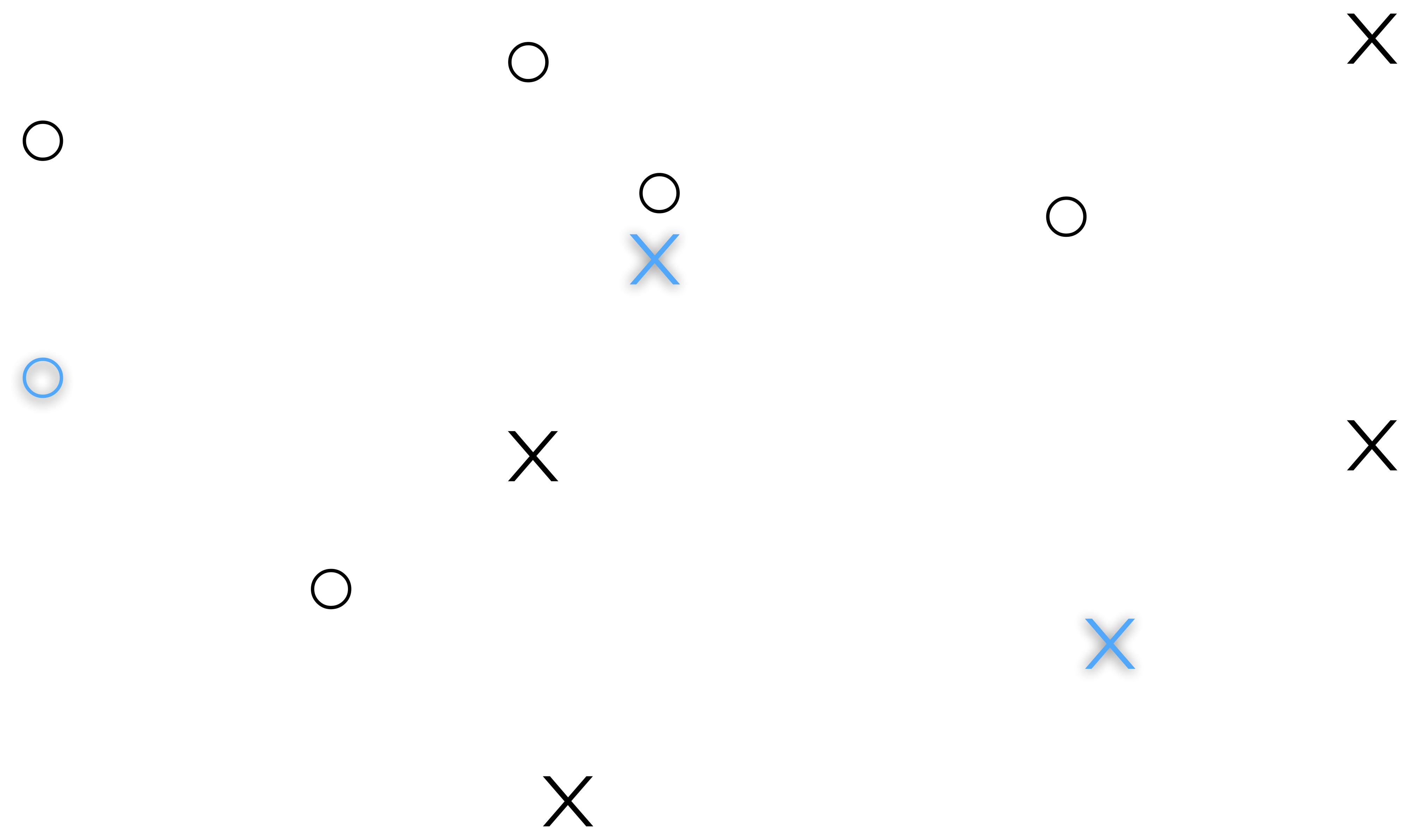
How can we detect overfitting?

- Hold-out set (aka validation set)



How can we detect overfitting?

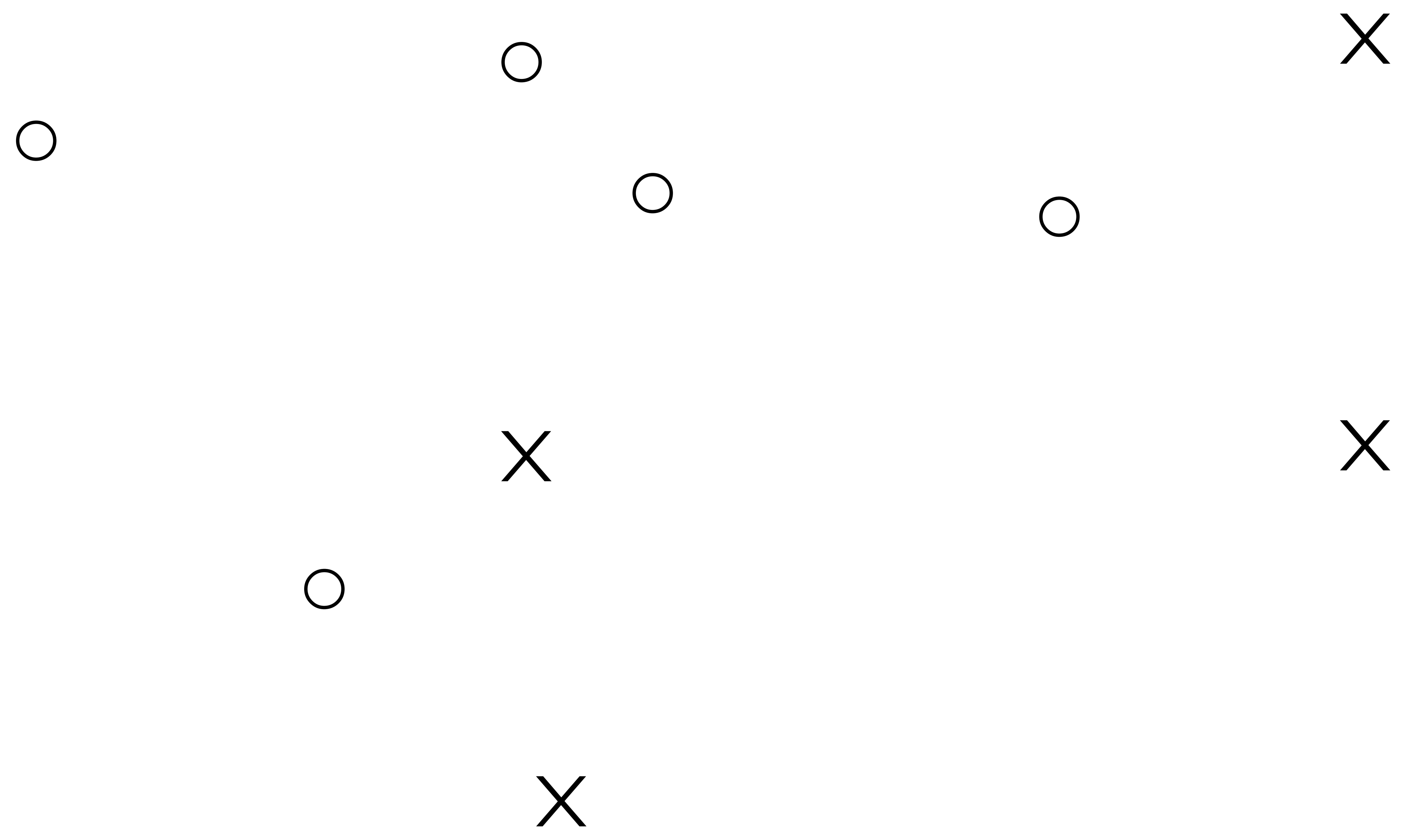
- Hold-out set (aka validation set)



remove hold-out group, fit on rest

How can we detect overfitting?

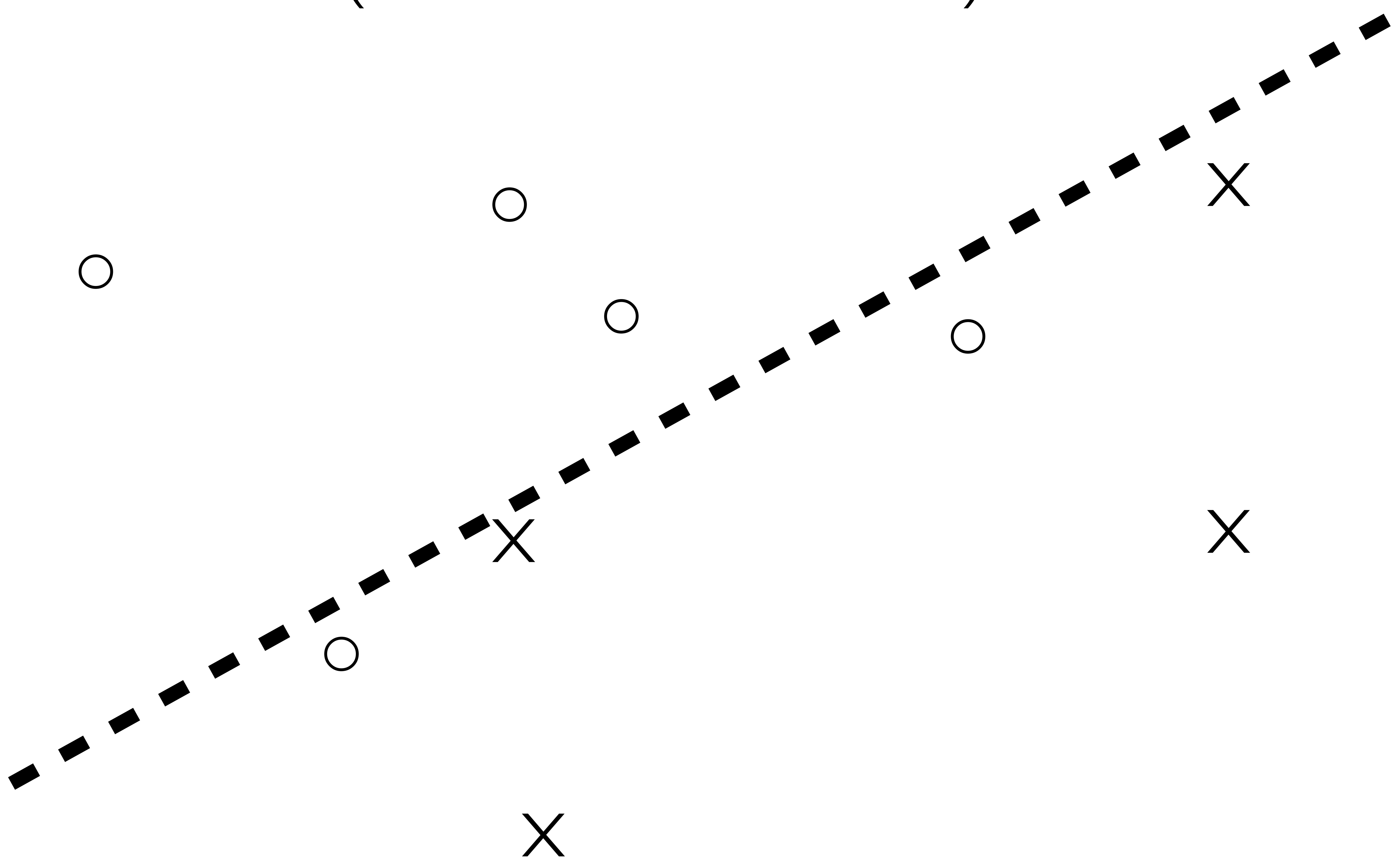
- Hold-out set (aka validation set)



remove hold-out group, fit on rest

How can we detect overfitting?

