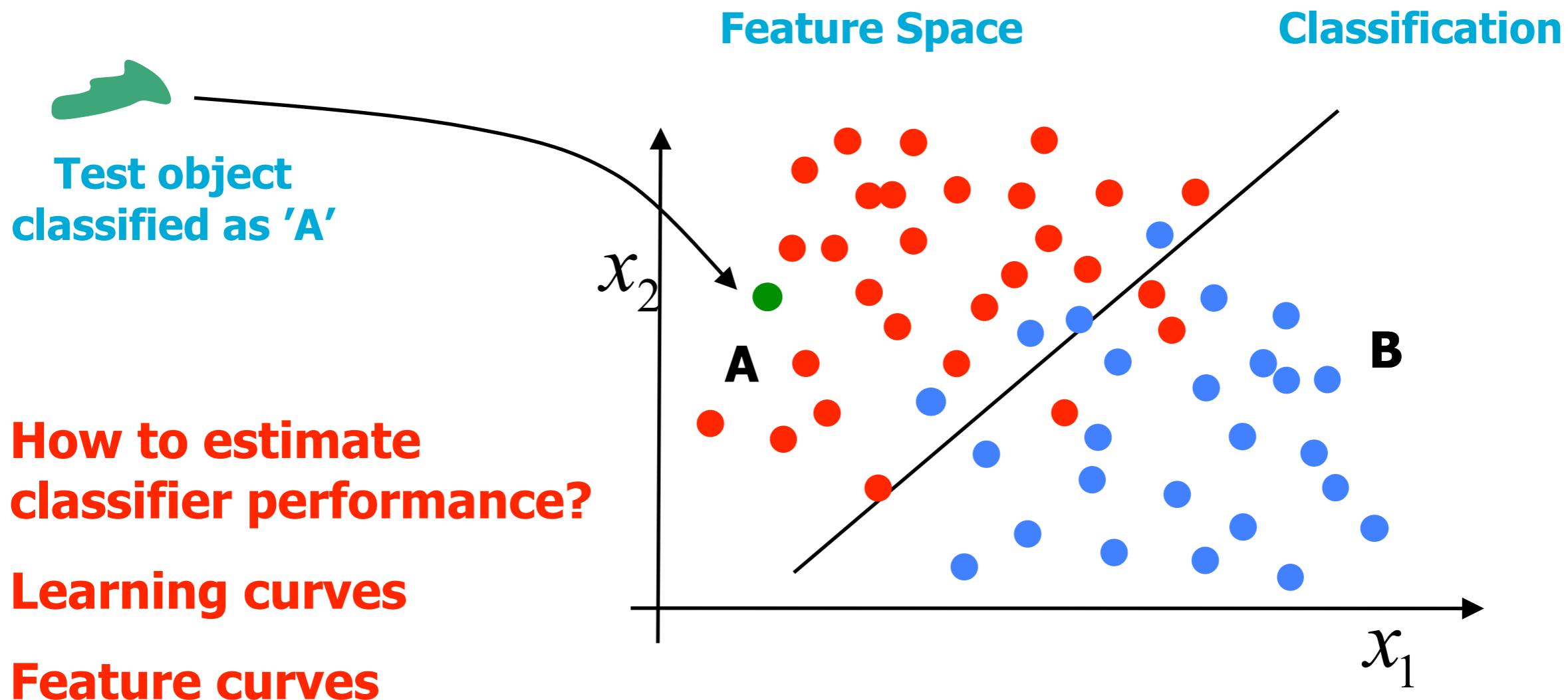
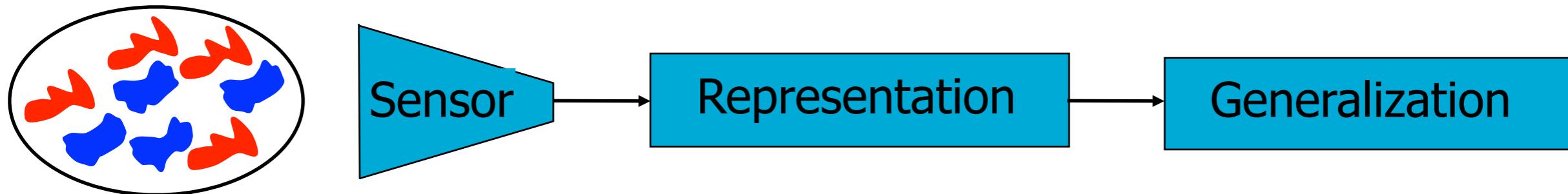


