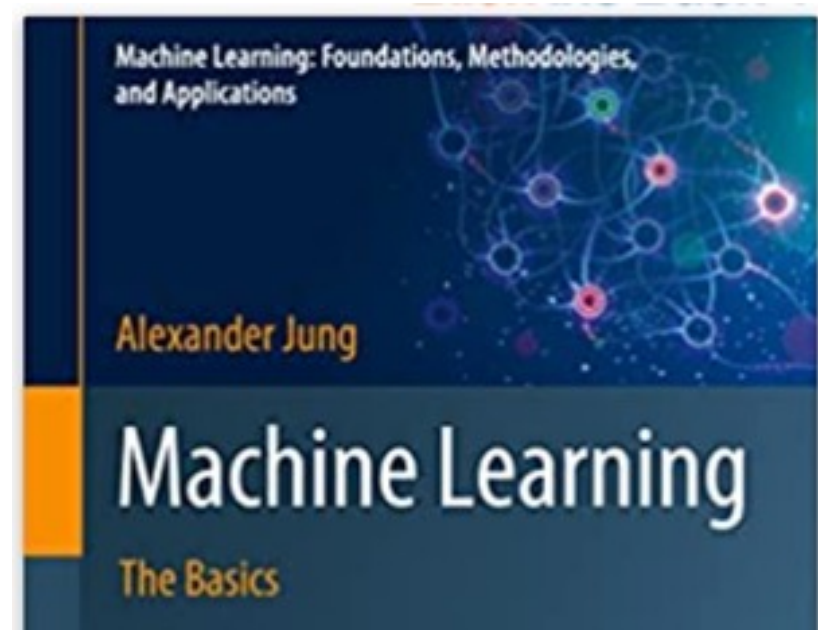


# Model Validation and Selection

Alexander Jung  
Assoc. Professor for Machine Learning  
Department of Computer Science  
Aalto University

# Reading.

Ch. 6 of <https://mlbook.cs.aalto.fi>



This is a screenshot of the scikit-learn website. The top navigation bar includes the scikit-learn logo, 'Install', 'User Guide', 'API', 'Examples', 'Community', and a 'More' dropdown. Below the navigation bar, there are buttons for 'Prev', 'Up', and 'Next'. A pink box highlights 'scikit-learn 1.1.1' with a link to 'Other versions'. A yellow box contains the text 'Please cite us if you use the'. The main content area shows the section '3. Model selection and evaluation' in a light blue header, followed by the subsection '3.1. Cross-validation: evaluating estimator performance' in blue text. Below this, a list item '3.1.1. Computing cross-validated metrics' is visible.

[https://scikit-learn.org/stable/model\\_selection.html](https://scikit-learn.org/stable/model_selection.html)

“Model”  
=  
Hypothesis Space

# Learning Goals

- know train err is bad quality measure for ML method
- val.err. is more useful as quality measure for a ML model
- basic idea of k-fold CV
- hyper-parameter tuning = model selection
- Python implementations of k-fold CV / gridsearch

# ML – In a Nutshell

- learn hypothesis  $h(\cdot)$  out of **model** such that for any **data** point  $h(x) \approx y$
- approximation quality measured by **loss**  $L((x,y),h)$
- approximate “any data point” by a training set