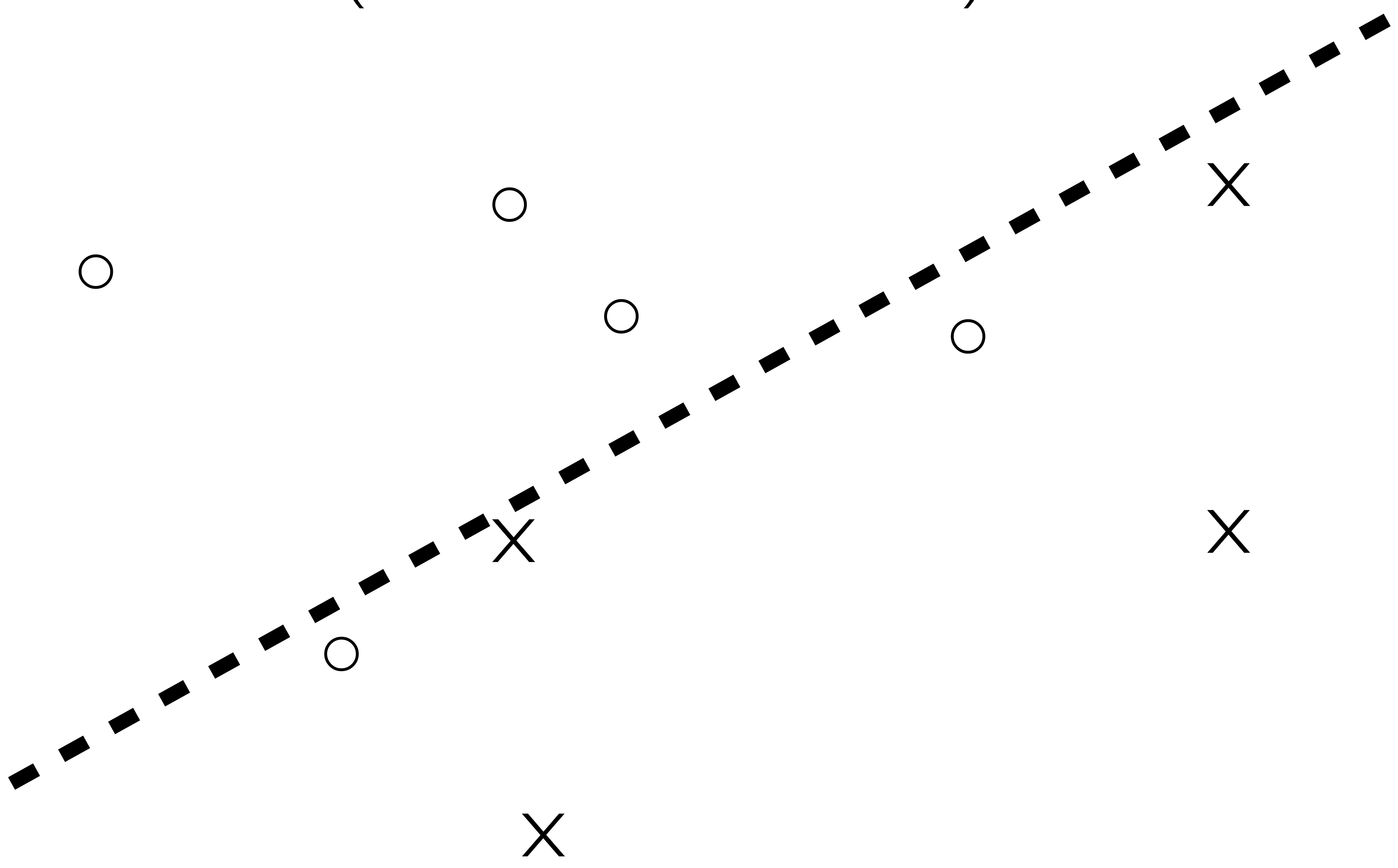
- Hold-out set (aka validation set)



remove hold-out group, fit on rest

How can we detect overfitting?

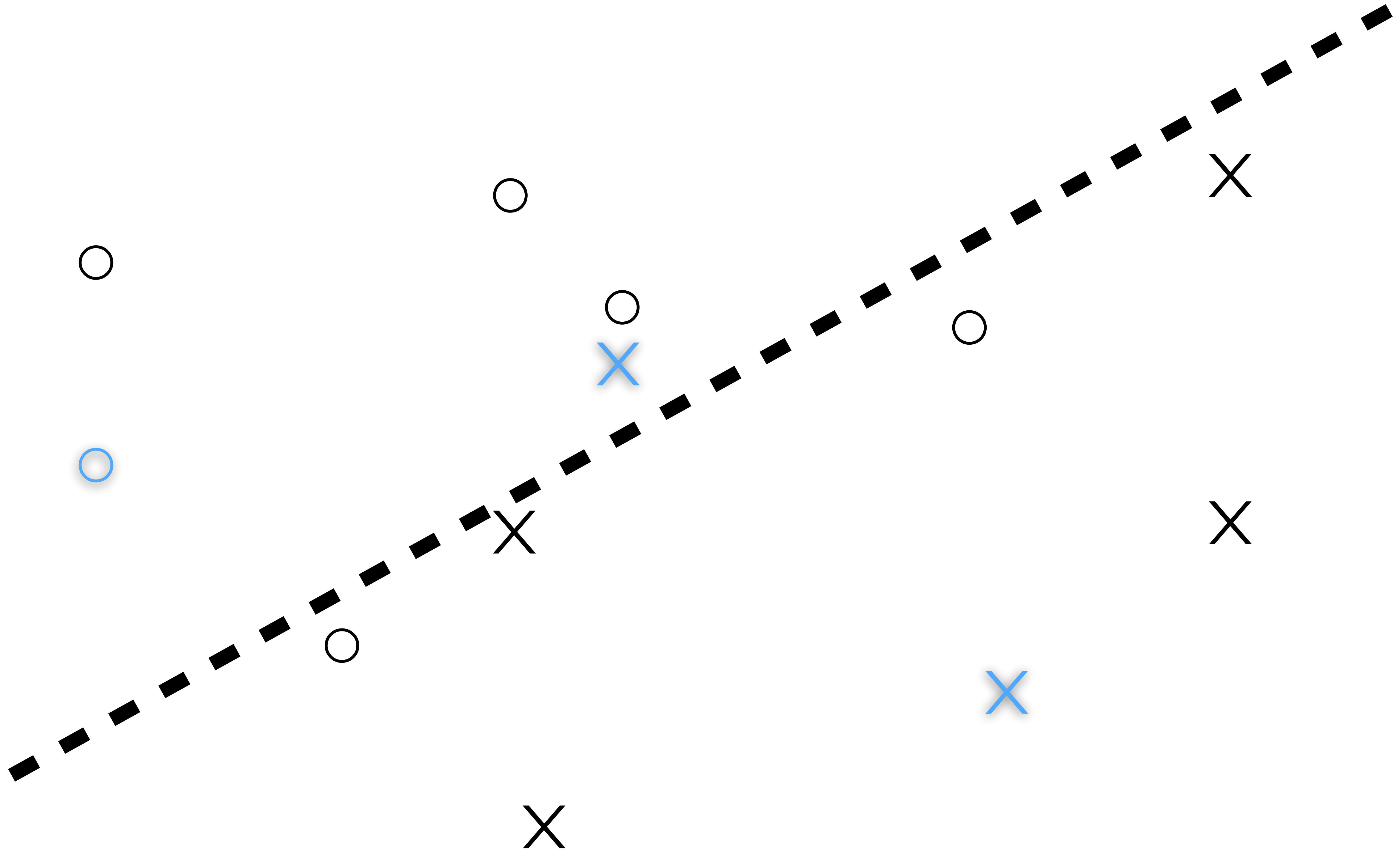
- Hold-out set (aka validation set)



add back in hold-out group, compute error

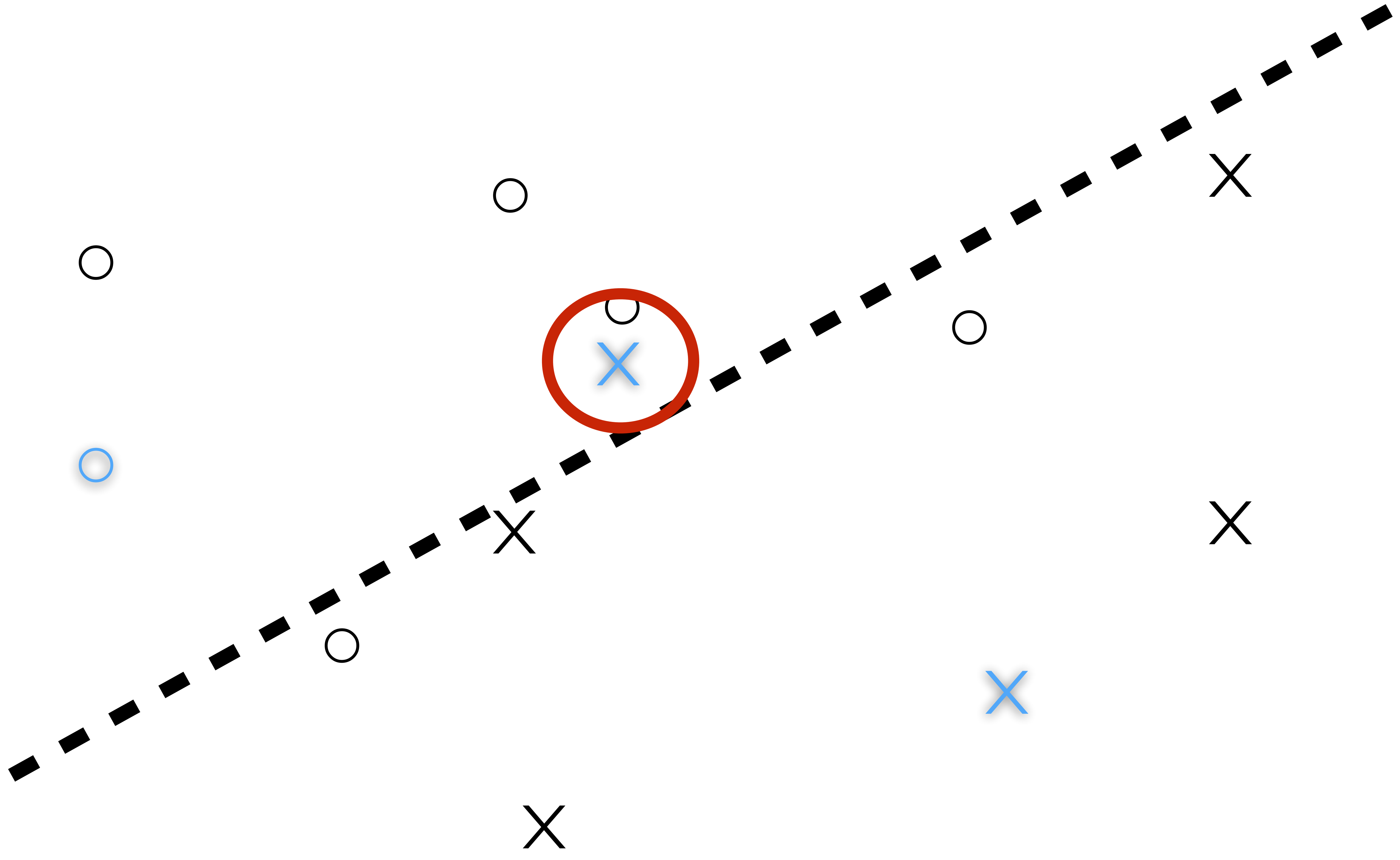
How can we detect overfitting?