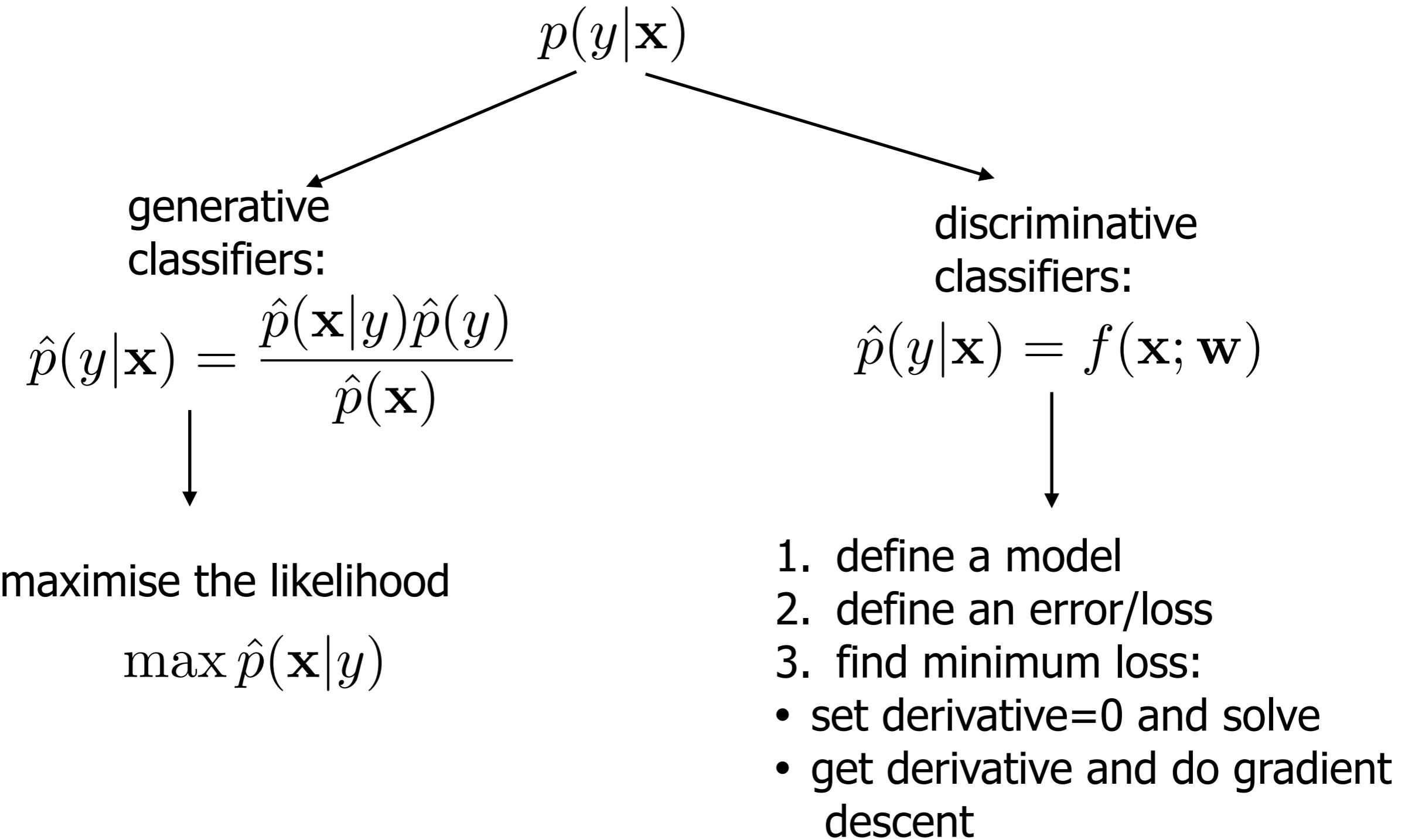
Classifier evaluation



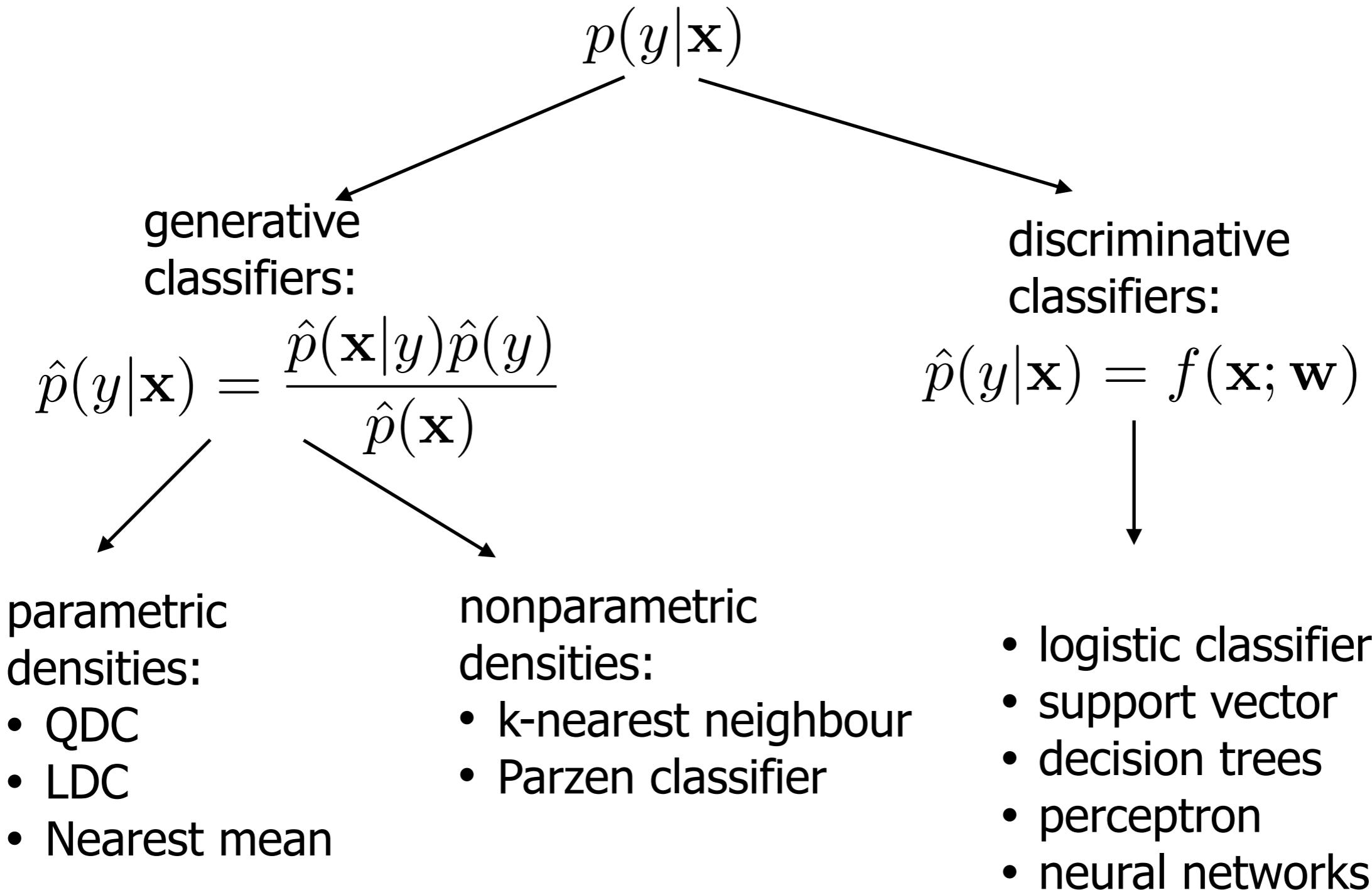
Recap classifiers

$$p(y|\mathbf{x})$$

Recap classifiers

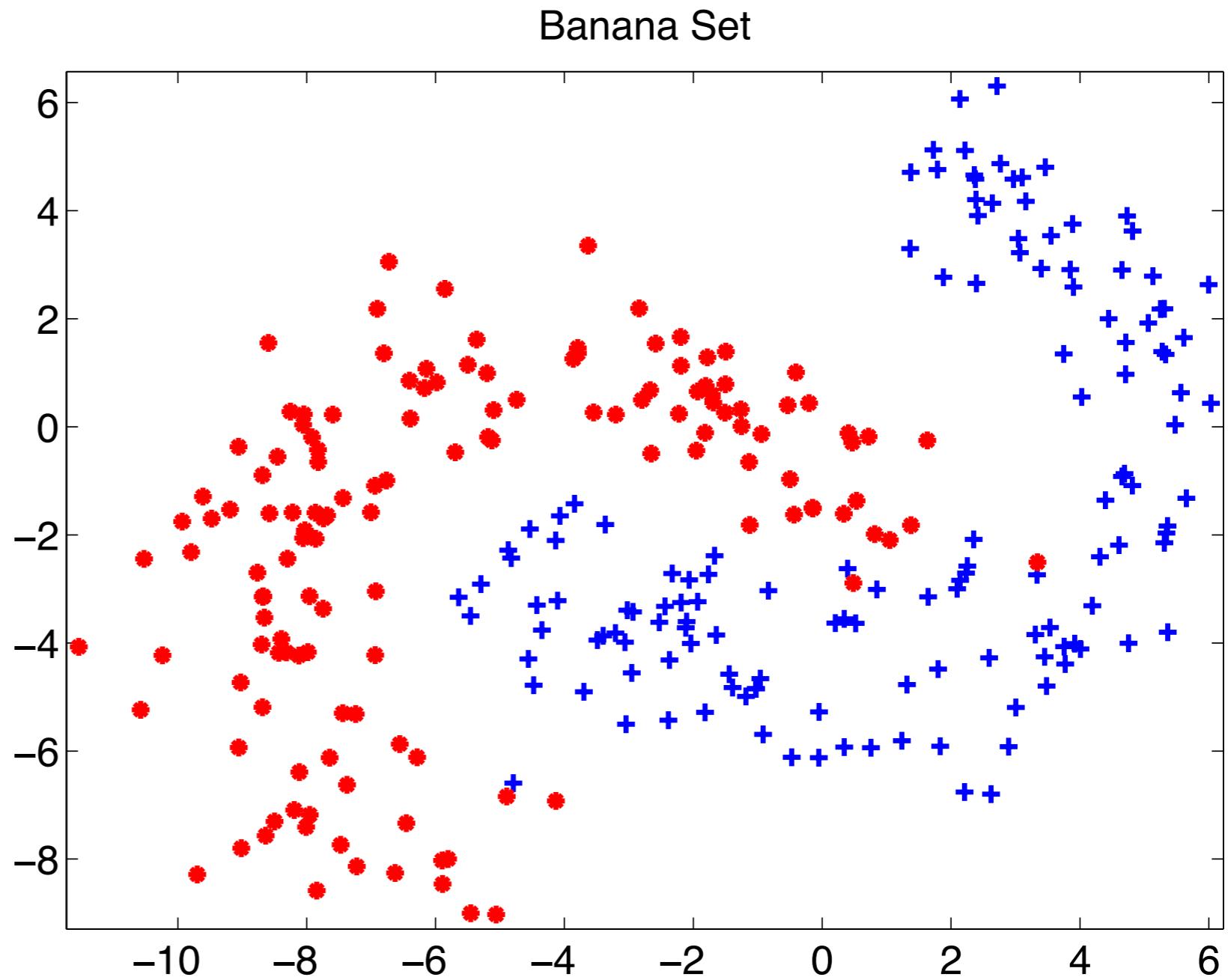


Recap classifiers



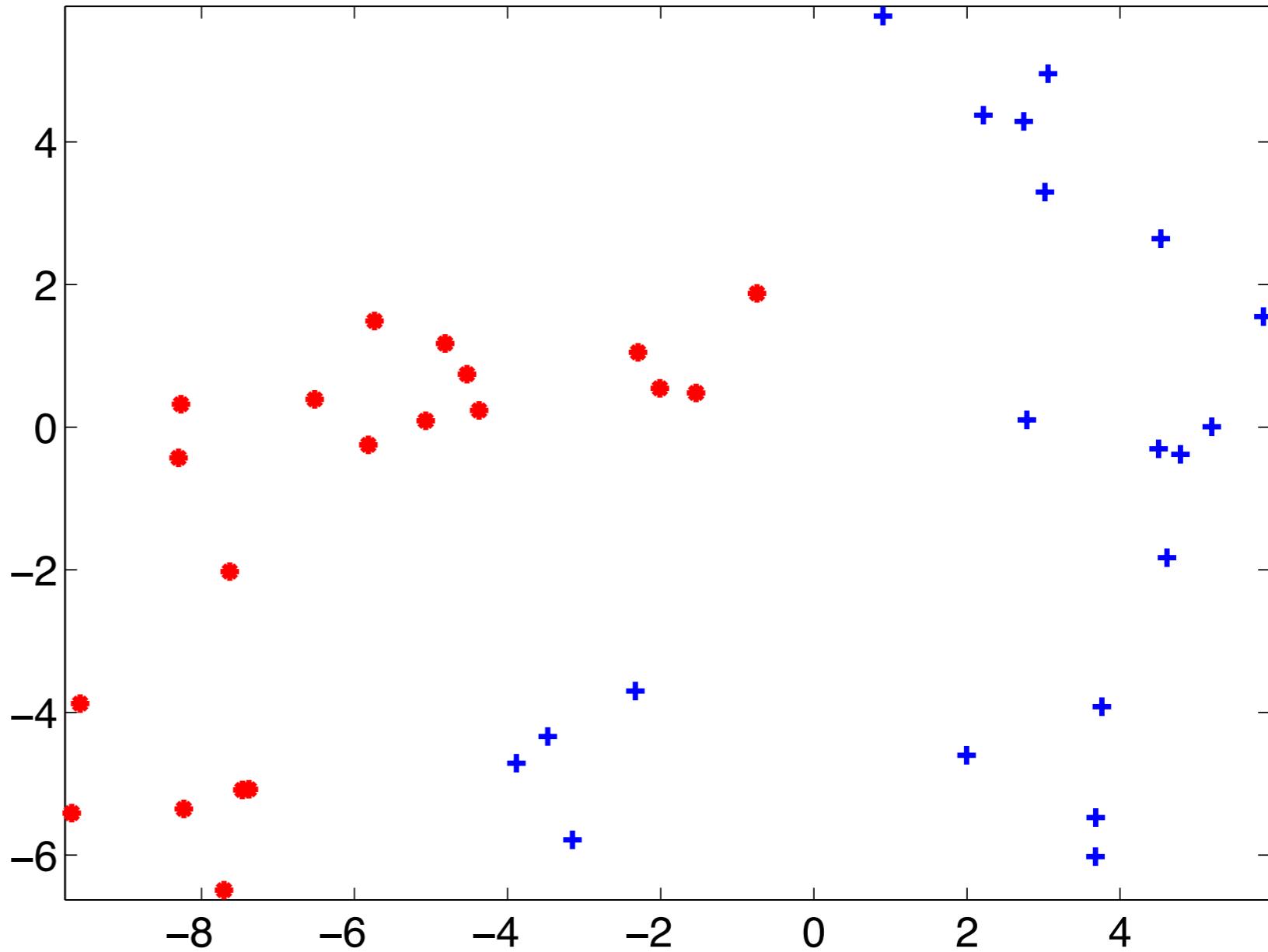
What classifier to use here?

- NMC
- LDC
- QDC
- k-NN
- Parzen classifier
- Logistic
- Support vector
- Decision tree
- Neural network
- ...



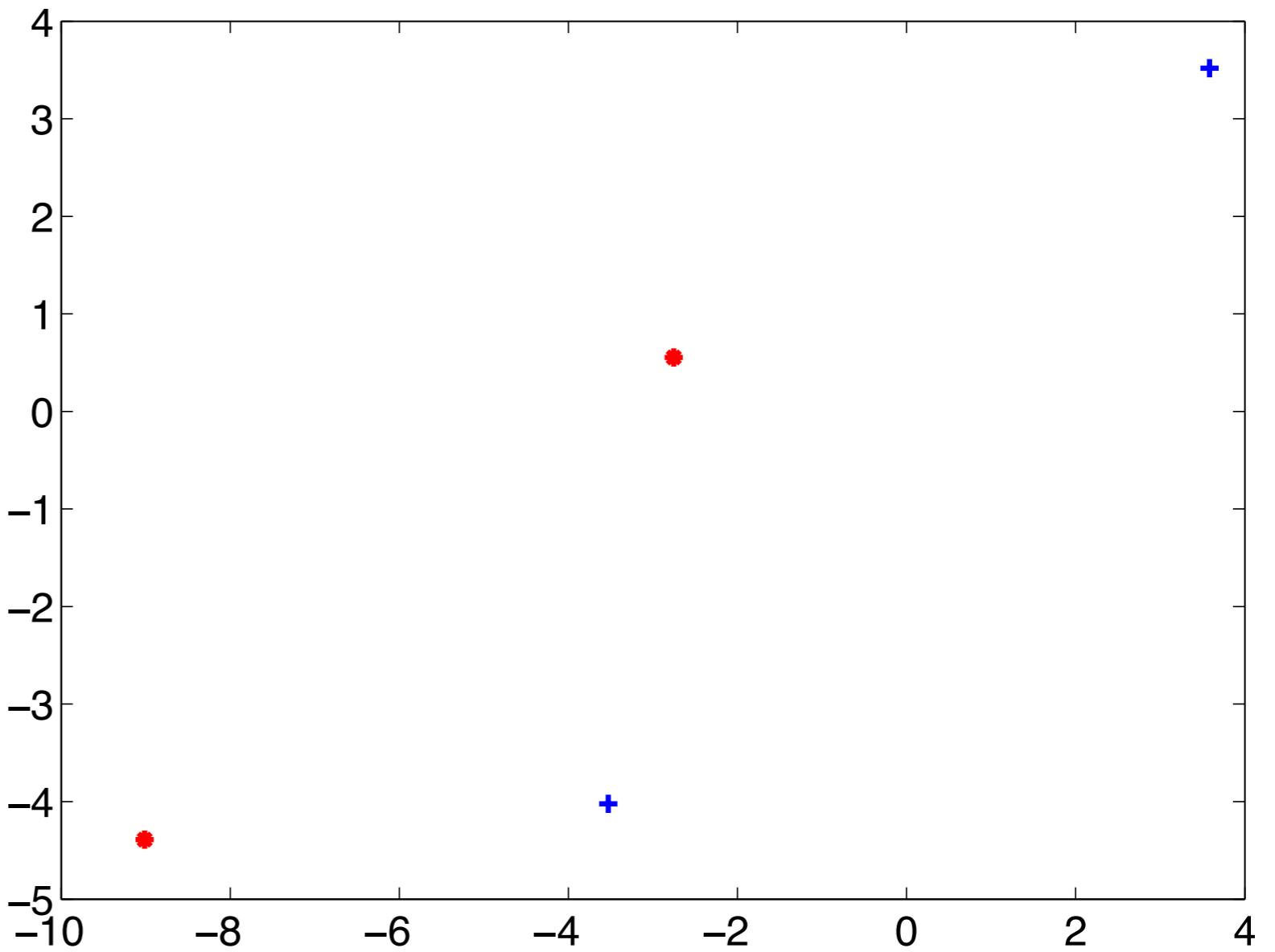
What classifier to use here?

- NMC
- LDC
- QDC
- k-NN
- Parzen classifier
- Logistic
- Support vector
- Decision tree
- Neural network
- ...



What classifier to use here?

- NMC
- LDC
- QDC
- k-NN
- Parzen classifier
- Logistic
- Support vector
- Decision tree
- Neural network
- ...



What classifier to use here?

- Loaded our famous Iris dataset
- 150 objects in 4D
- What now?

iris	=	sepal l.	sepal w.	petal l.	petal w.
		5.1000	3.5000	1.4000	0.2000
		4.9000	3.0000	1.4000	0.2000
		4.7000	3.2000	1.3000	0.2000
		4.6000	3.1000	1.5000	0.2000
		5.0000	3.6000	1.4000	0.2000
		5.4000	3.9000	1.7000	0.4000
		4.6000	3.4000	1.4000	0.3000
		5.0000	3.4000	1.5000	0.2000
		4.4000	2.9000	1.4000	0.2000
		4.9000	3.1000	1.5000	0.1000
		5.4000	3.7000	1.5000	0.2000
		4.8000	3.4000	1.6000	0.2000
		4.8000	3.0000	1.4000	0.1000
		4.3000	3.0000	1.1000	0.1000
		5.8000	4.0000	1.2000	0.2000
		5.7000	4.4000	1.5000	0.4000
		5.4000	3.9000	1.3000	0.4000

What classifier to use?

- If cannot visualise high-dimensional data: how to choose?
 - Need some criterion!
 - Typically: classification performance/error
- (1) Test it on independent data
- (2) What if we still get the same error?

True or false?

- The classification error of the training set is a good measure of the true classification error
TRUE? FALSE?
 - A good estimate of the actual, true, error is all we are after
TRUE? FALSE?

What classifier to use?

- If cannot visualise high-dimensional data: how to choose?
- Need some criterion!
- Typically: classification performance/error
- Test it on independent data
- For simplicity we assume now that classification error is good enough

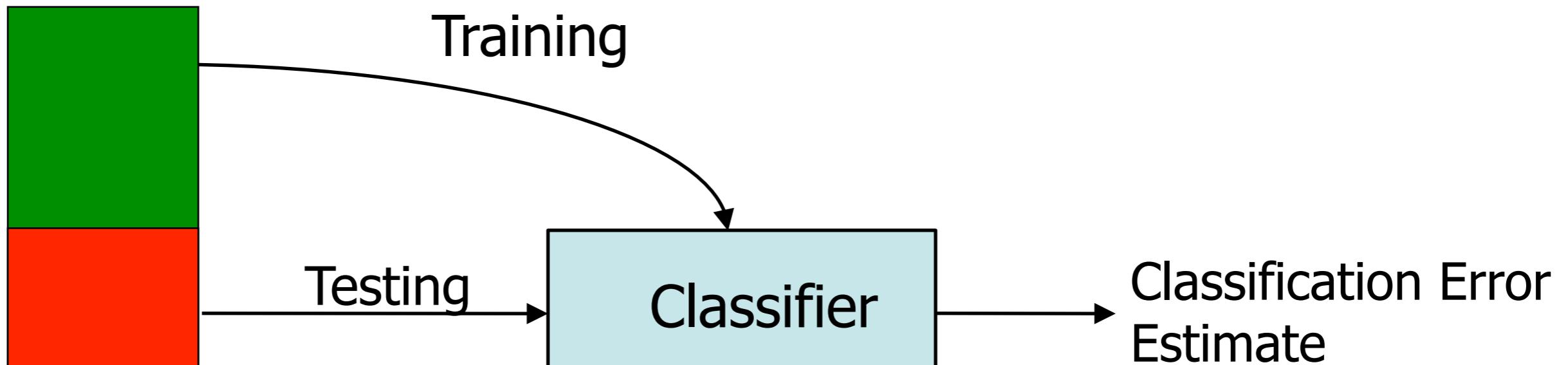
Material/literature

- Chapter 19 “Design and Analysis of Machine Learning experiments” in ‘Introduction to Machine Learning’ by E. Alpaydin
- Ebook available at library.tudelft.nl
- Some sections are not discussed (19.8 and further)
- ‘Reject option’ is (unfortunately) not in the book: get it from the slides.