# Model Validation

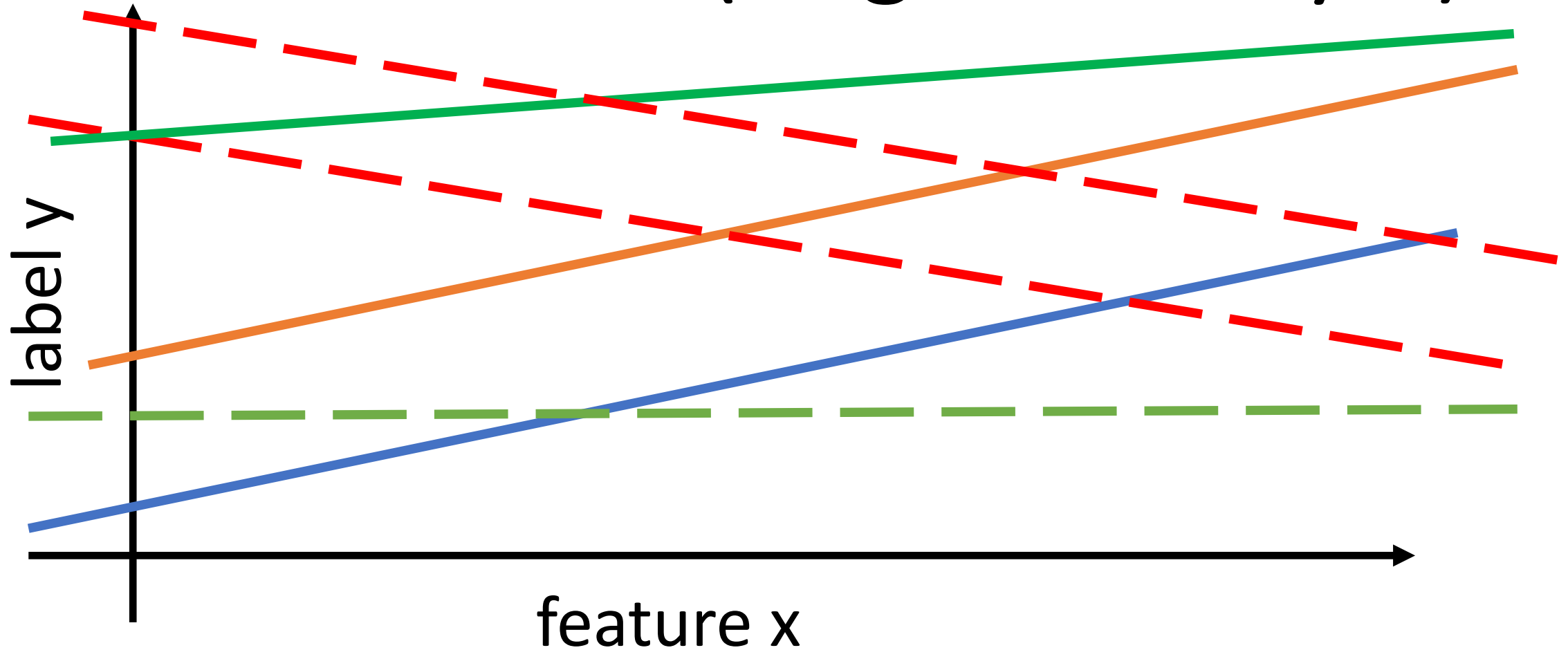
How do we know a model  
is any good ?

# Model Selection

How to choose between different alternative models?