- Hold-out set (aka validation set)



How can we detect overfitting?

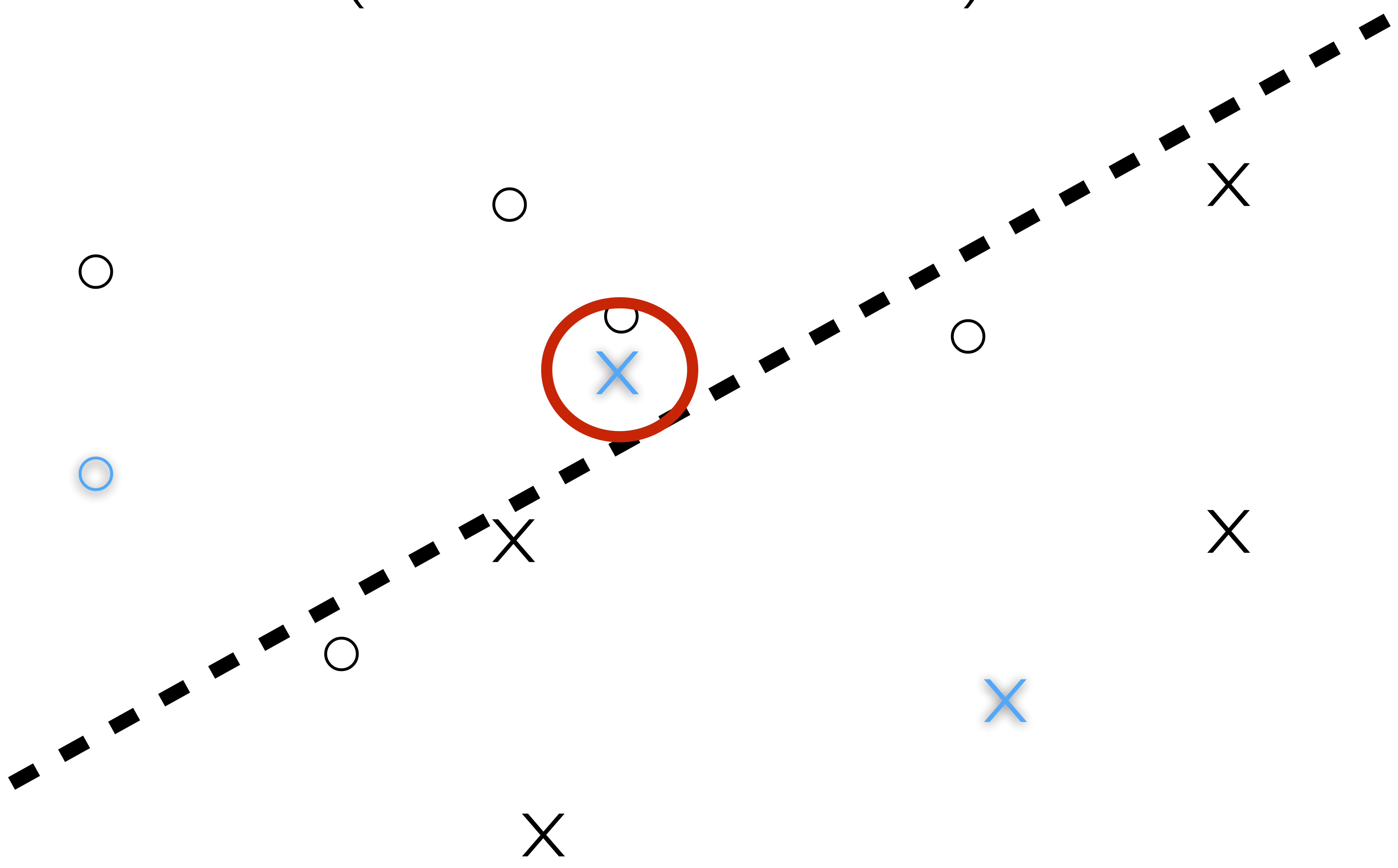
- Hold-out set (aka validation set)



add back in hold-out group, compute error

How can we detect overfitting?

- Hold-out set (aka validation set)



estimated error rate: 1/3

add back in hold-out group, compute error

Hold-out error is unbiased

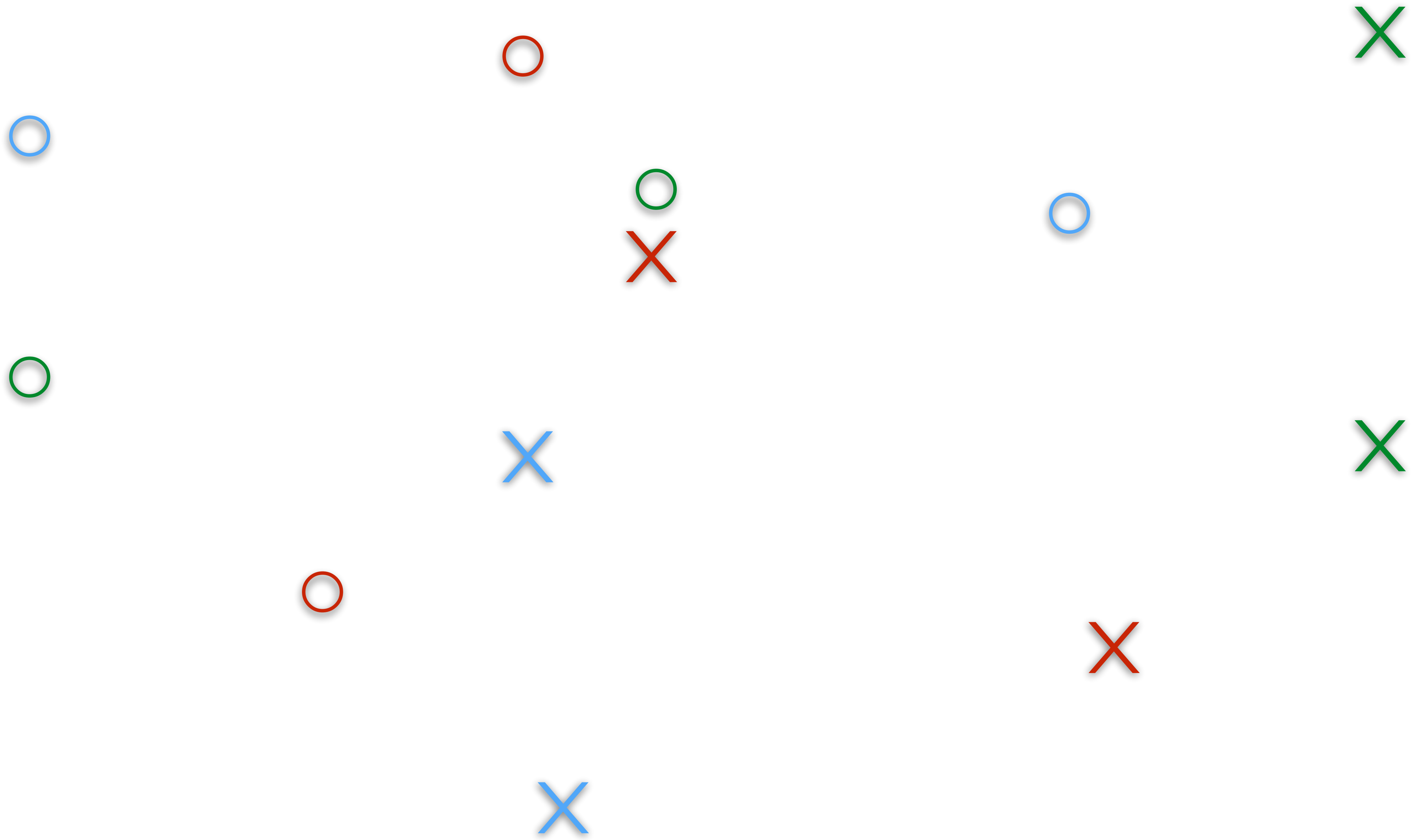
- We didn't optimize on hold-out set, so our error estimate is unbiased (E(holdout error) = true error)
  - so, overfitting detector: holdout error ≫ training set error
- Variance may be high
  - especially if we can only afford a small hold-out set
- We only trained our classifier on some of our data
  - might not reflect amount of overfitting if we used all data

Suppose we detect overfitting

- Hold-out error is much bigger than training error
- What now?
- Tempting to use hold-out set to make some choices (reduce overfitting, improve hold-out performance)
  - which kernel to use? how many iterations of SGD?
  - we'll get to this use case later
  - for now, **warning**: as soon as we optimize anything based on hold-out error, the hold-out error becomes biased!

How can we detect overfitting?