Classifier evaluation

- This lecture:
 - Train, validation and test set
 - Learning curve
 - Bootstrapping, cross-validation, leave-one-out
 - Classifier complexity
 - Feature curves
- Second lecture:
 - Confusion matrices,
 - Reject curve
 - ROC curve
- Question hour

Error Estimation by Test Set



Design Set

Other training set → other classifier

Other test set → other error estimate $\hat{\epsilon}$

Some Error Estimate Variability

- Error is a sum of Bernoulli random variables:

$$\hat{\varepsilon} = \frac{1}{N} \sum_{i=1}^N Z_i$$

where

$$Z_i = \begin{cases} 0 & \text{if } \mathbf{x}_i \text{ is correctly classified} \\ 1 & \text{if } \mathbf{x}_i \text{ is incorrectly classified} \end{cases}$$

- You can show that the variance is:

$$\sigma_{\hat{\varepsilon}}^2 = \text{Var}(\hat{\varepsilon} \mid \text{test set size } N) = \frac{\varepsilon(1 - \varepsilon)}{N}$$

Some Error Estimate Variability

- Error is a sum of Bernoulli random variables:

$$\hat{\varepsilon} = \frac{1}{N} \sum_{i=1}^N Z_i$$

- When: $\sigma_{\hat{\varepsilon}}^2 = \text{Var}(\hat{\varepsilon} \mid \text{test set size } N) = \frac{\varepsilon(1 - \varepsilon)}{N}$

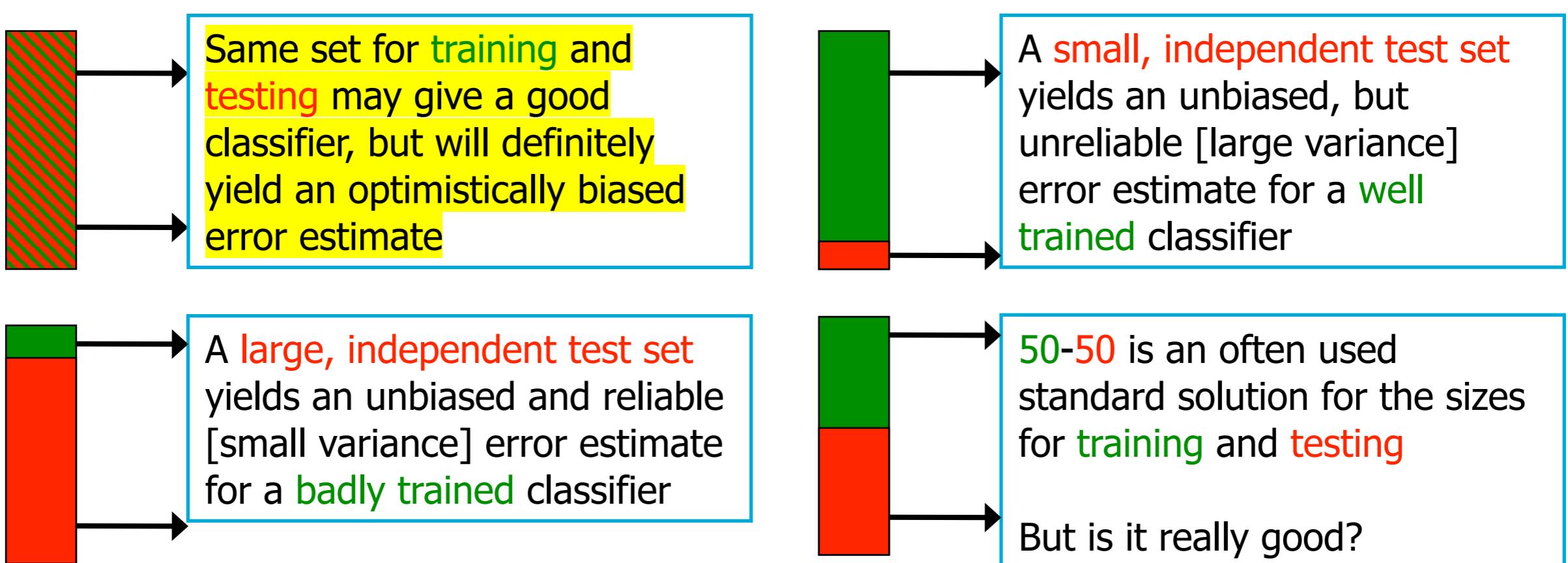
you can compute the standard deviation for different sample sizes and errors:

$$\sigma_{\hat{\varepsilon}} = \sqrt{\frac{\varepsilon(1 - \varepsilon)}{N}}$$

N	ε	0.01	0.03	0.1
10	0.031	0.054	0.095	
100	0.010	0.017	0.030	
1000	0.003	0.005	0.0095	

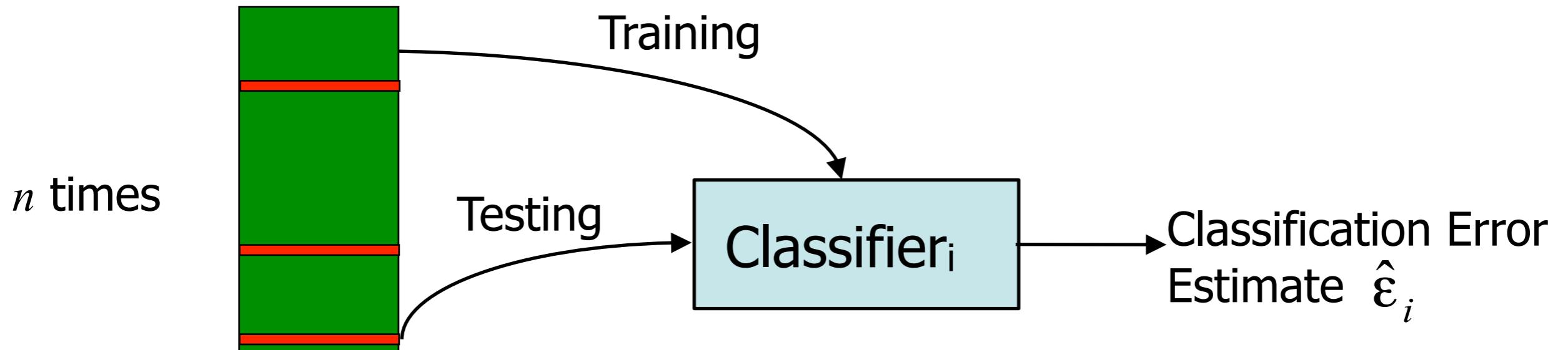
Training Set Size vs. Test Set Size

- Large training set -> good classifiers
- Large test set -> reliable, unbiased error estimate
- In practice often just a single design set is given



Bootstrapping

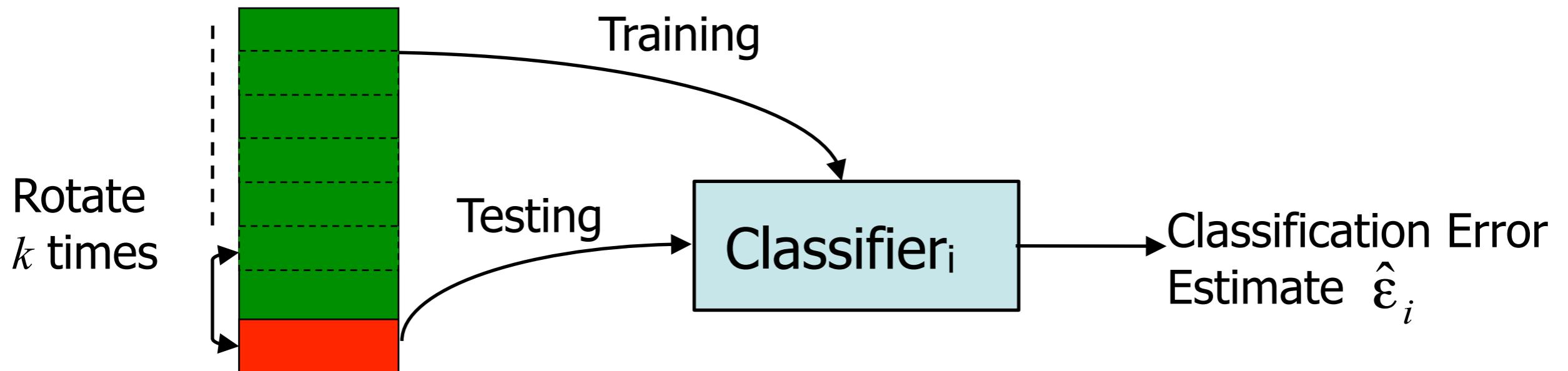
- Randomly draw N objects from N objects (**with replacement**)



$$\hat{\varepsilon} = \frac{1}{m} \sum_i \hat{\varepsilon}_i$$

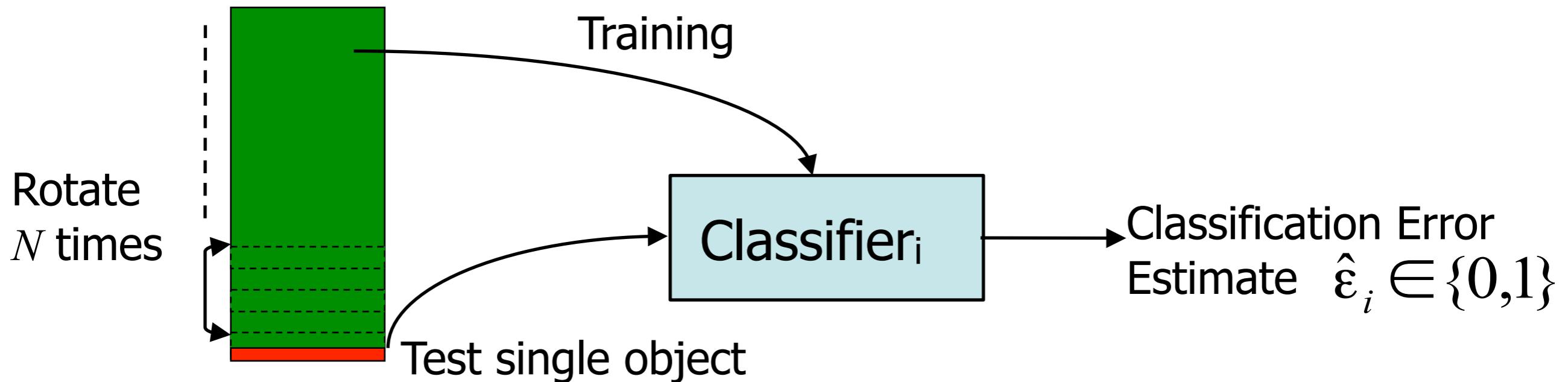
- Left-over objects are used for testing
- Repeat m times

k-fold cross validation



$$\hat{\epsilon} = \frac{1}{k} \sum_i \hat{\epsilon}_i$$

Leave-one-out Procedure



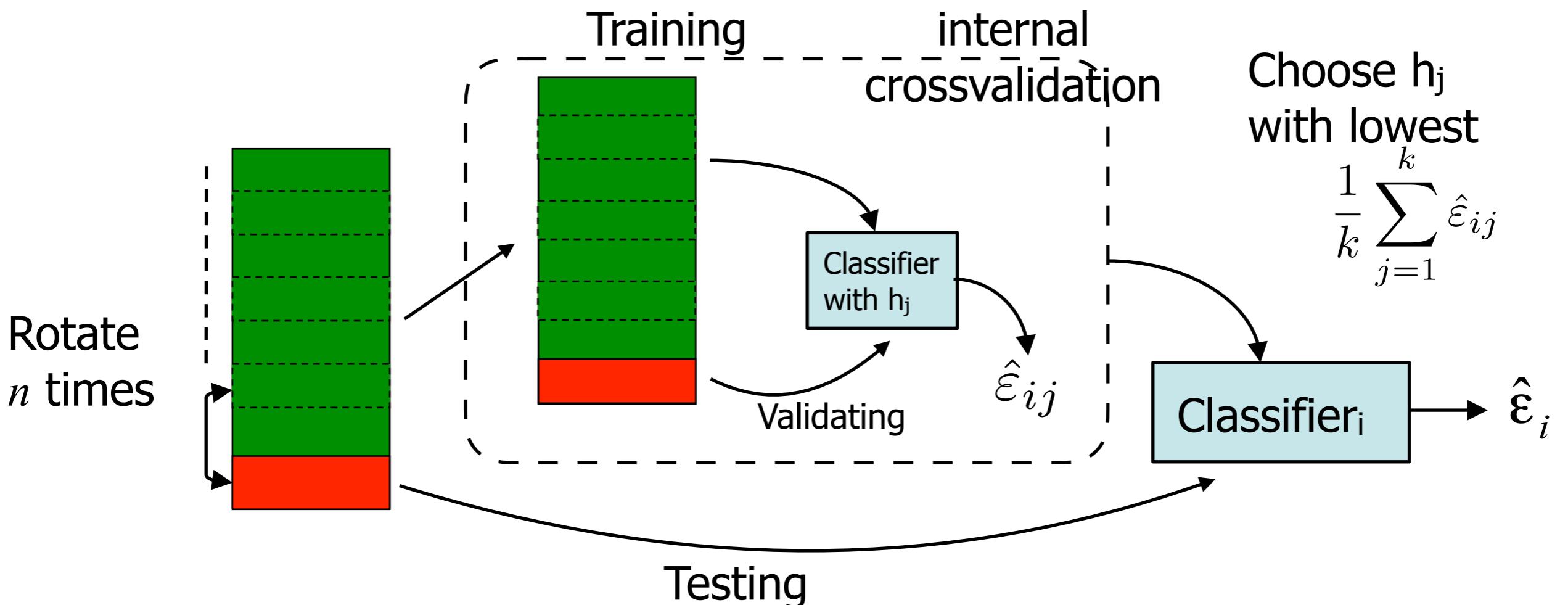
$$\hat{\varepsilon} = \frac{1}{N} \sum_i \hat{\varepsilon}_i$$

Optimising 'hyperparameters'

- Machine learning methods often have 'hyperparameters'
 - Parzen density estimator: width parameter h
 - k-nearest neighbour: number of neighbours k
 - Decision trees: pruning method, stopping criterion
 - Neural networks: architecture, learning rate, ...
- DON'T optimise these numbers by looking at the test set!
Then you're **CHEATING!**