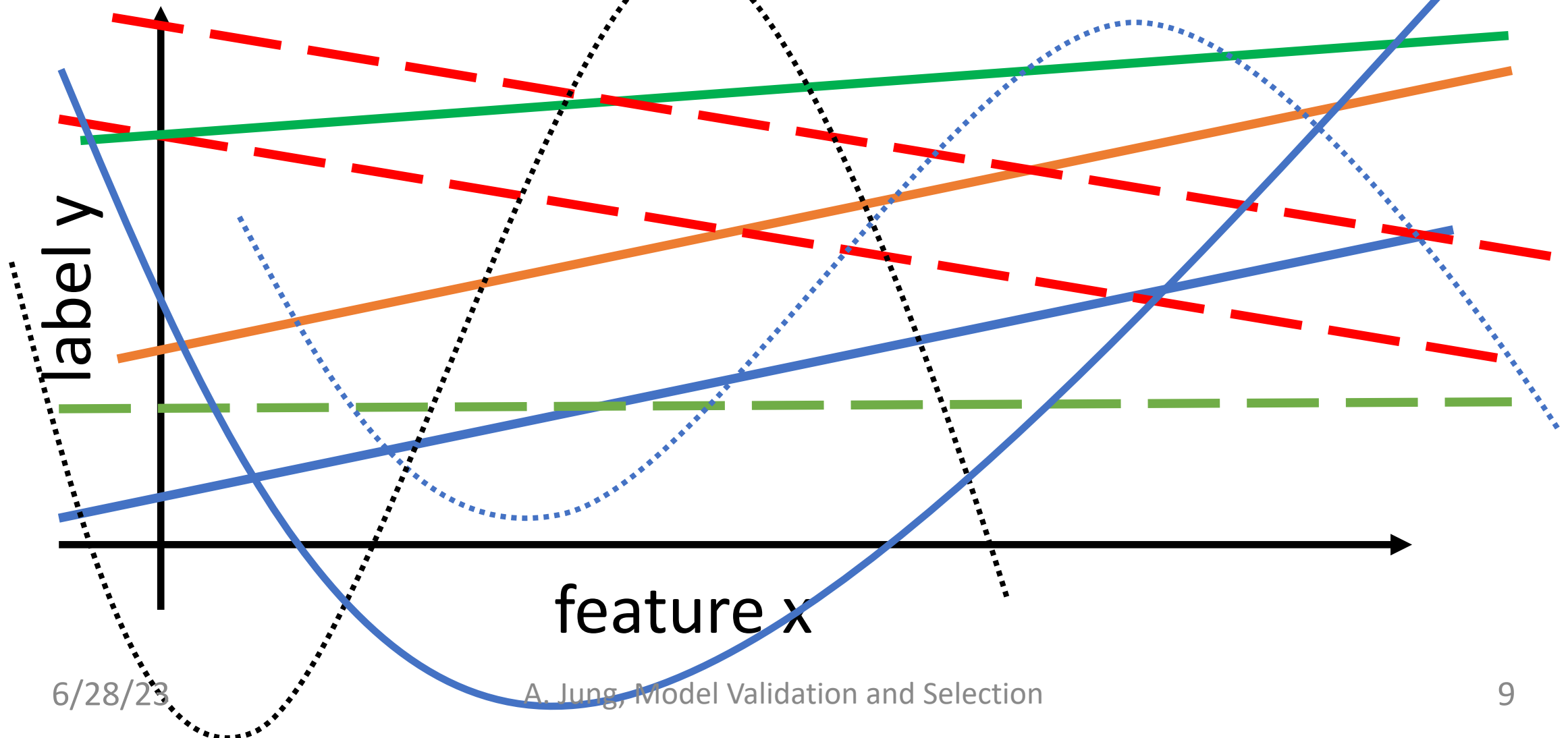
# Model 1:

## Linear Predictors (Degree 1 Polyn.)

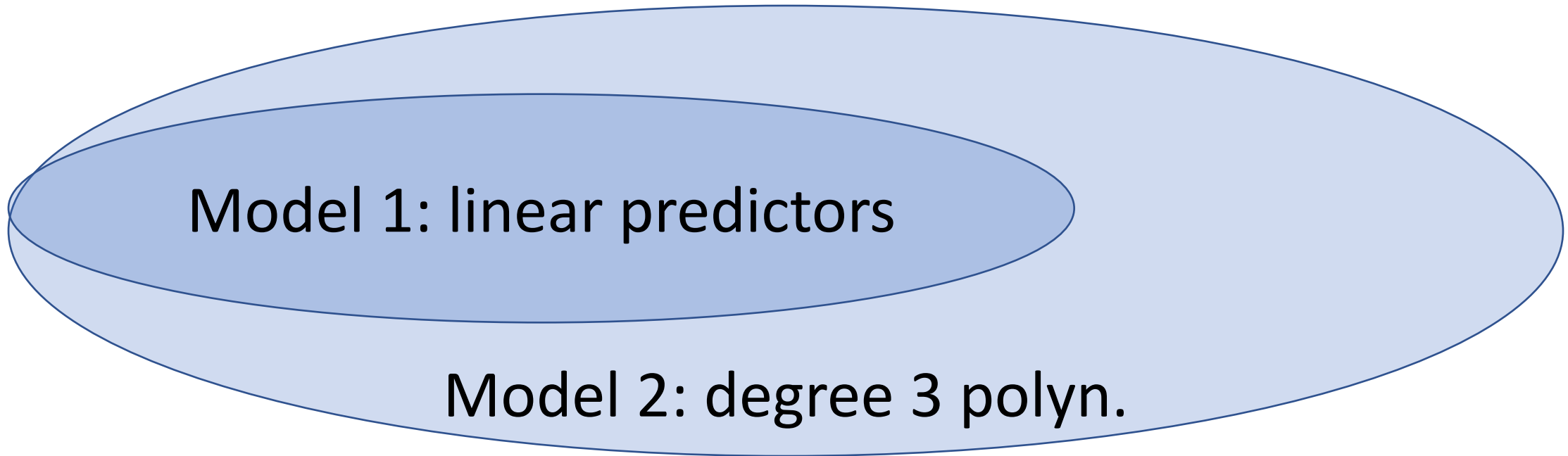




# Model 2: Degree 3 Polyn. Predictors



# Nested Models – I



# Math Notation

$$\mathcal{H}^{(n)} = \left\{ h(x) = \sum_{l=0}^n w_l x^l \text{ with some } w_l \right\}$$

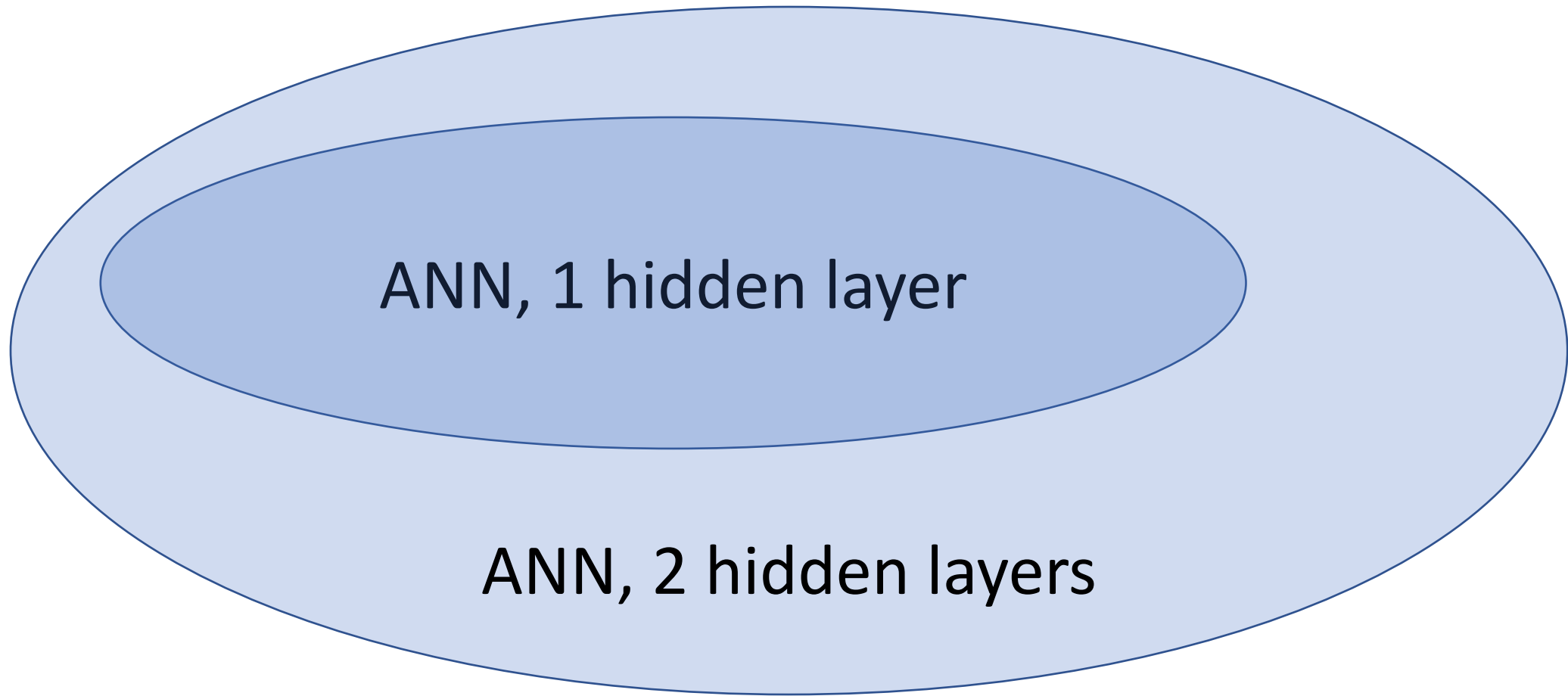
$\mathcal{H}^{(0)}$  ... constant prediction (ignores feature)