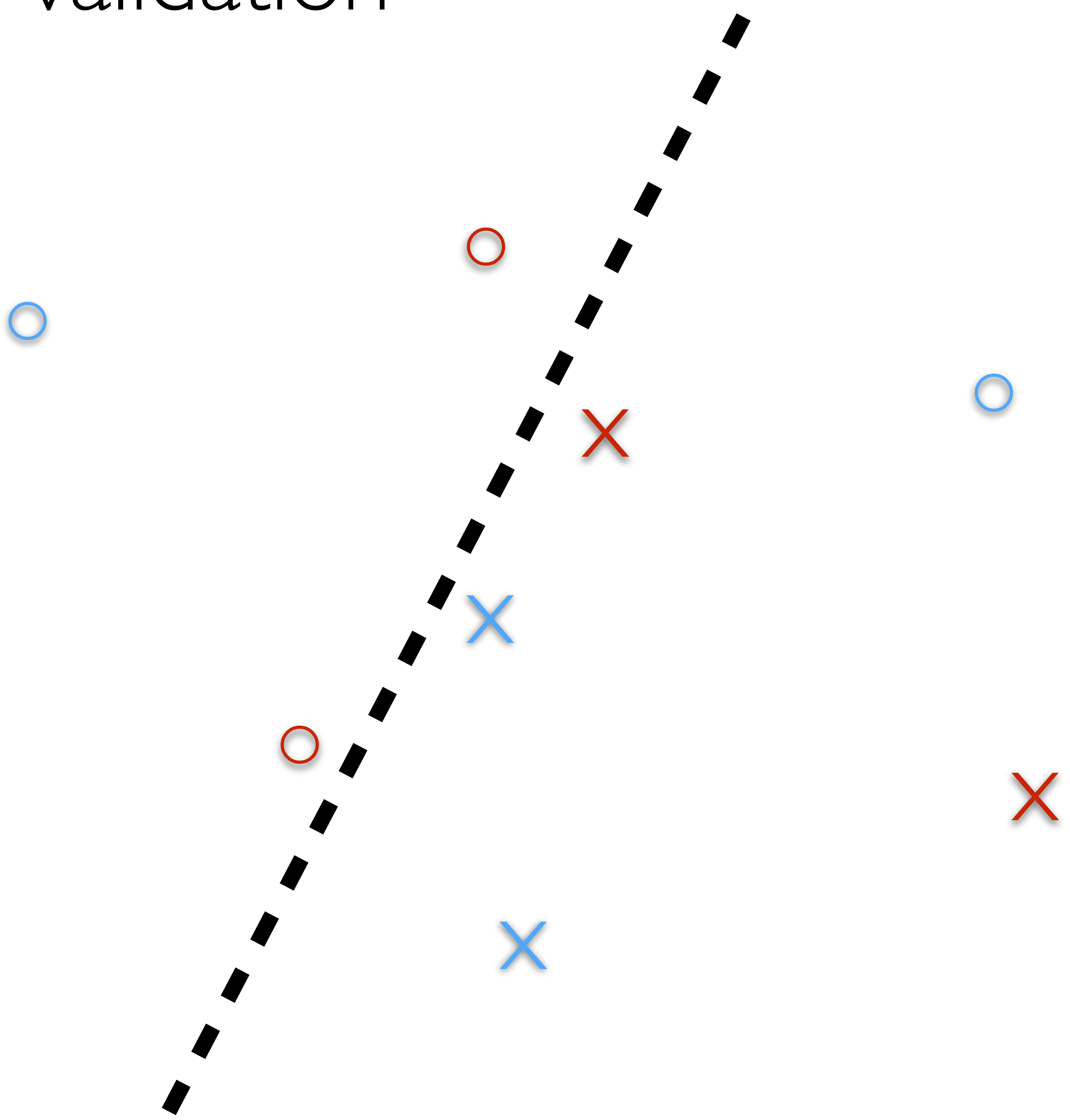
- Cross-validation



split data evenly into groups ("folds")

How can we detect overfitting?

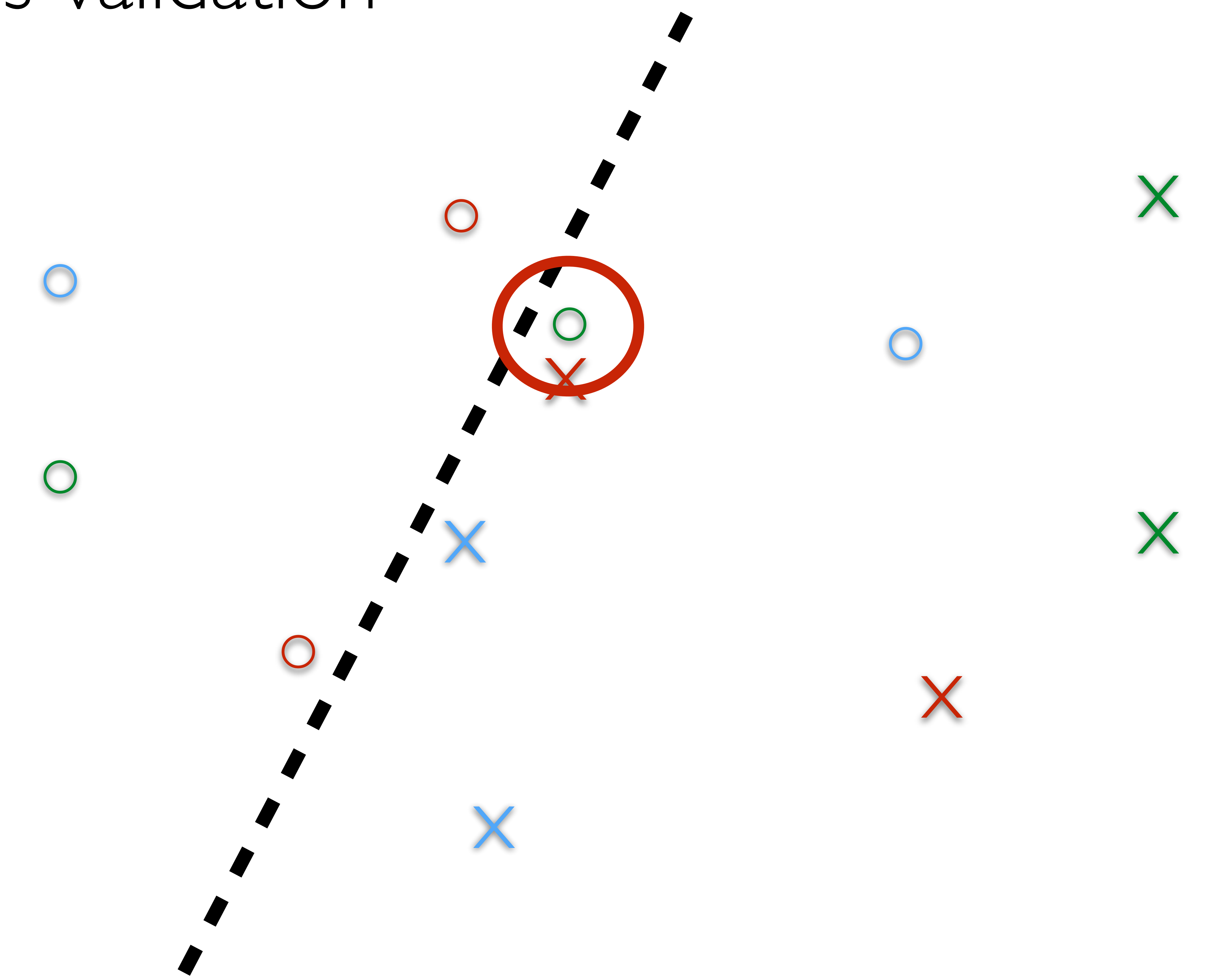
- Cross-validation



remove green group, fit on rest

How can we detect overfitting?

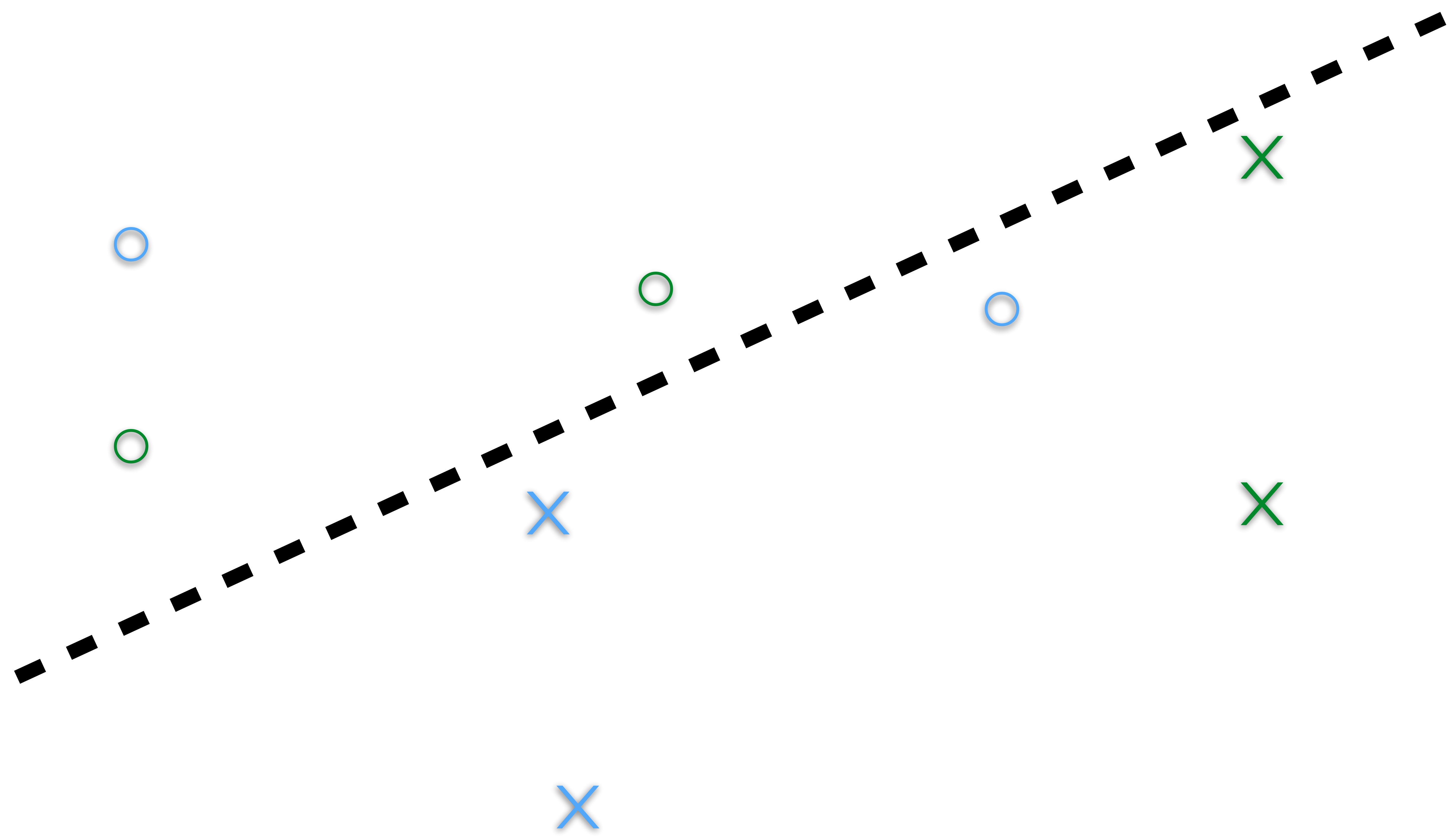
- Cross-validation



add back green group: error 1/4

How can we detect overfitting?