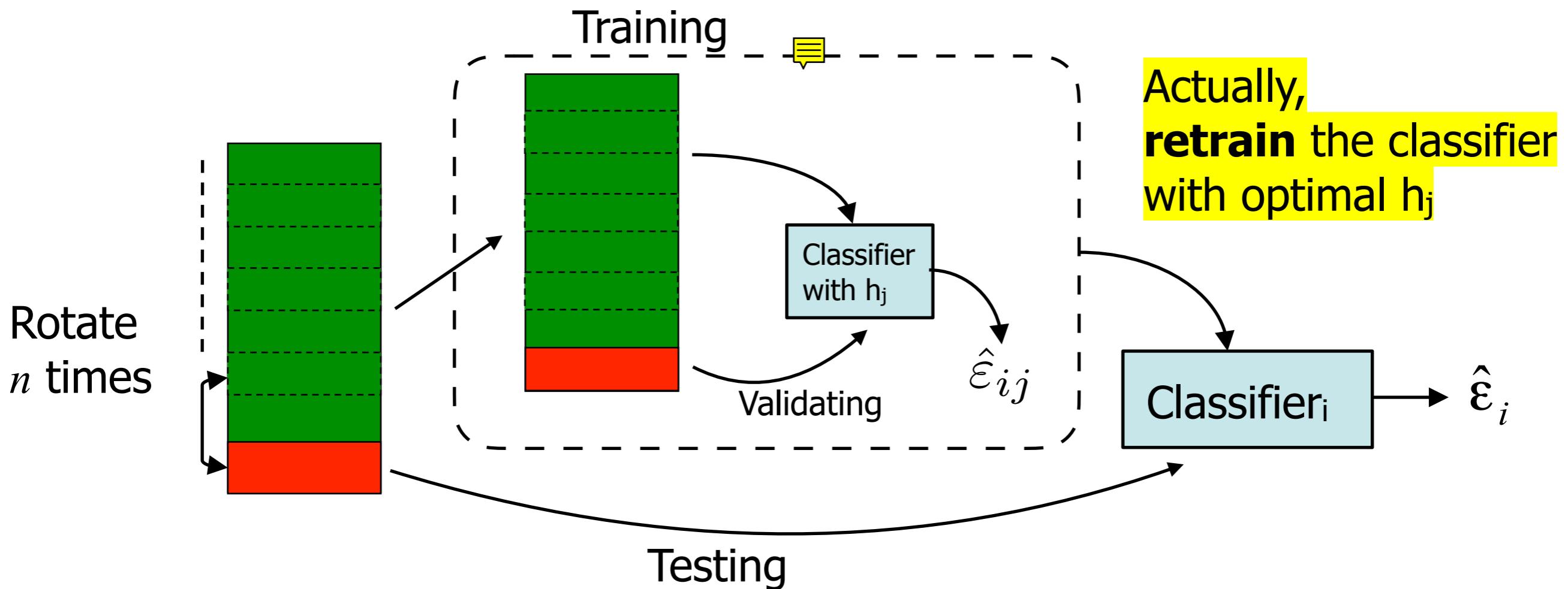
Double cross-validation

- To optimise hyperparameters, do cross-validation **inside** another cross-validation:

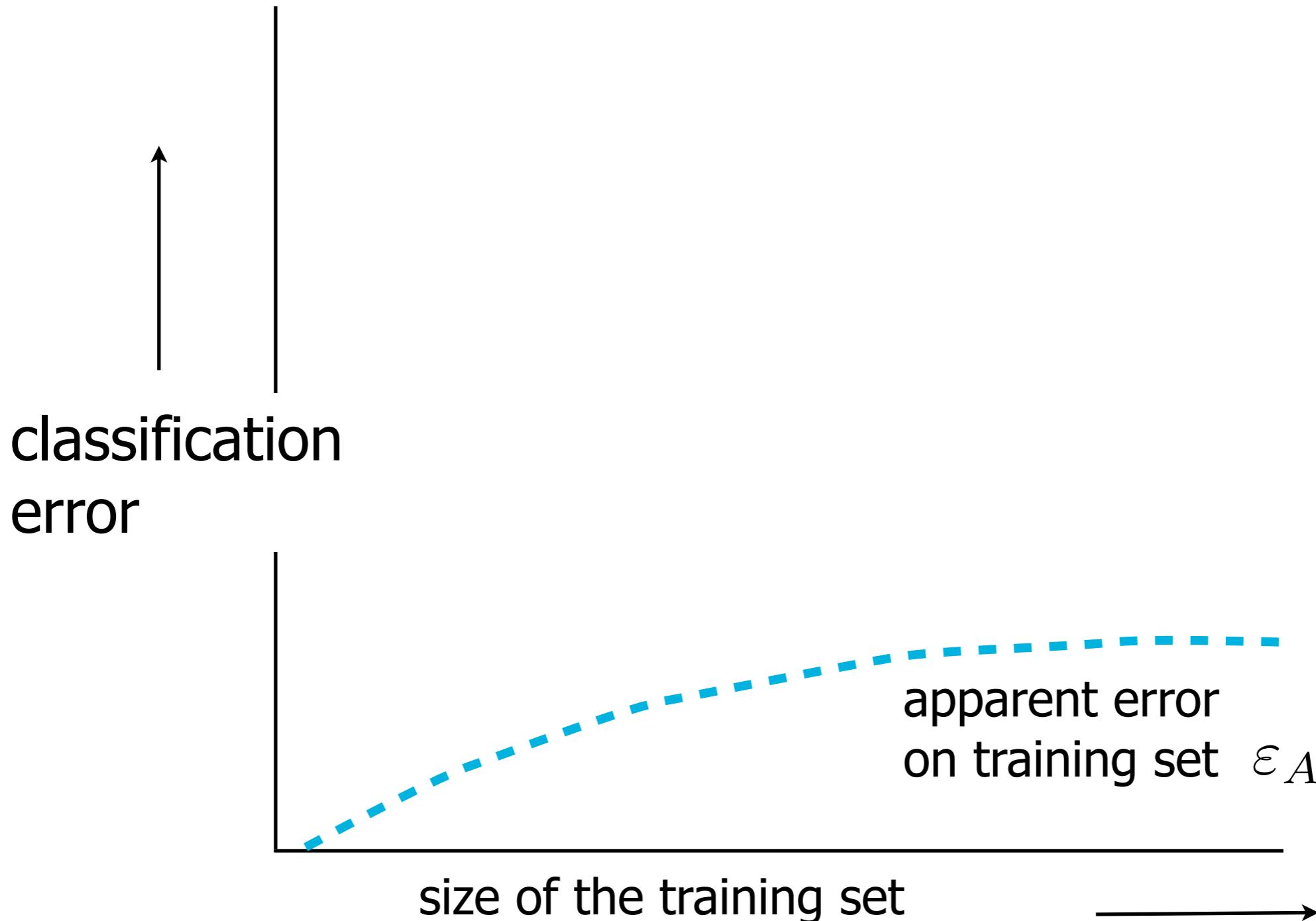


Double cross-validation

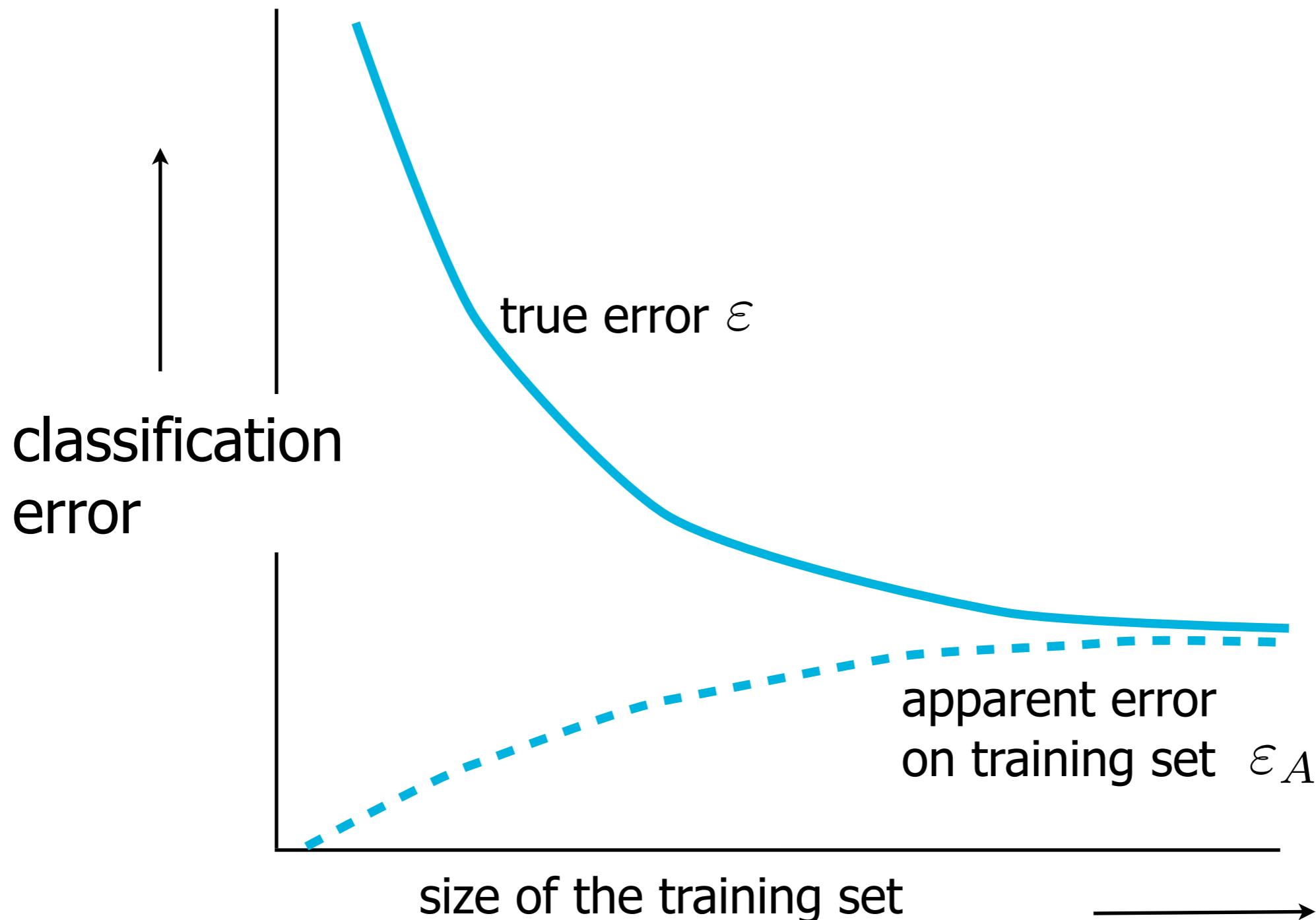
- To optimise hyperparameters, do cross-validation **inside** another cross-validation:



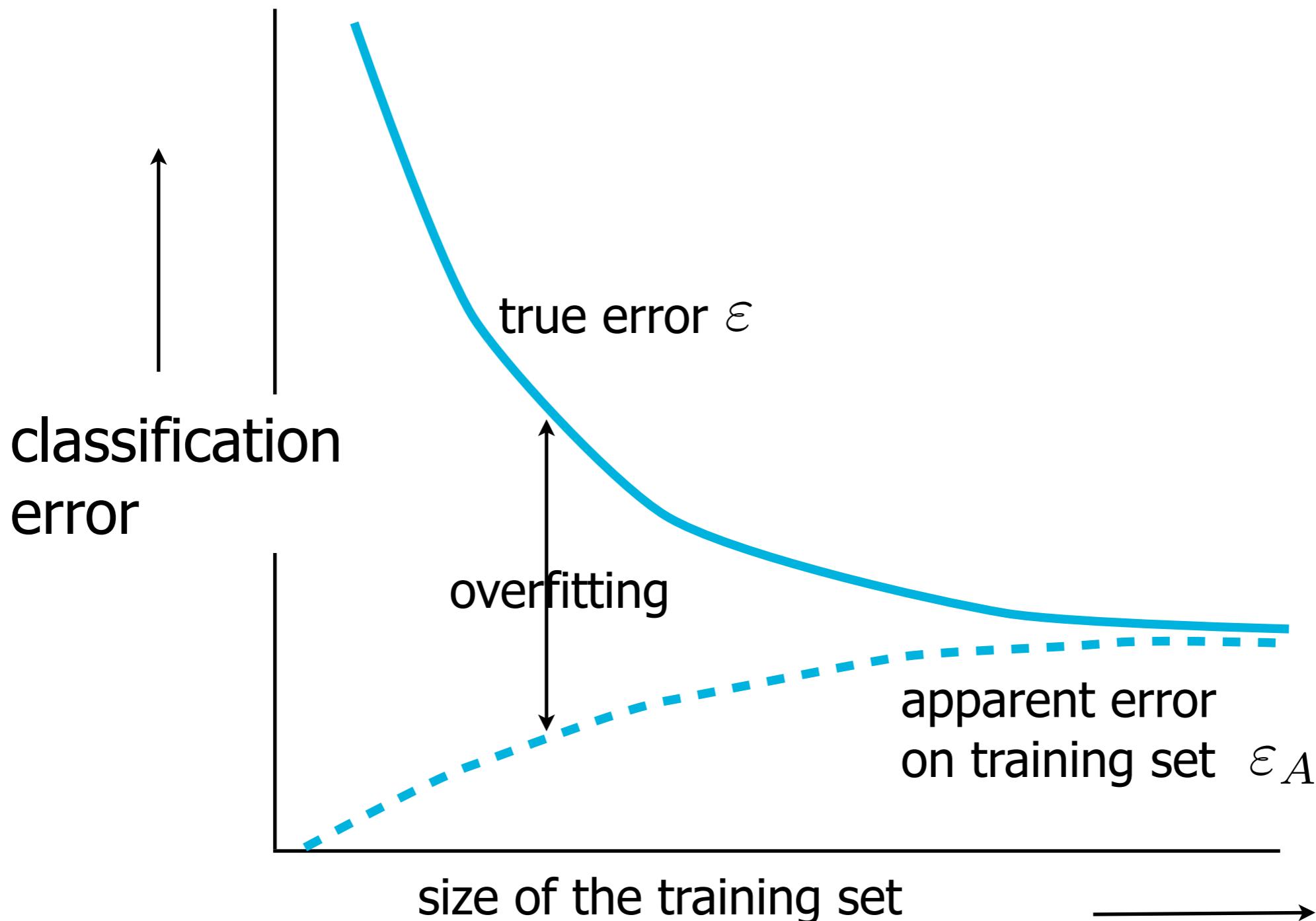
Apparent Classification error



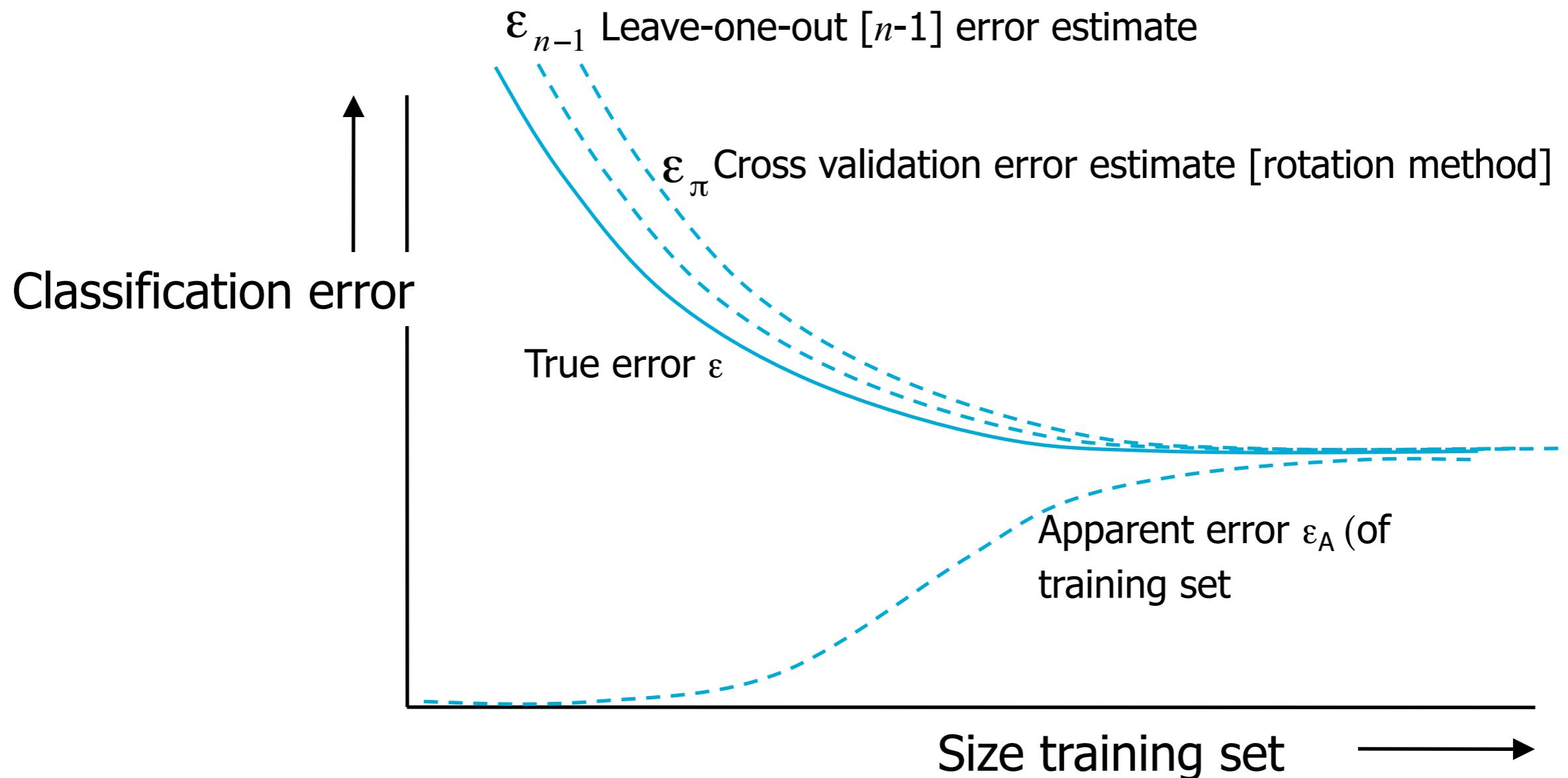
Classification error



Apparent Classification error



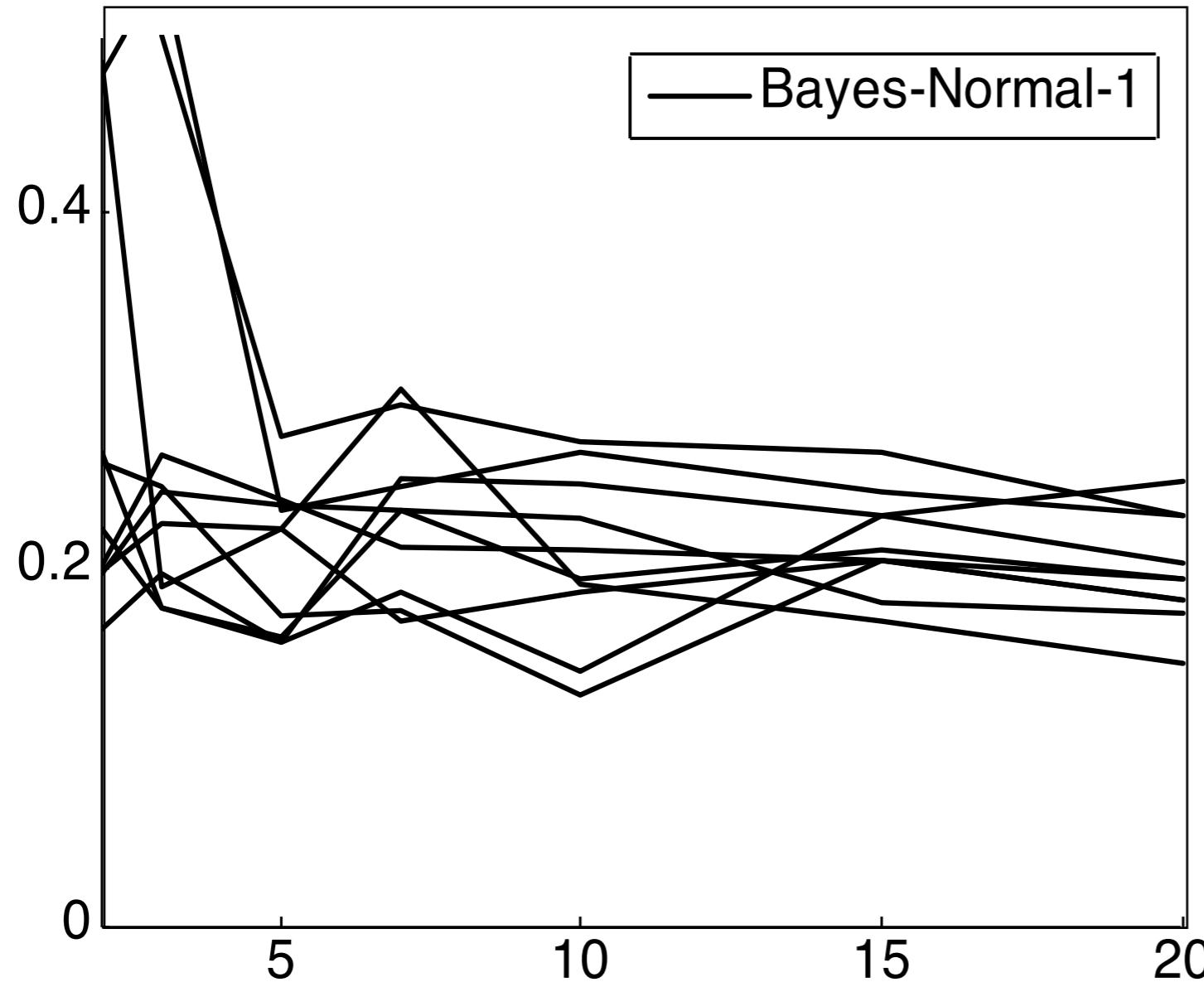
Cross Validation Curves [and Related]



Learning Curves

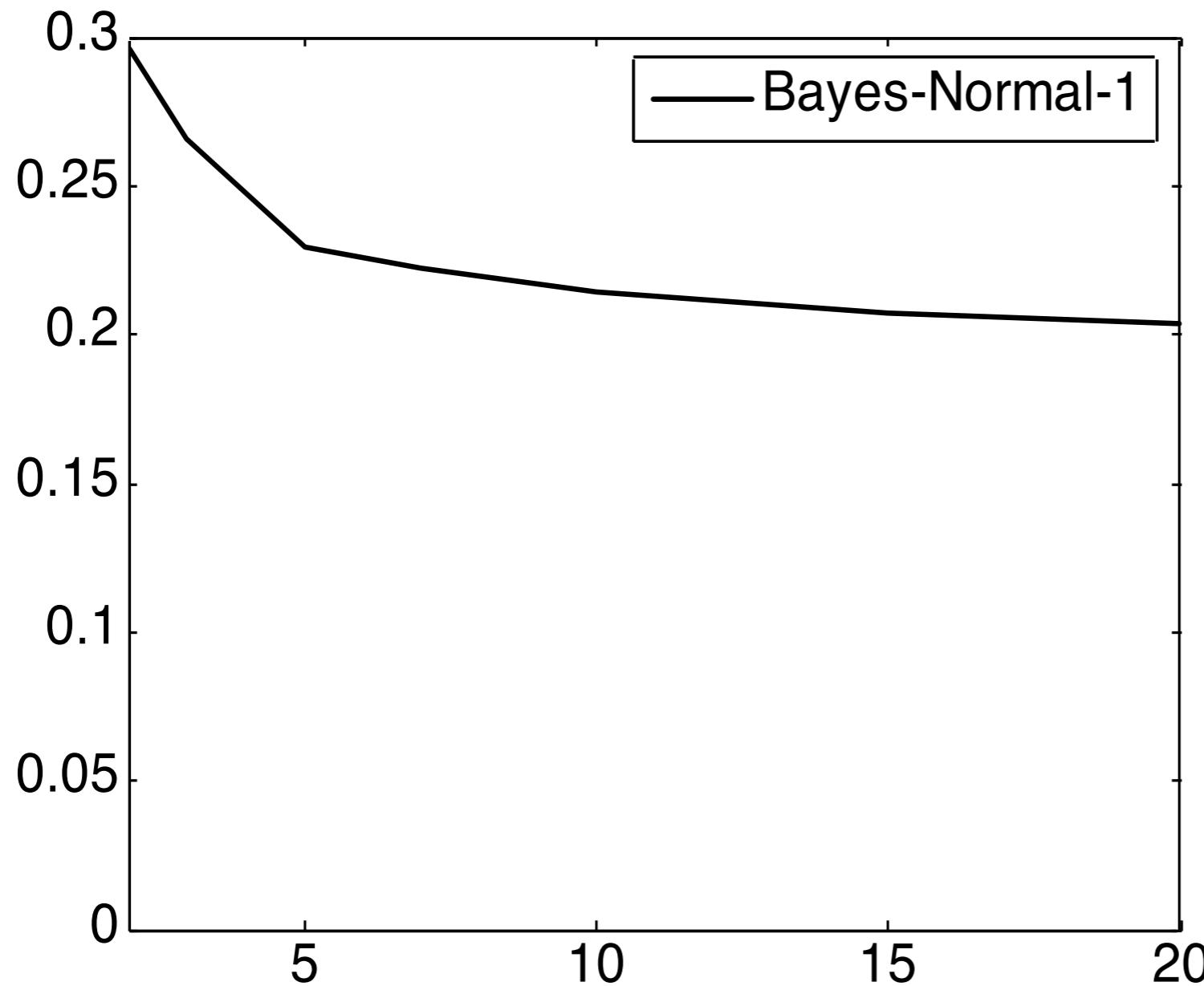
- Curves that plot [estimated] classification errors against the number of samples in training set
 - Usually plot error both on training and on test set
 - Gives insight in, e.g.
 - Amount of overtraining
 - Usefulness of additional data
 - How different classifiers compare
 - Stability of training
 - ...

Repeated Learning Curves



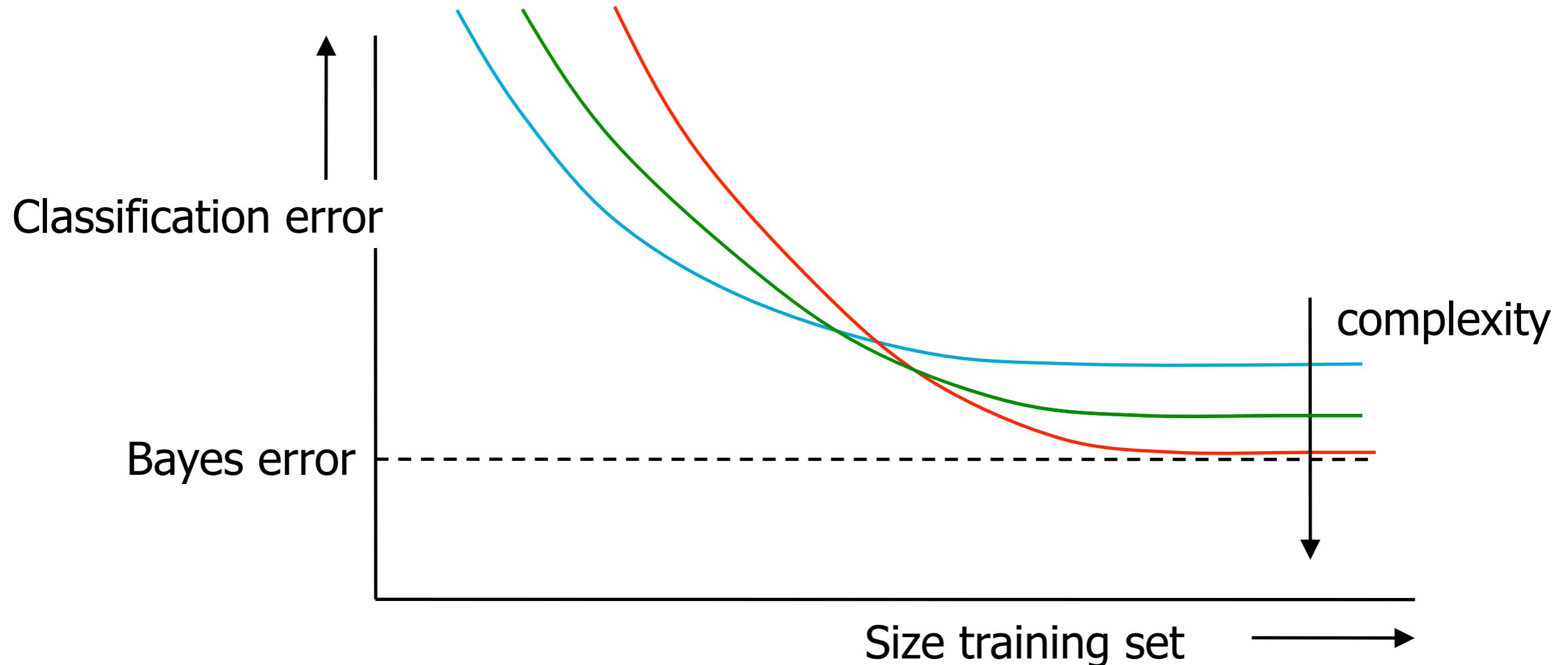
Small sample sizes have
a very large variability