$\mathcal{H}^{(1)}$  ... linear hypotheses

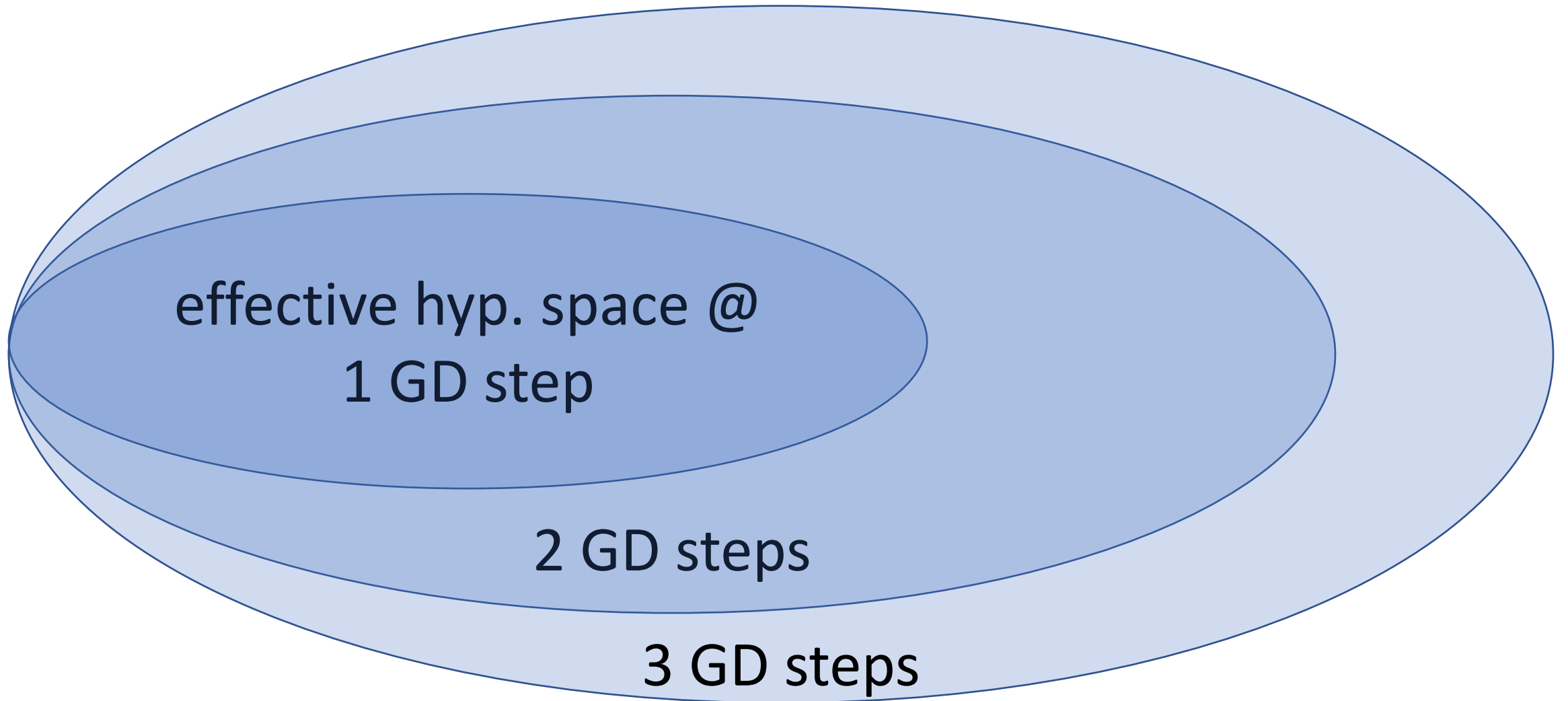
$\mathcal{H}^{(3)}$  ... degree 3 polyn.

$$\mathcal{H}^{(0)} \subseteq \mathcal{H}^{(1)} \subseteq \mathcal{H}^{(2)} \subseteq \mathcal{H}^{(3)} \subseteq \dots$$