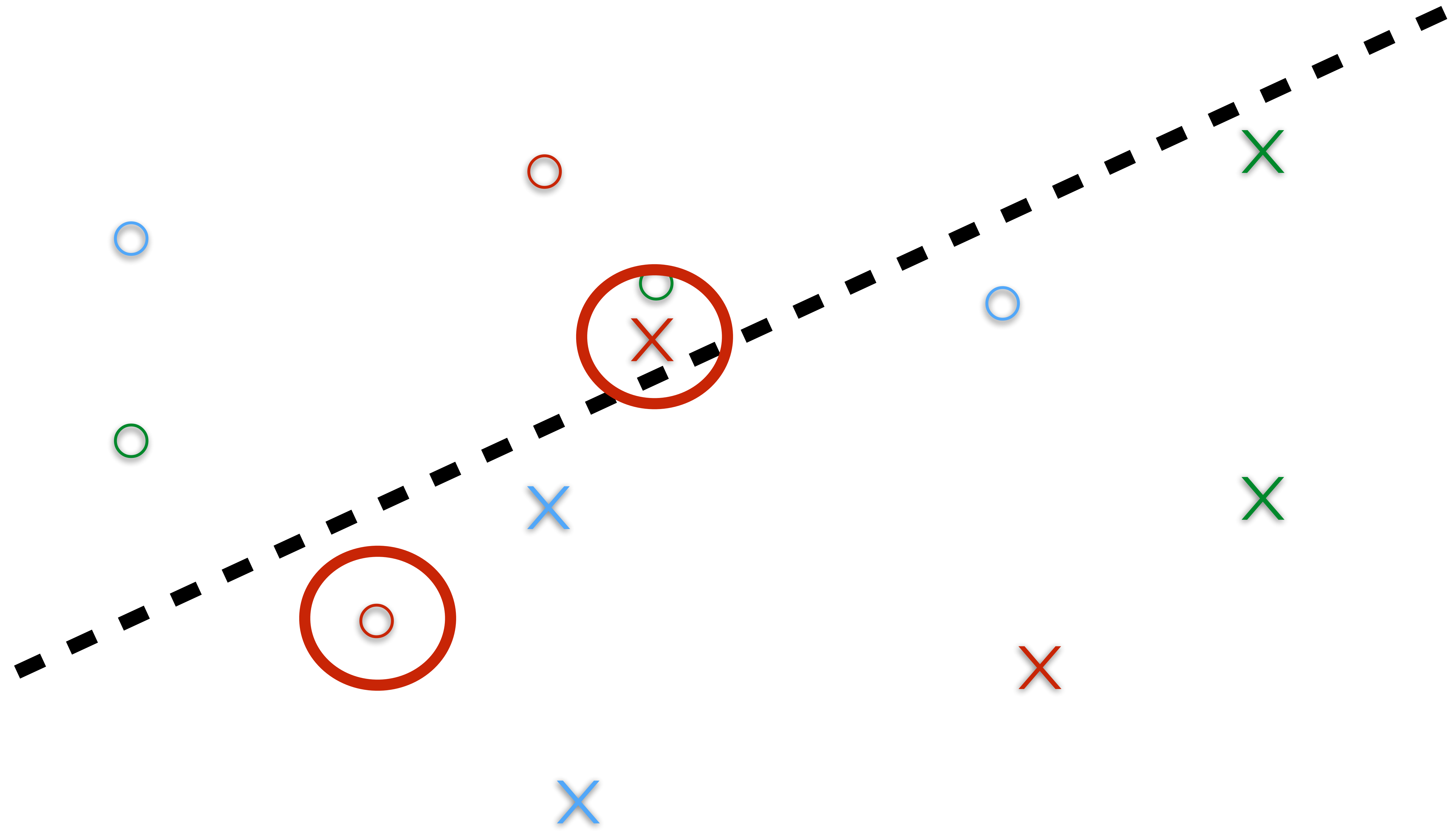
- Cross-validation



remove red group, fit on rest

How can we detect overfitting?

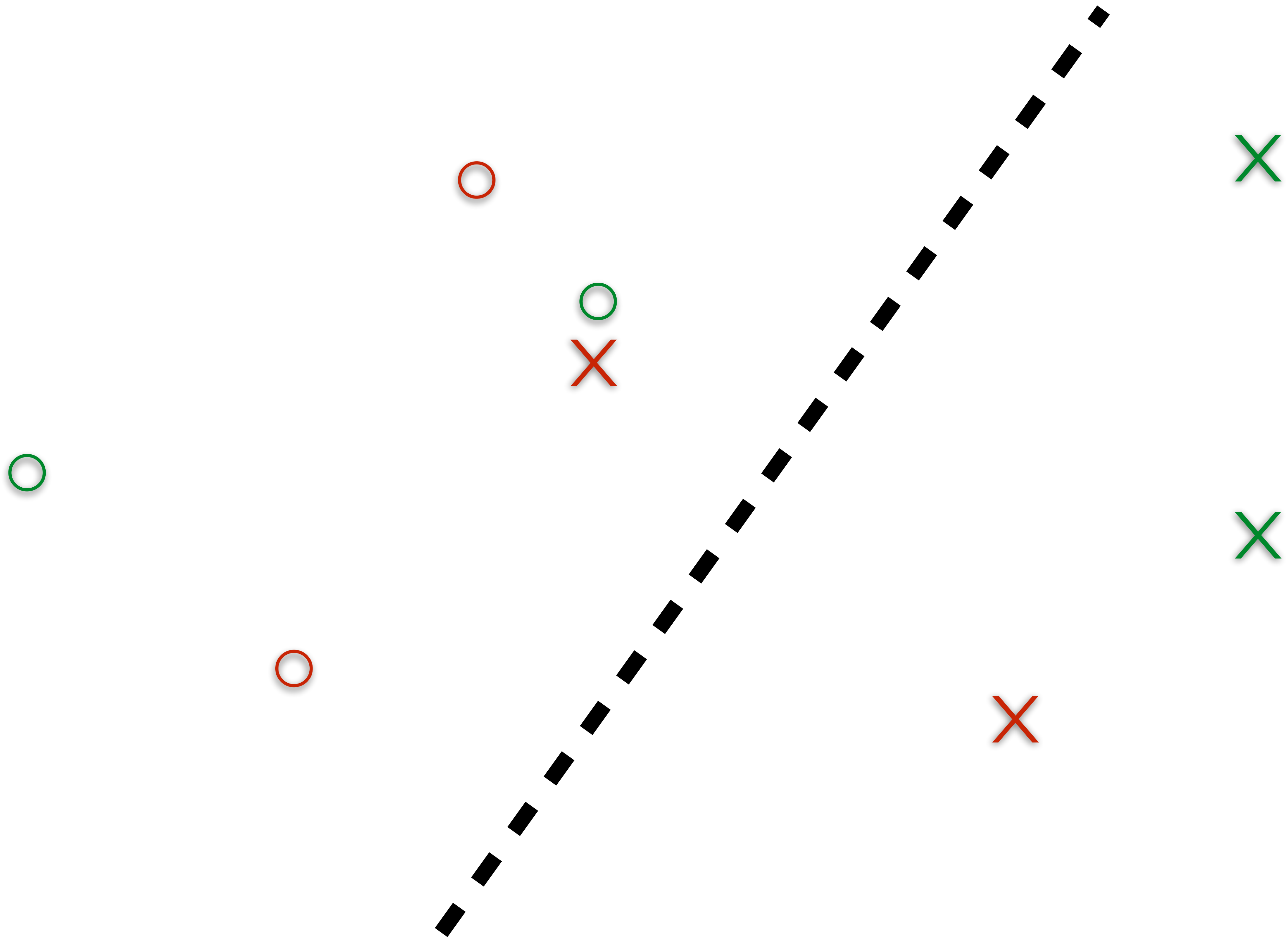
- Cross-validation



add back red group: error 2/4

How can we detect overfitting?

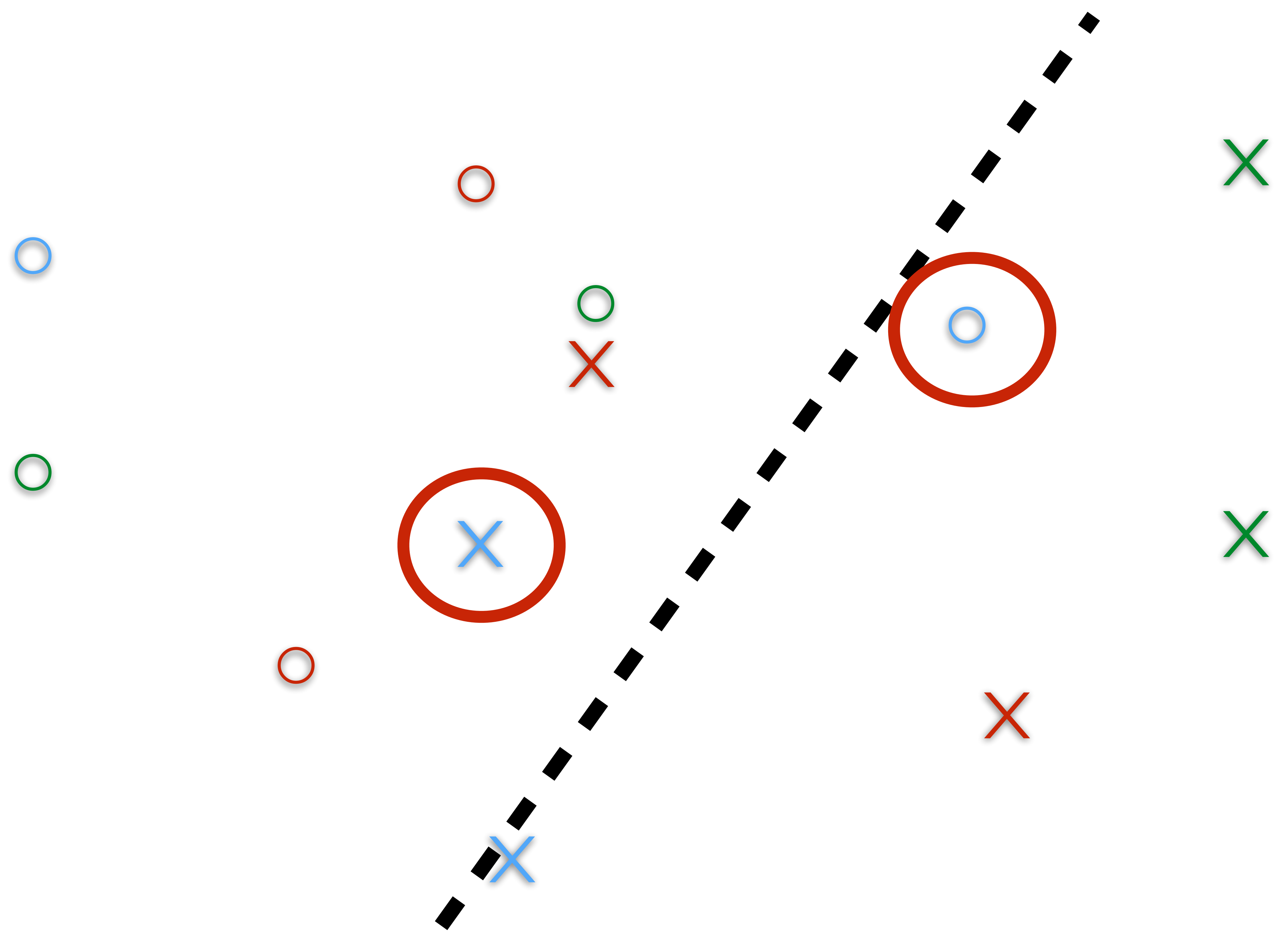
- Cross-validation



remove blue group, fit on rest

How can we detect overfitting?