Averaged Learning Curve

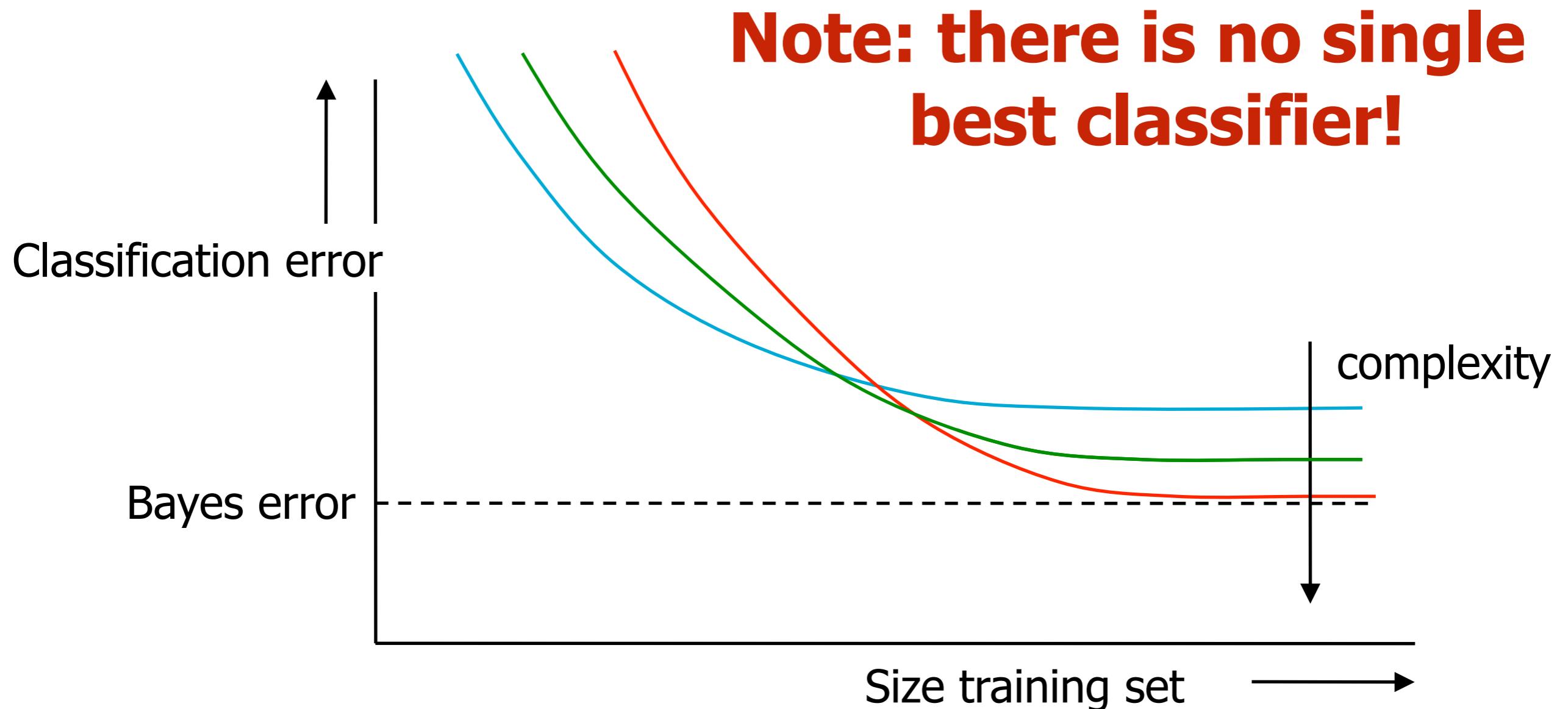


Average a few times

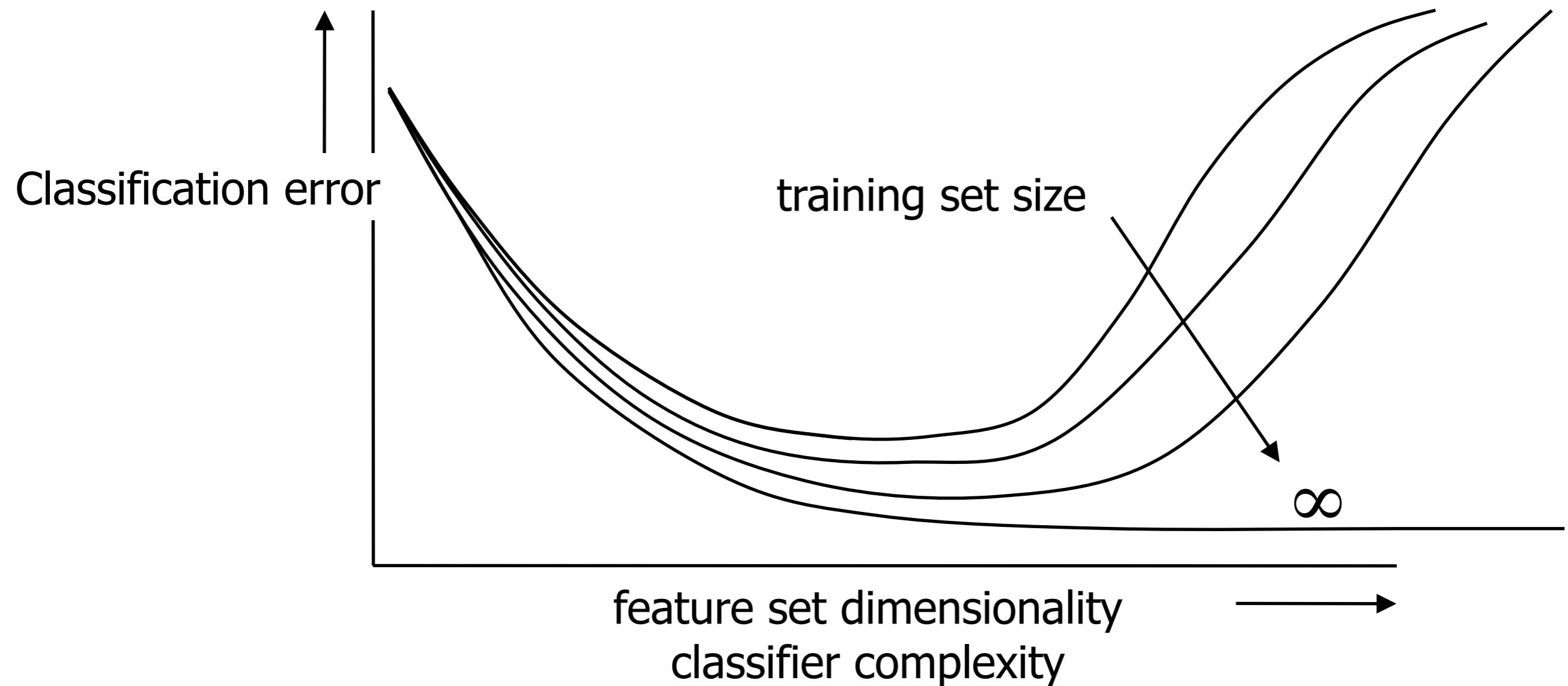
Different Classifier Complexity



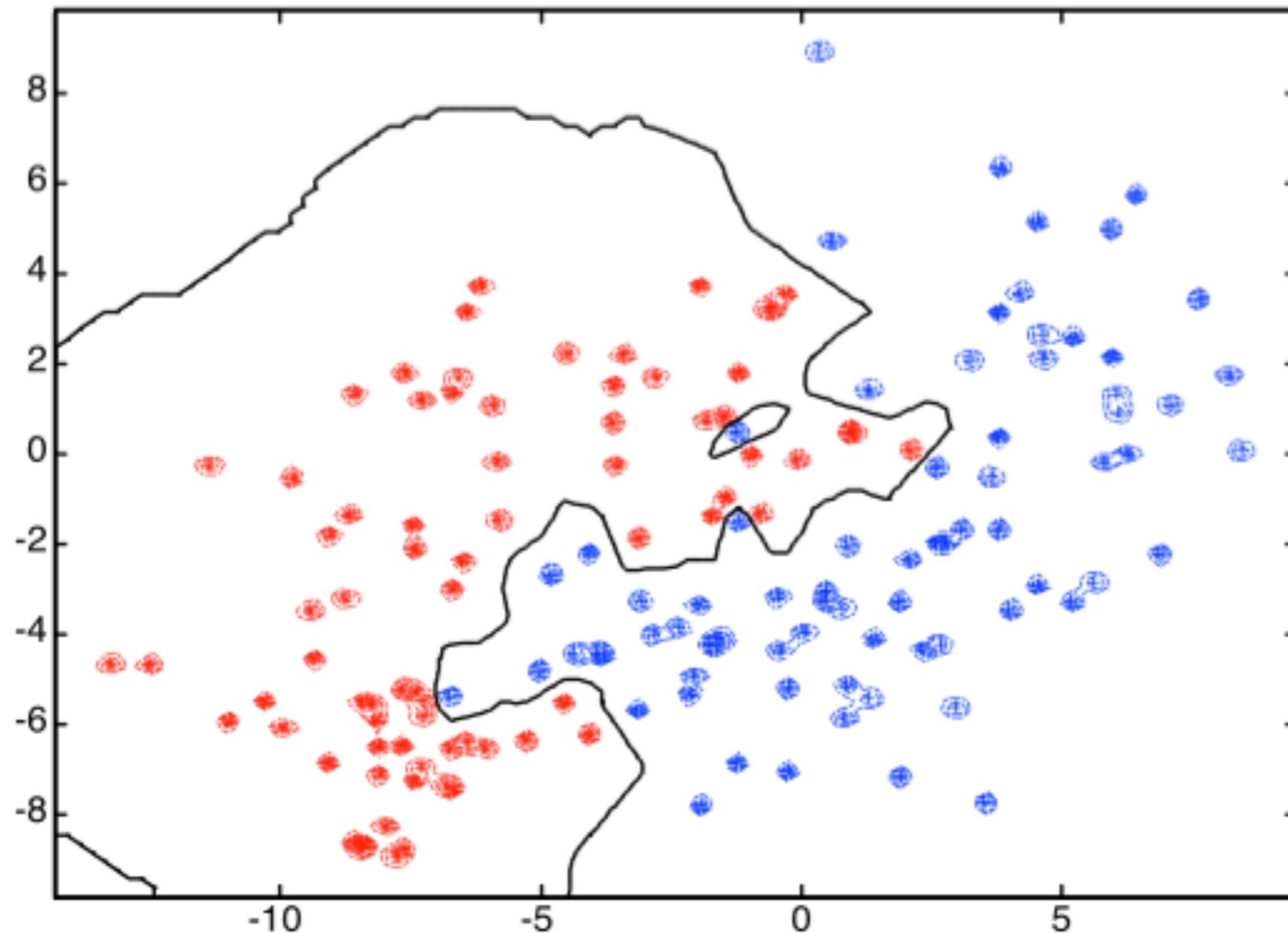
Different Classifier Complexity

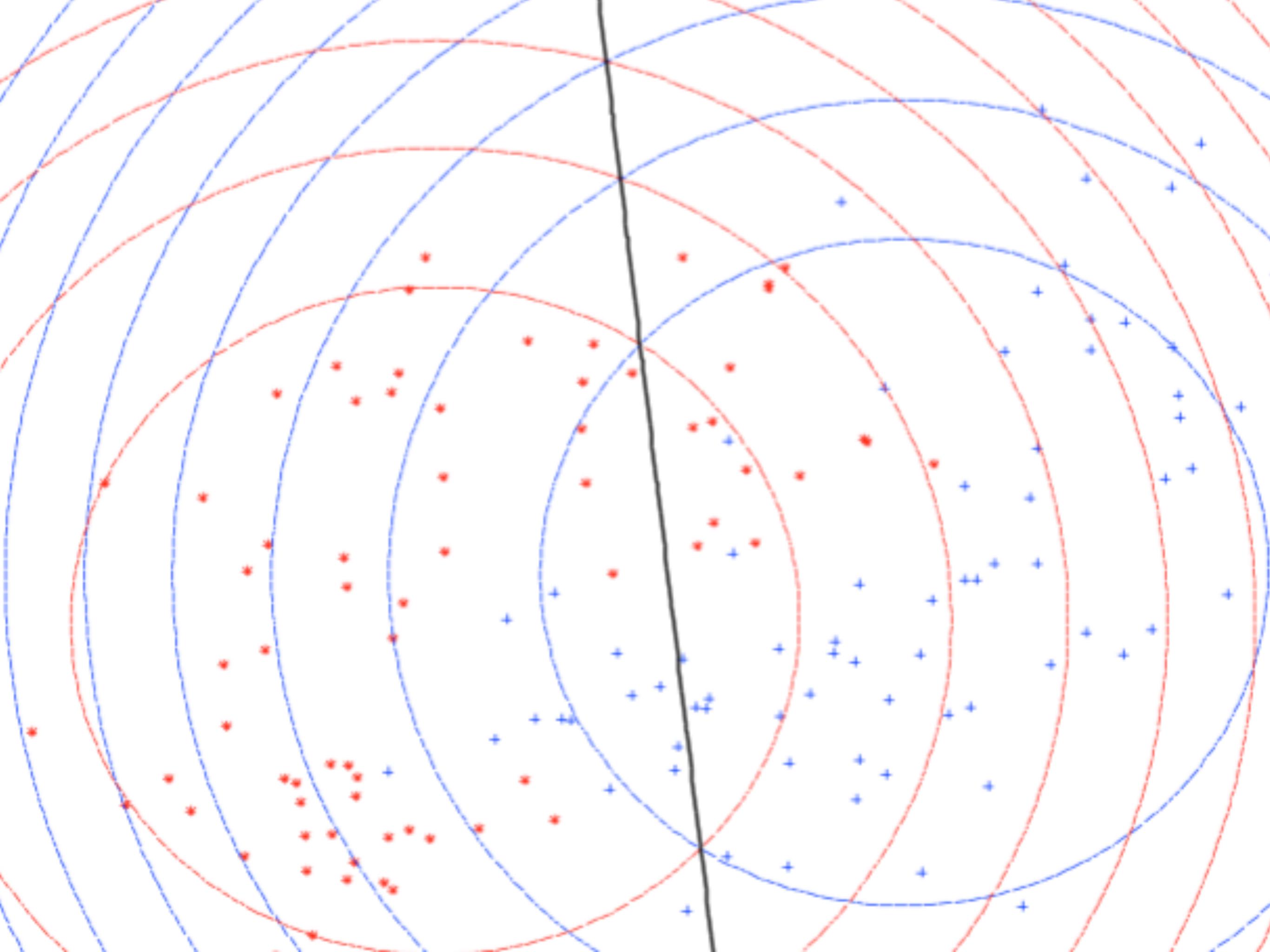


Feature Curves

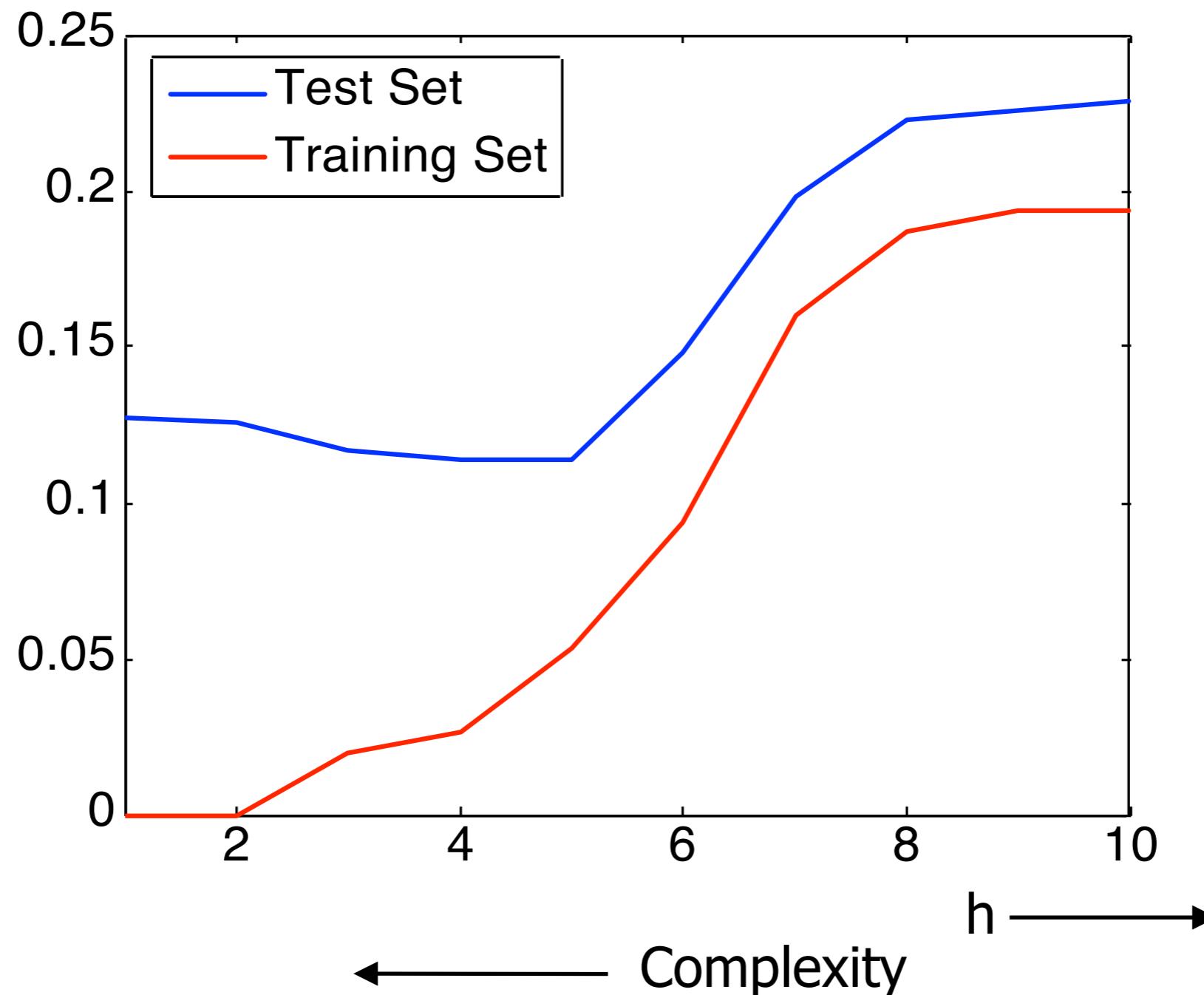


Parzen Complexity Example





Parzen Complexity Example



Some intermediate conclusions...

- Larger training sets yield better classifiers
- Independent test sets needed for unbiased error estimates
- Larger test sets yield more accurate error estimates
- LOO cross validation “optimal”, but might be infeasible
- 10-fold crossvalidation is often used
- More complex classifiers need larger training sets
 - Same holds for larger feature set sizes
- Small training sets need simpler classifiers or smaller feature sets

Squared Error

- When we have the error

$$L(\mathbf{w}) = E[\|\mathbf{w}^T \mathbf{x} - y\|^2]$$

we can actually derive something more general...

- In general?!
- Unfortunately, theory in Pattern Recognition is tough...
how to deal with **all** possible datasets?

Bias-variance dilemma

- When we are given some data, we may get lucky, or unlucky:
sometimes we get very aypical data:-()

- To say something general, we need to **average** over different (training-) sets

$$\mathcal{D} = \{(y_i, \mathbf{x}_i); i = 1, \dots, N\}$$

- The classifier is now also a function of the training set:
 $g(\mathbf{x}; \mathcal{D})$

Bias-variance dilemma

- Consider the squared error:

$$E_{\mathcal{D}} [(g(\mathbf{x}; \mathcal{D}) - E[y|\mathbf{x}])^2]$$

- $E[y|\mathbf{x}]$ is the optimal mean-squared regressor (not proven now; see book)
- Now we do a trick:

$$\begin{aligned} &= E_{\mathcal{D}} \left[\underbrace{(g(\mathbf{x}; \mathcal{D}) - E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})]}_{+ E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})] - E[y|\mathbf{x}])^2} \right. \\ &\quad \left. + \underbrace{E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})] - E[y|\mathbf{x}])^2}_{\text{---}} \right] \end{aligned}$$