- Cross-validation

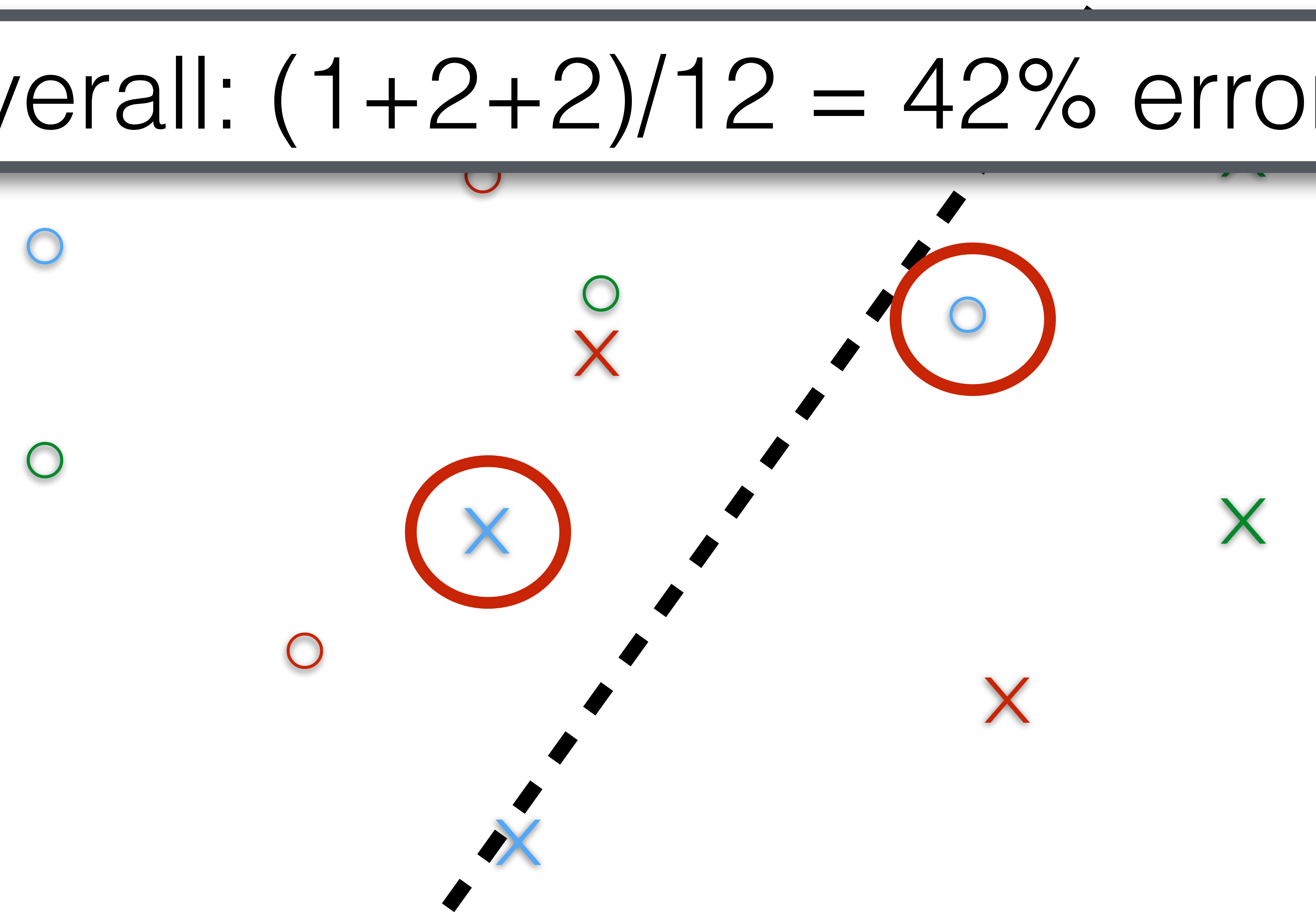


add back blue group: error 2/4

How can we detect overfitting?

- Cross-validation

Overall: (1+2+2)/12 = 42% error rate



add back blue group: error 2/4

Why the name?

- Each fold serves as validation set for other F–1 folds
- Do this in all possible ways = ***cross***-validation

Cross-validation error is unbiased

- In each round, we didn't optimize on hold-out fold, so error estimate is unbiased
  - therefore, so is overall CV error
  - so, overfitting detector: CV error ≫ training set error
- Variance of CV is better than plain hold-out
  - especially if we can only afford a small hold-out set
  - note: folds are not independent!
- We only trained our classifier on some of our data
  - might not reflect amount of overfitting if we used all data

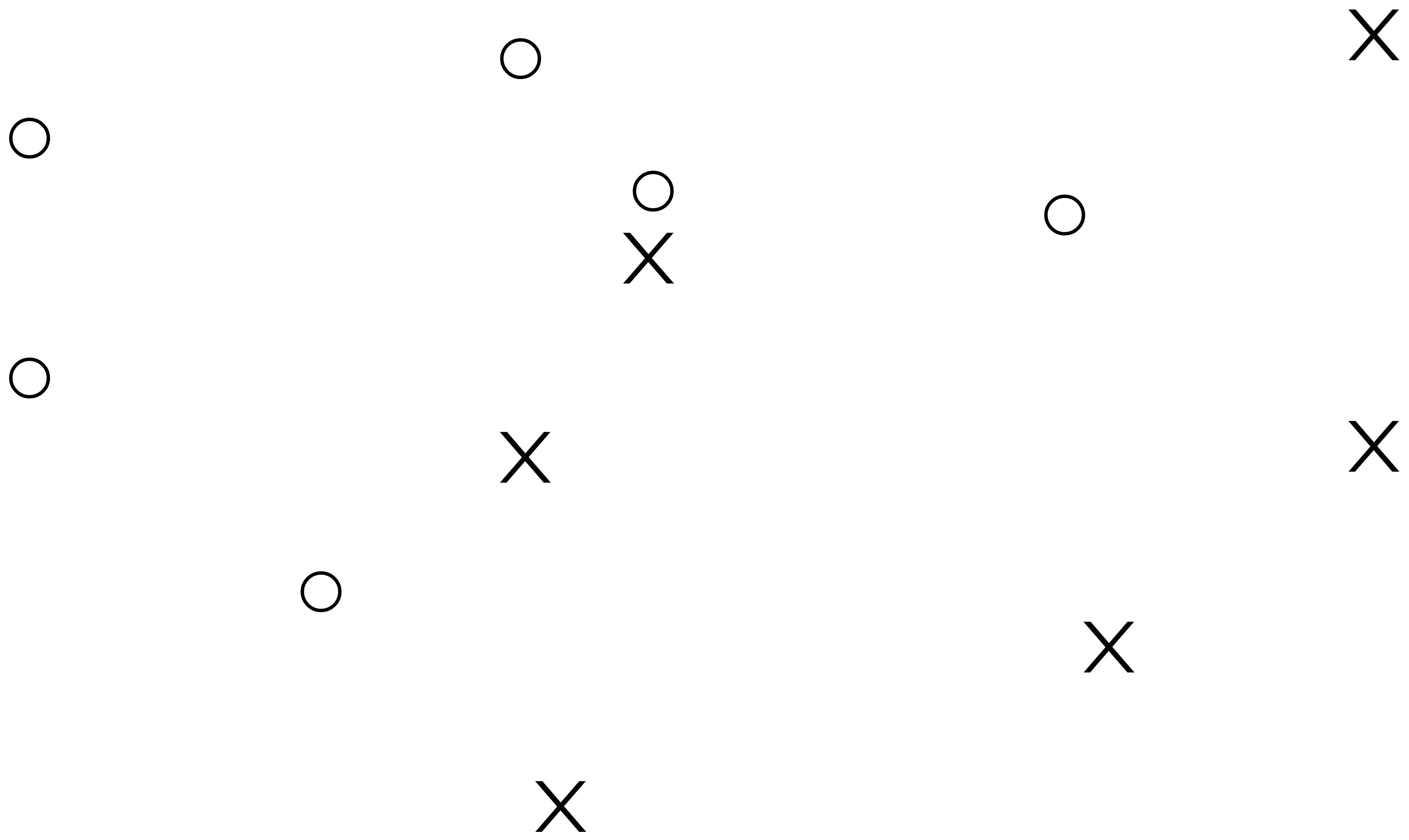
How many folds?

- More folds (F big):
  - train on more data: $(F-1)/F$ — good
  - more computation — bad
    - sometimes, tricks apply: e.g., F=N is cheap in k-nearest-neighbor
- Fewer folds (F small)
  - train on less data — bad
  - less computation = can afford more expensive-to-train models — good

typical: F = 2..10

How can we detect overfitting?