Bias-variance dilemma

- Now working out the square:

$$\begin{aligned} &= E_{\mathcal{D}} \left[(\underline{g(\mathbf{x}; \mathcal{D})} - \underline{E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})]})^2 \right. \\ &\quad + 2(\underline{g(\mathbf{x}; \mathcal{D})} - \underline{E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})]})(\underline{E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})]} - \underline{E[y|\mathbf{x}]}) \\ &\quad \left. + (\underline{E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})]} - \underline{E[y|\mathbf{x}]})^2 \right] \end{aligned}$$

$$\begin{aligned} &= E_{\mathcal{D}} \left[(g(\mathbf{x}; \mathcal{D}) - E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})])^2 \right] \xrightarrow{\text{(variance)}} = 0 \\ &\quad + \boxed{E_{\mathcal{D}} [2(g(\mathbf{x}; \mathcal{D}) - E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})])(E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})] - E[y|\mathbf{x}])]} \\ &\quad + E_{\mathcal{D}} \left[(E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})] - E[y|\mathbf{x}])^2 \right] \xrightarrow{\text{(bias}^2)} \end{aligned}$$

Bias-variance dilemma

(variance)

$$\begin{aligned} MSE &= E_{\mathcal{D}} \left[(g(\mathbf{x}; \mathcal{D}) - E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})])^2 \right] \\ &\quad + E_{\mathcal{D}} \left[(E_{\mathcal{D}}[g(\mathbf{x}; \mathcal{D})] - E[y|\mathbf{x}])^2 \right] \end{aligned}$$

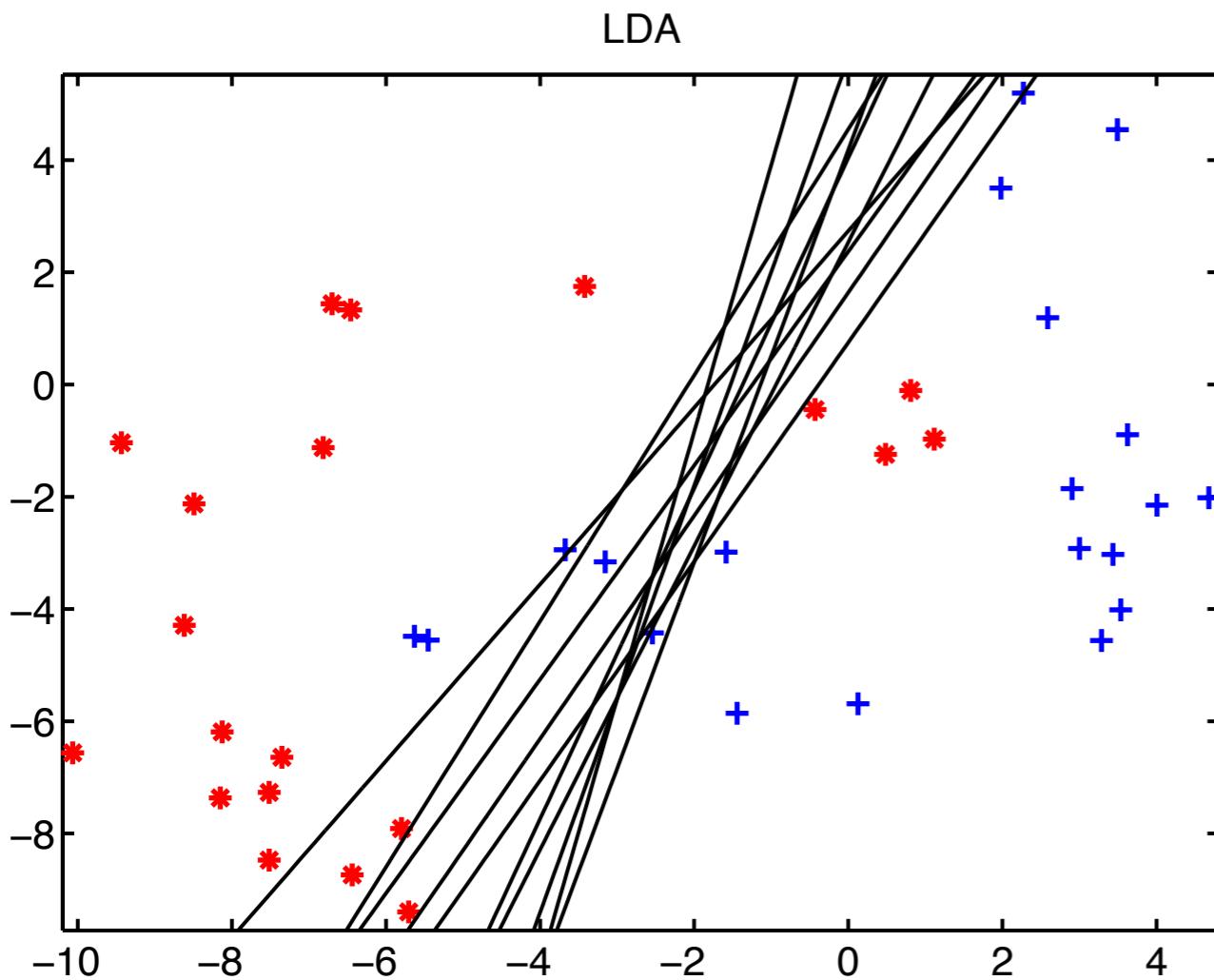
(bias²)

- variance: how much does classifier g vary over different training sets
- bias: how much does the average classifier g differ from the true output

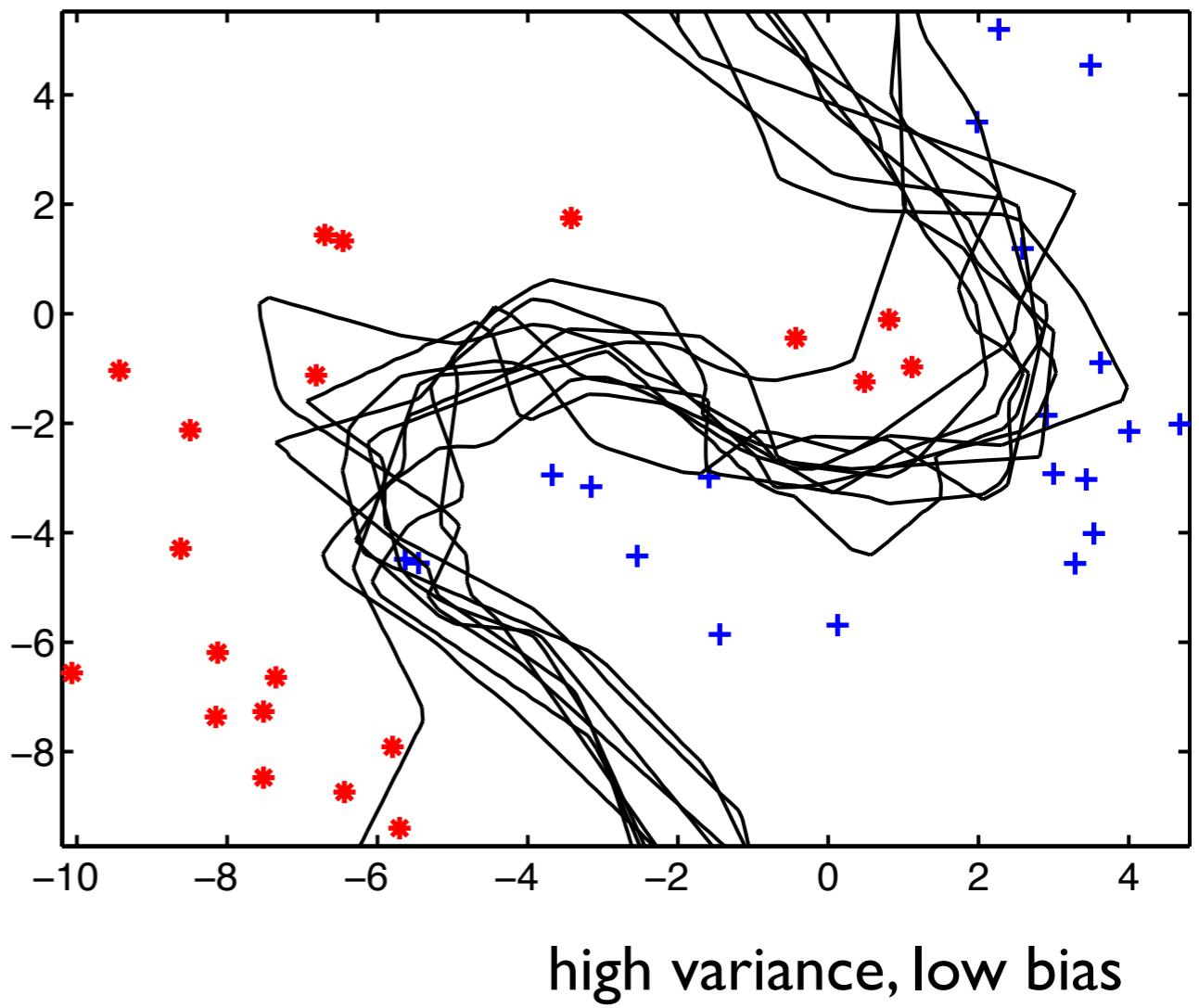
Bias-variance dilemma

- Compare LDA with kNN:

low variance, high bias



kNN

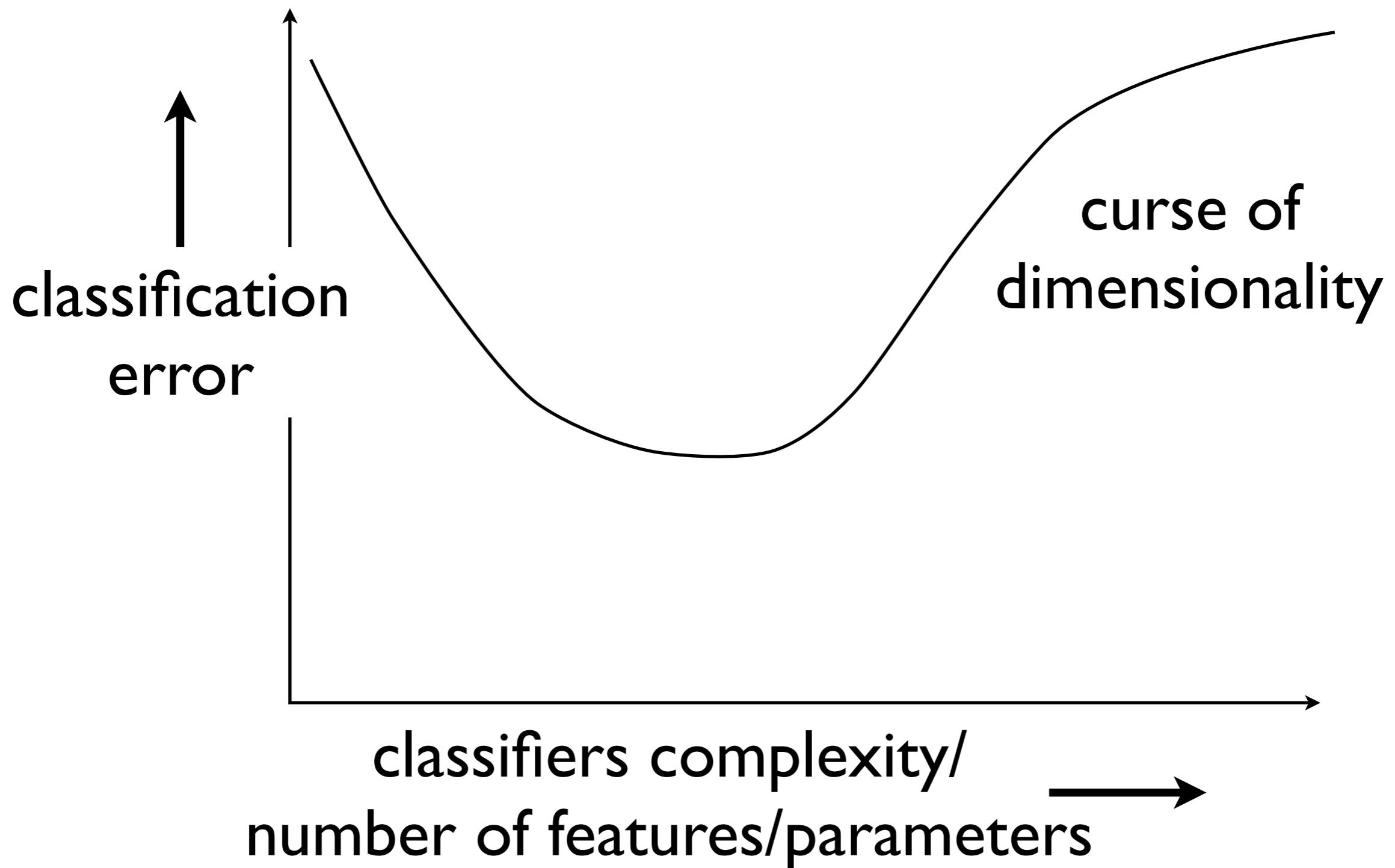


high variance, low bias

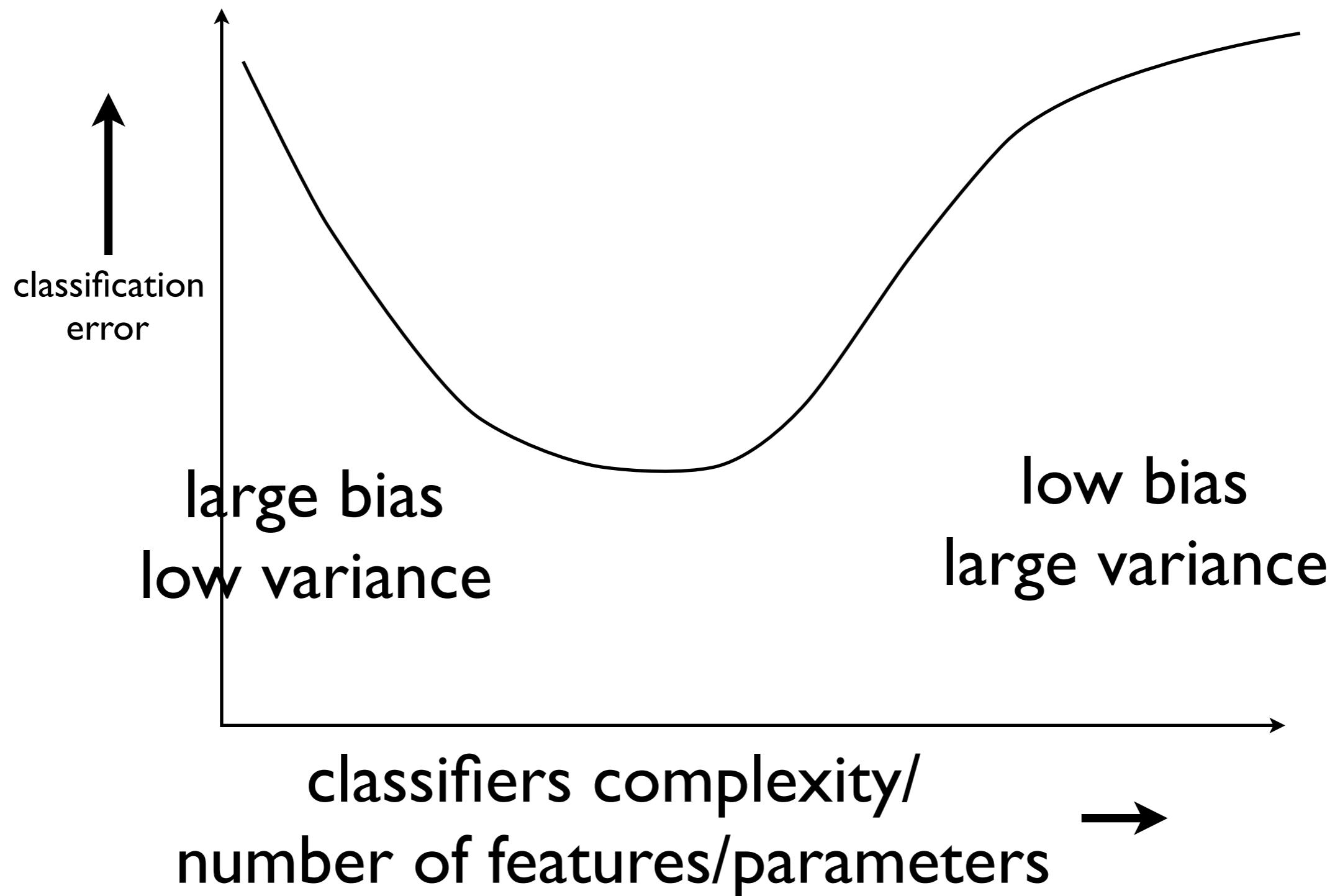
Bias-variance tradeoff

- Originally derived for neural networks
- It is a very general phenomenon: we encounter it more often in pattern recognition
- More simple classifier is more stable (and need less data to train)
- More complex classifier only works when you have sufficient number of training data

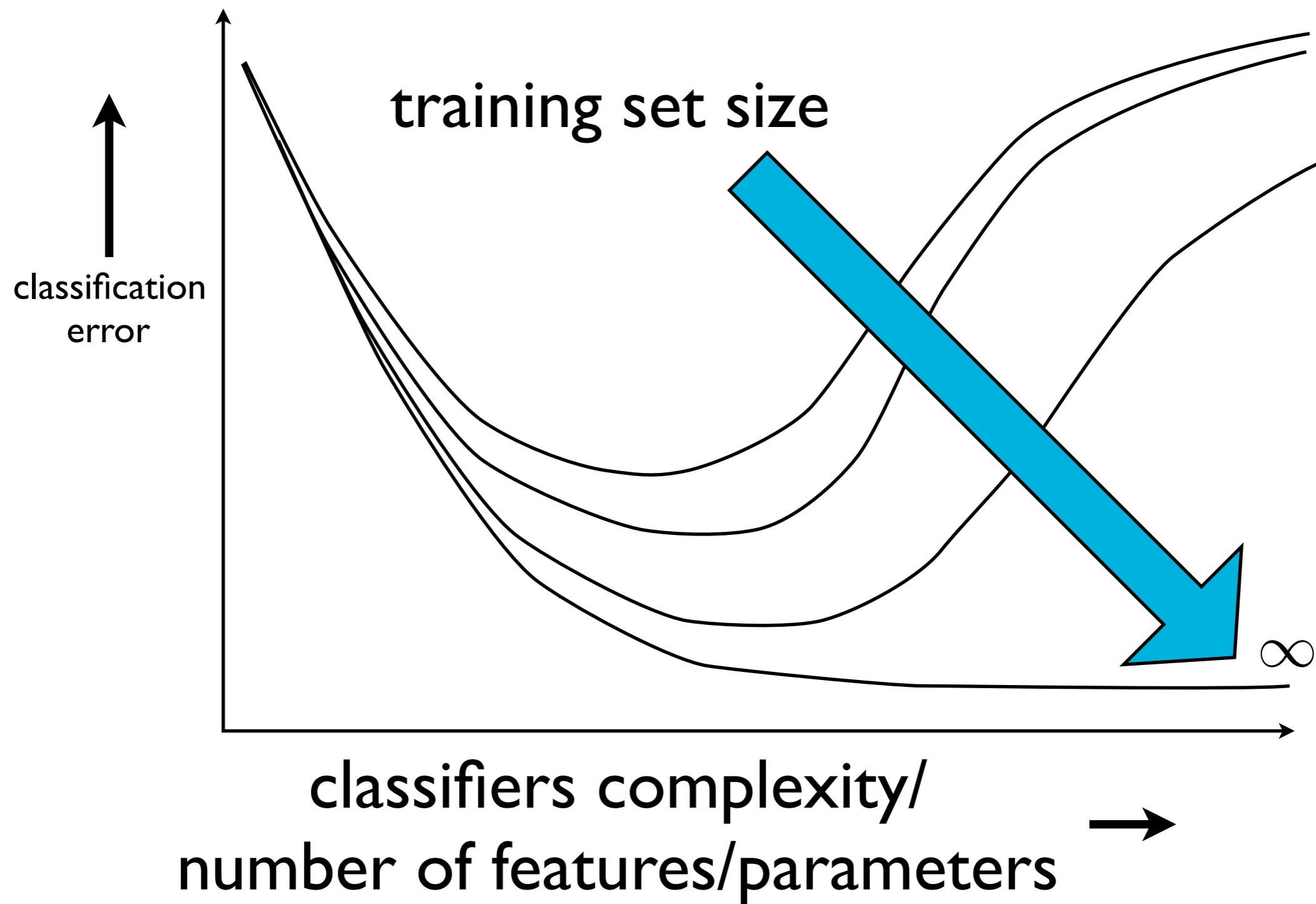
Feature curve



Feature curve



Feature curve



Regularisation

- (Almost) all machine learning methods optimise a loss/cost function like:

$$L = \frac{1}{N} \sum_{i=1}^N l(f(\mathbf{x}_i; \mathbf{w}), y_i)$$

- This holds for NMC, LDA, QD, logistic classifier, perceptron, MLP, SVM, decision tree
- Sometimes people add an additional regulariser $\Omega(\mathbf{w})$:

$$L = \frac{1}{N} \sum_{i=1}^N l(f(\mathbf{x}_i; \mathbf{w}), y_i) + \lambda \Omega(\mathbf{w})$$

regularisation parameter regulariser

Conclusions

- There is no best classifier
- There are alternative principles to find a good classifier:
 - minimising the classification error
 - minimising the mean squared error
 - maximising the likelihood
 - ...
- There is a fundamental tradeoff between the bias and variance of a classifier (depending on how flexible/complex a classifier is)
- Finding the correct regulariser is the ‘black art’ of ML