- Bootstrap

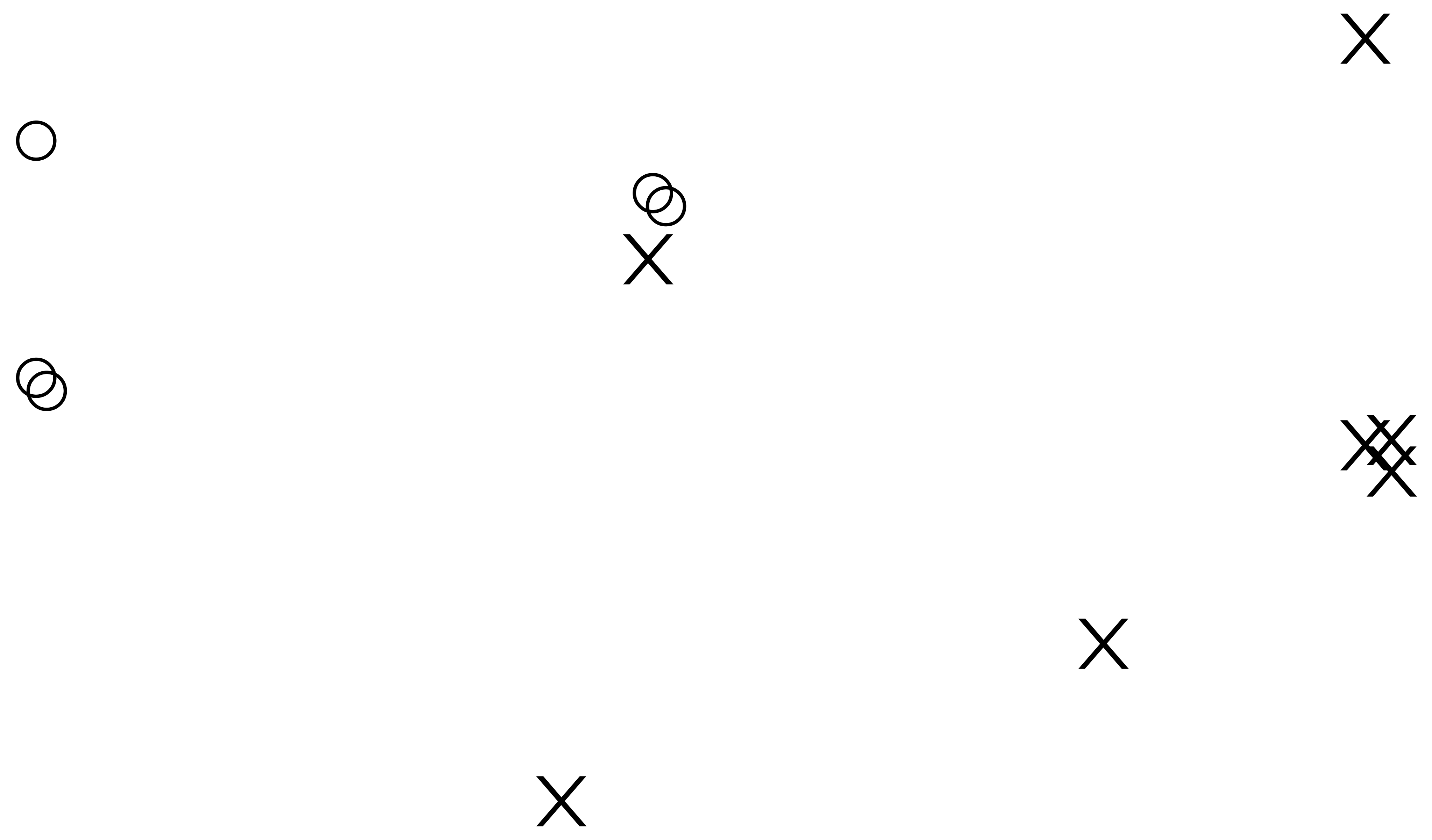


make a ***bootstrap resample*** of our data

How can we detect overfitting?

- Bootstrap

really on top
of each other →



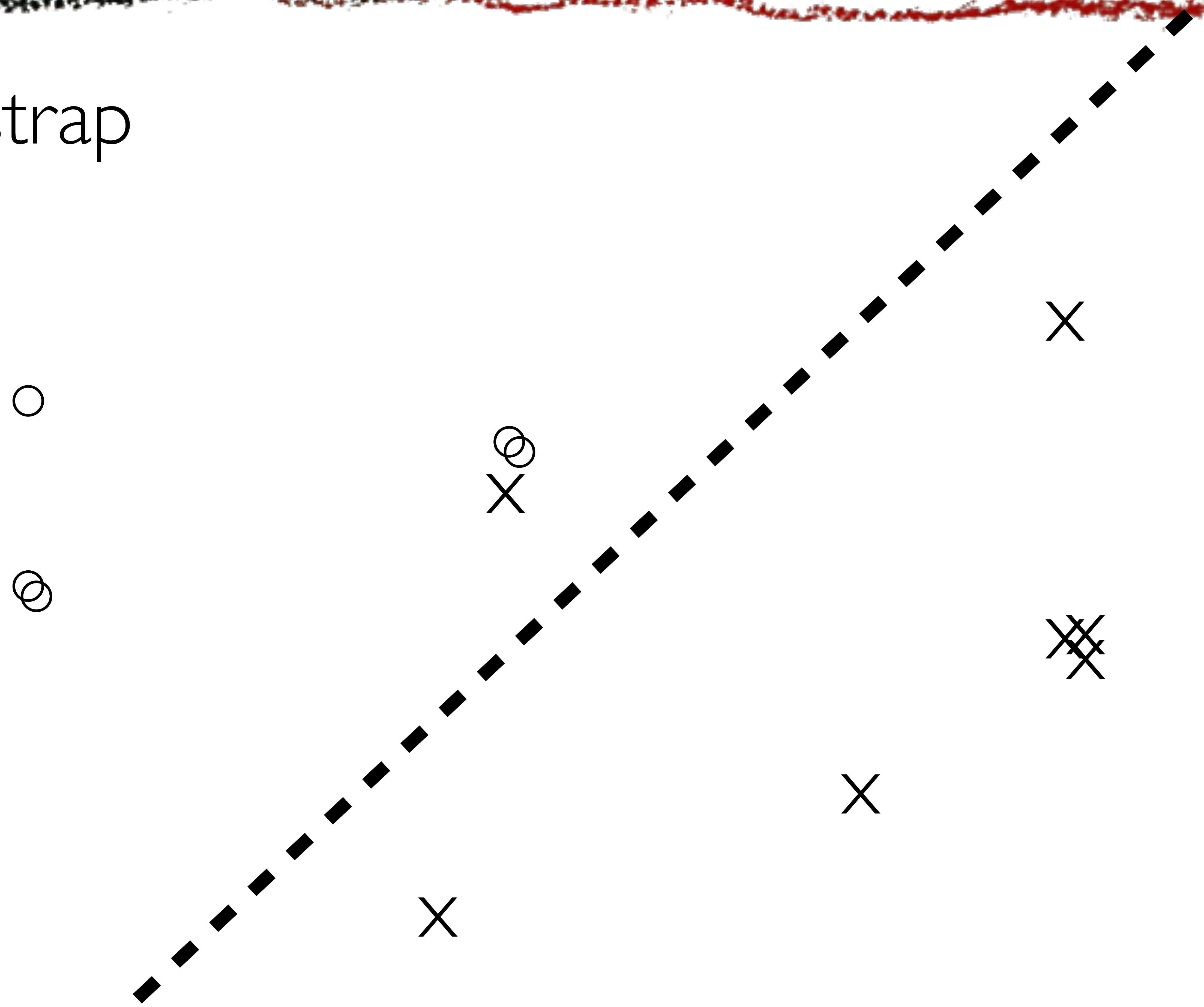
size = N, each example drawn independently w/ replacement from original training set

make a ***bootstrap resample*** of our data

How can we detect overfitting?

- Bootstrap

really on top
of each other →

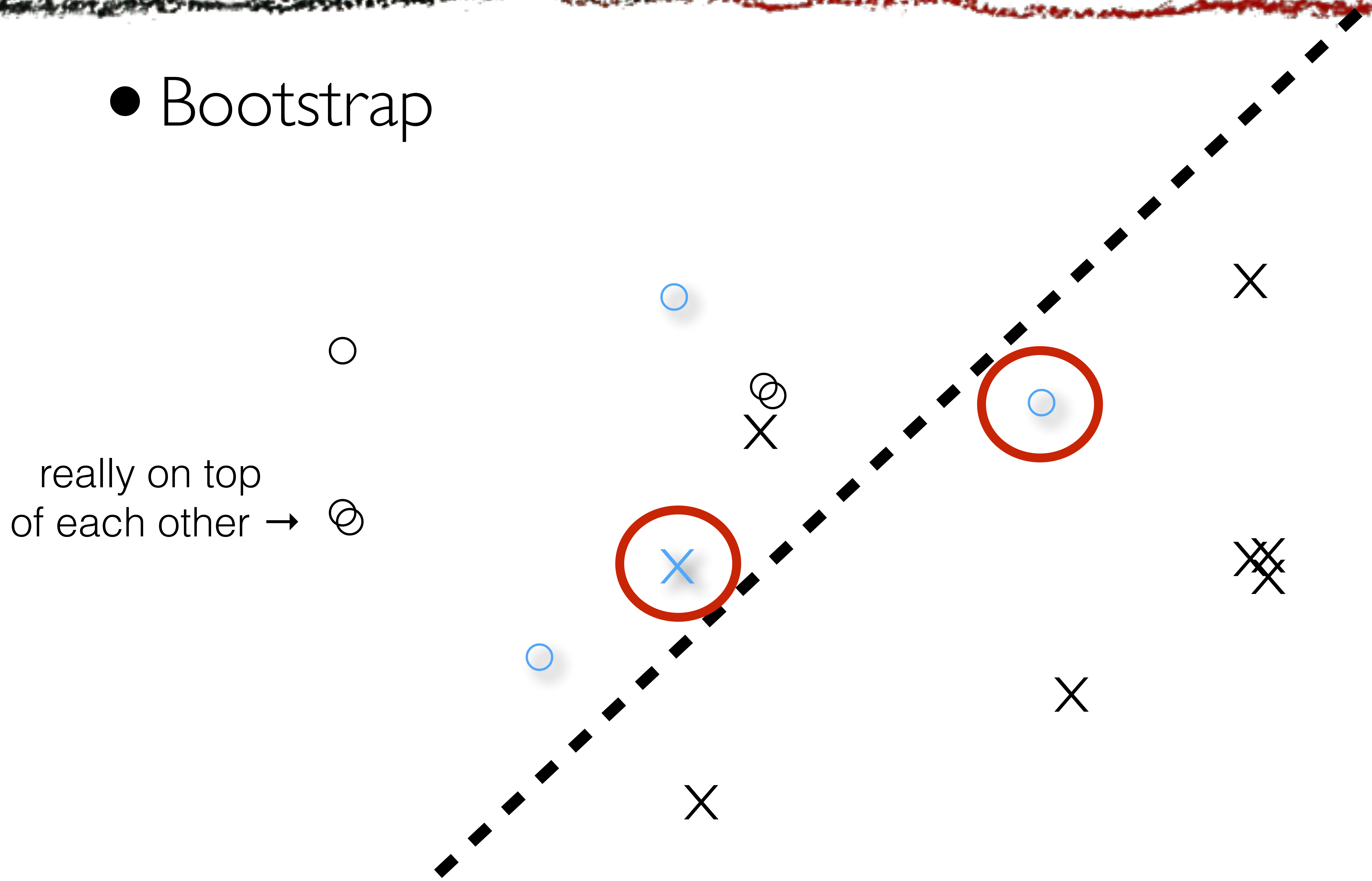


size = N, each example drawn independently w/ replacement from original training set

fit our classifier on the new sample (often called a *bag*)

How can we detect overfitting?

- Bootstrap



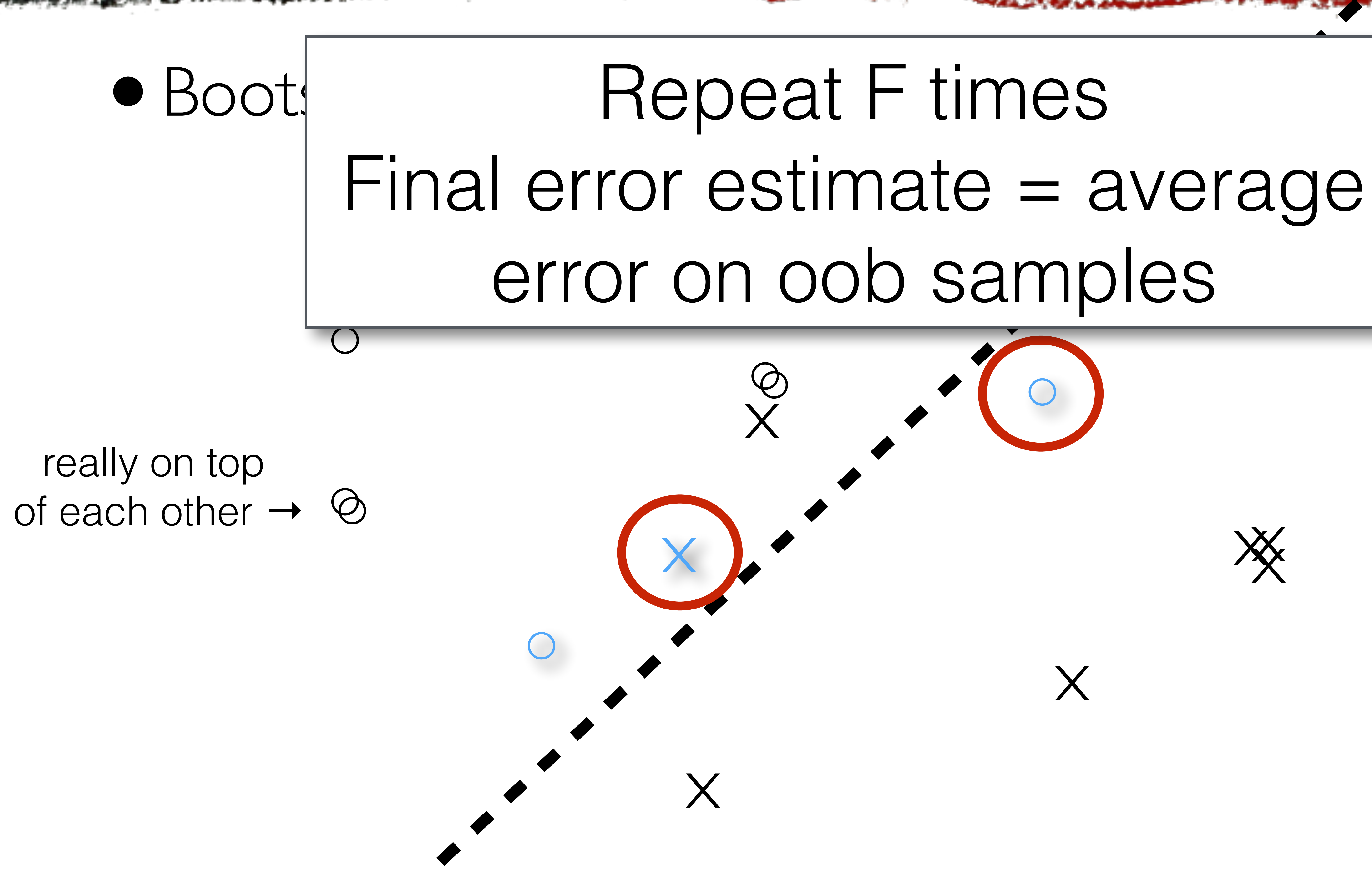
size = N, each example drawn independently w/ replacement from original training set

evaluate on out-of-bag (oob) samples

How can we detect overfitting?

- Boot

Repeat F times
Final error estimate = average error on oob samples



size = N, each example drawn independently w/ replacement from original training set

evaluate on out-of-bag (oob) samples

How can we detect overfitting?

- Boot

Repeat F times
Final error estimate = average error on oob samples

really on top of each other →

Can treat fitted parameter vectors as a sample from ***posterior*** distribution over parameters (given data)

size = N, each example drawn independently w/ replacement from original training set

evaluate on out-of-bag (oob) samples

Why the name?

- Seems like we're getting something for nothing
  - an estimate of error on independent samples, even though we don't have any more independent samples
  - "pulling one's self up by the bootstraps"

Use error estimate to pick model

- Instead of picking model or hyper-parameters (features, kernel, optimizer, etc.) based on training set error, pick them to minimize hold-out, cross-validation, or bootstrap error
- Now put all of our data together (all F folds) and re-optimize the parameters of the model we picked

Model selection by CV

Algorithm	TRAINERR	10-fold-CV-ERR	Choice
1-NN			
10-NN			
Linear Reg'n			
Quad reg'n			⌦
LWR, KW=0.1			
LWR, KW=0.5			

[table credit: Andrew Moore, http://www.autonlab.org/tutorials/]

Fit best model on all the data

Bagging

- For bootstrap or CV, instead of re-fitting best model, make an ensemble
  - vote among the models (one per fold or bag)
  - "bootstrap aggregating" = "bagging"
  - e.g., bagged decision trees → random forests
  - voted prediction approximates Bayesian predictive distribution

What's the catch?

- Two problems with doing model/hyper-parameter selection this way
  - pick too simple a model
  - still don't know its performance

What can go wrong?

- Convergence is only asymptotic (large *original* sample)
  - here: what if original sample hits mostly the larger mode?
- Original sample might not be i.i.d.
  - unmeasured covariate
- We can still overfit the bootstrap / CV / holdout

Save some data for later

- Big data set: say, N=10,000
- Hide some of it
  - say N_v=7,000 visible, N_h=3,000 hidden
  - pretend we never had hidden part — really, no peeking!
- Do stuff that might overfit on our N_v points
  - pick kernel/features, test rules for removing outliers, …
  - use cross-validation within N_v points
- Done? OK, fix **just one** classifier. Test it on the N_h points. Report accuracy.

But I really want to try one more thing

- Often: didn't do as well as expected on the N_h hidden points
  - after all, the whole point was that we risked overfitting

But I really want to try one more thing

- Often: didn't do as well as expected on the N_h hidden points
  - after all, the whole point was that we risked overfitting
- So let's go back and try another idea — fit it on the N_v points.
  - OK, that didn't work — try something else
  - No, not that either — on to the next idea
  - Now it works better on the N_h points. Good, right?

But I really want to try one more thing

- Often: didn't do as well as expected on the N_h hidden points
  - after all, the whole point was that we risked overfitting
- So let's go back and try another idea — fit it on the N_v points.
  - OK, that didn't work — try something else
  - No, not that either — on to the next idea
  - Now it works better on the N_h points. Good, right?
- Strong risk that it doesn't actually work better…

Recursive hiding

- So, split our data into N_v visible points, N_h hidden ones, and N_{rh} "really hidden" ones
  - develop on the N_v
  - test rarely on the N_h
  - test only once at the end on the N_{rh}
- Practically, 3 groups are probably the limit
  - and only if we have lots of data

petal length	petal width	
6.8 cm	3.0 cm	I. virginica
5.7 cm	2.5 cm	I. versicolor
⋮	⋮	

μ_{ver} μ_{virg} $\rightarrow$ $w \cdot x + b \overset{?}{\geq} 0$

↑

$\mu_{ver} - \mu_{virg}$

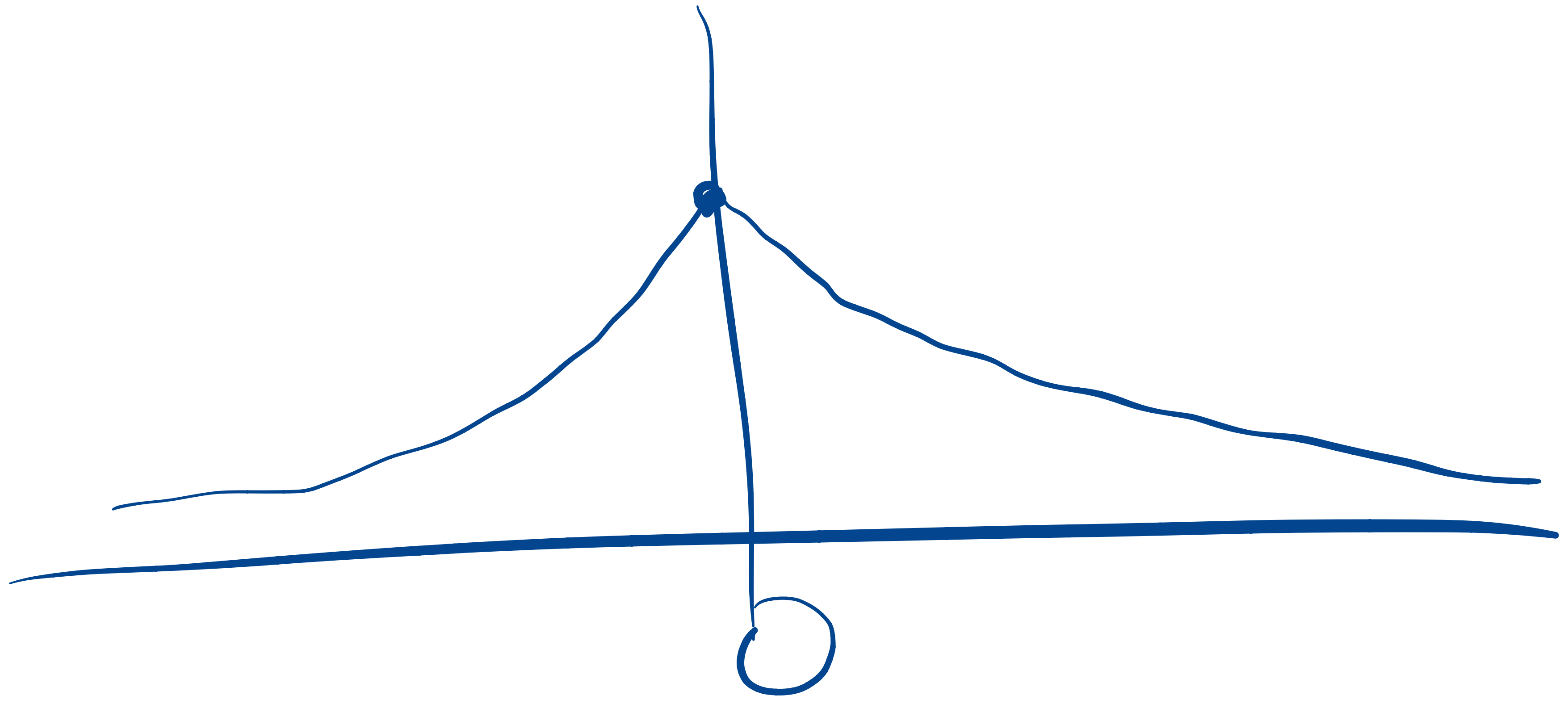
$S \subseteq H$

$$\exp(-\epsilon) \le \frac{P(A(D, r) \in S)}{P(A(D', r) \in S)} \le \exp(+\epsilon)$$

whenever D' near D

change 1 example in D
to get D'

$$P(n) \propto \exp(-|n|/c)$$



$c = 20$