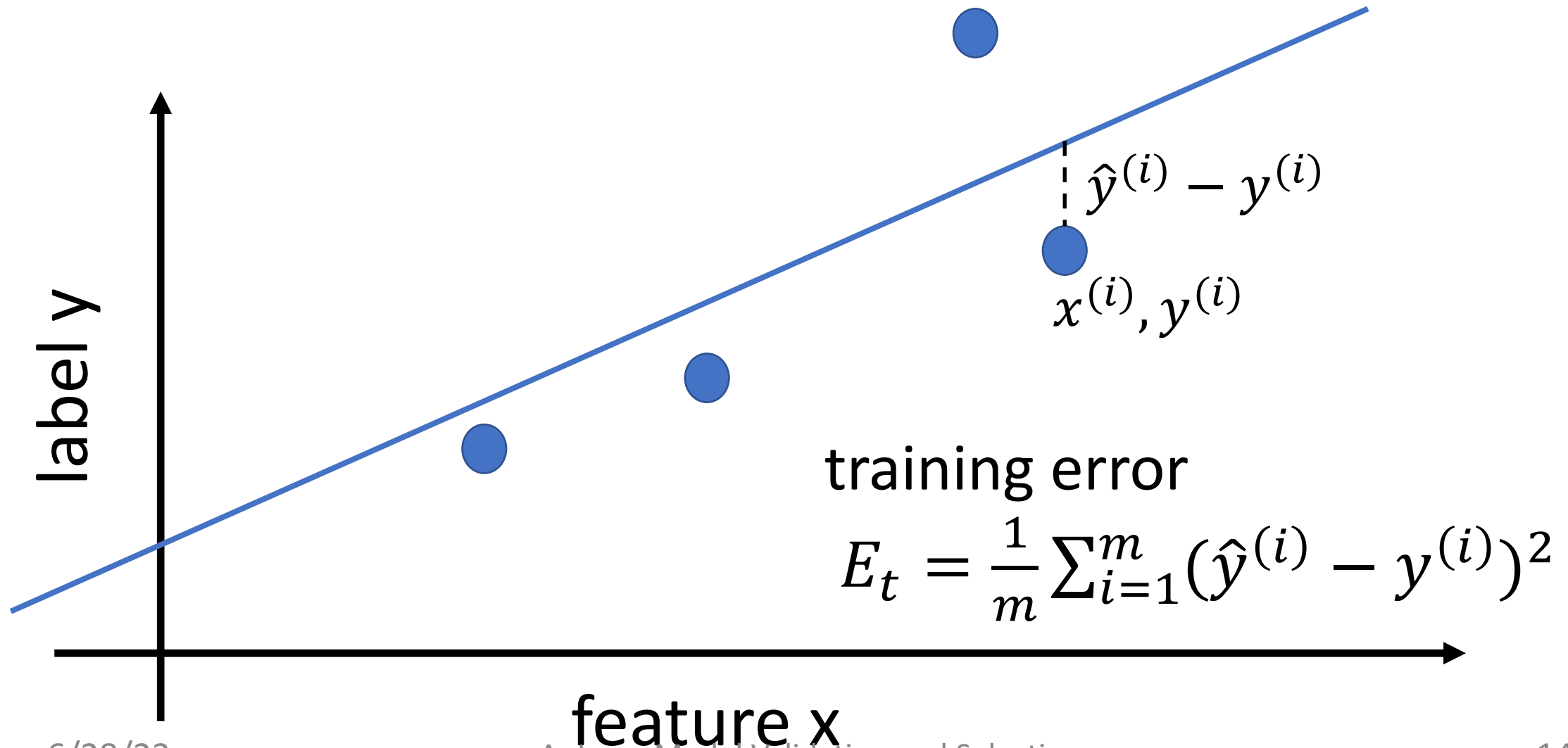
# Nested Models - II



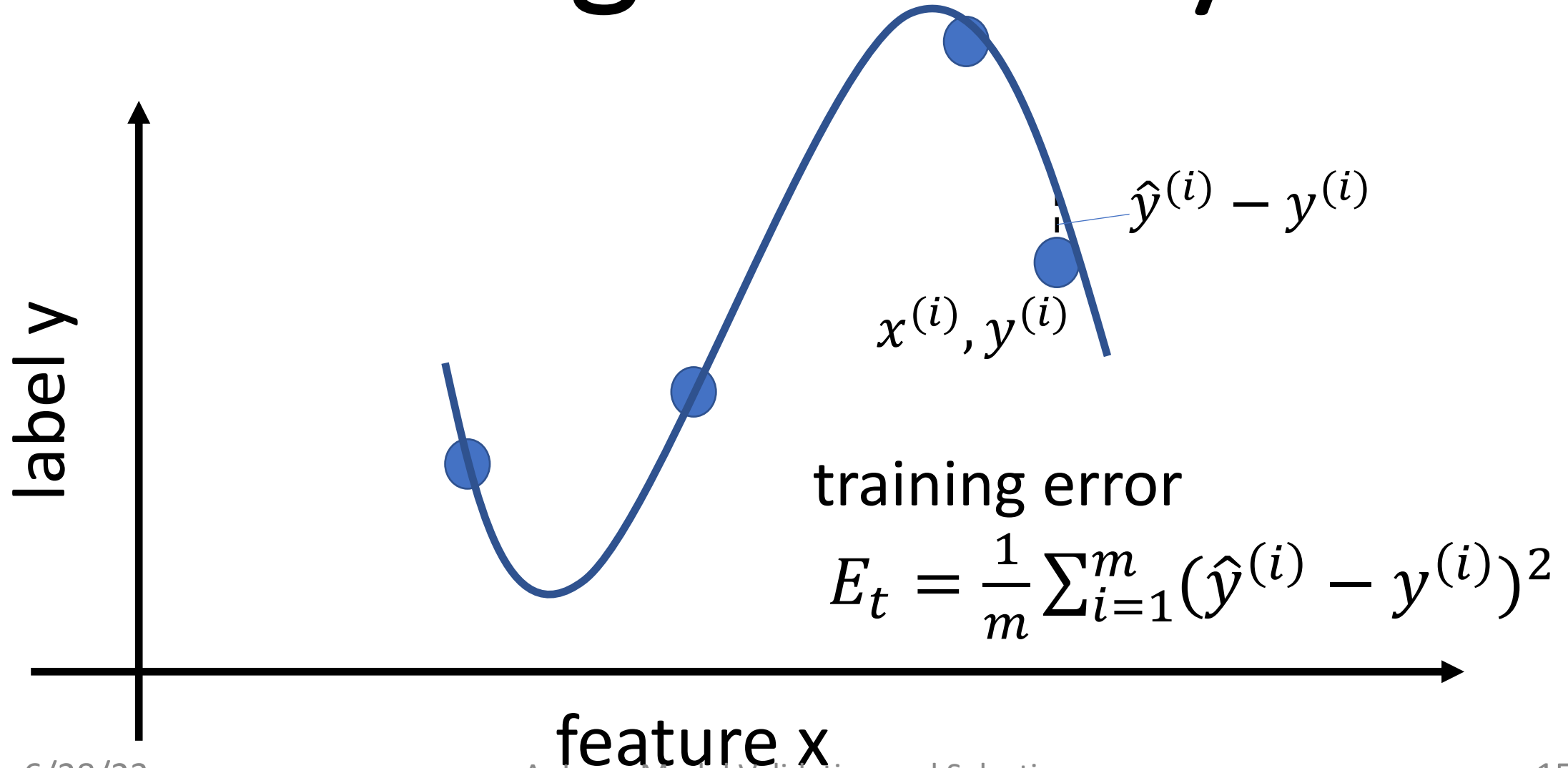
# Nested Models - III



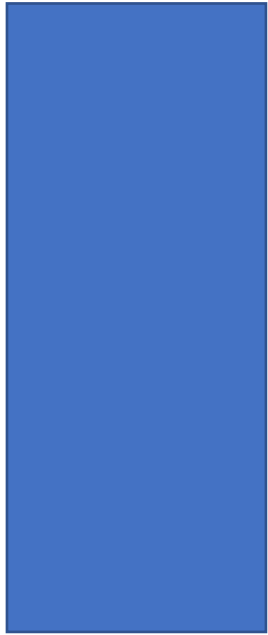
# Learn Linear Predictor



# Learn Degree 3 Polyn.



# Training Errors



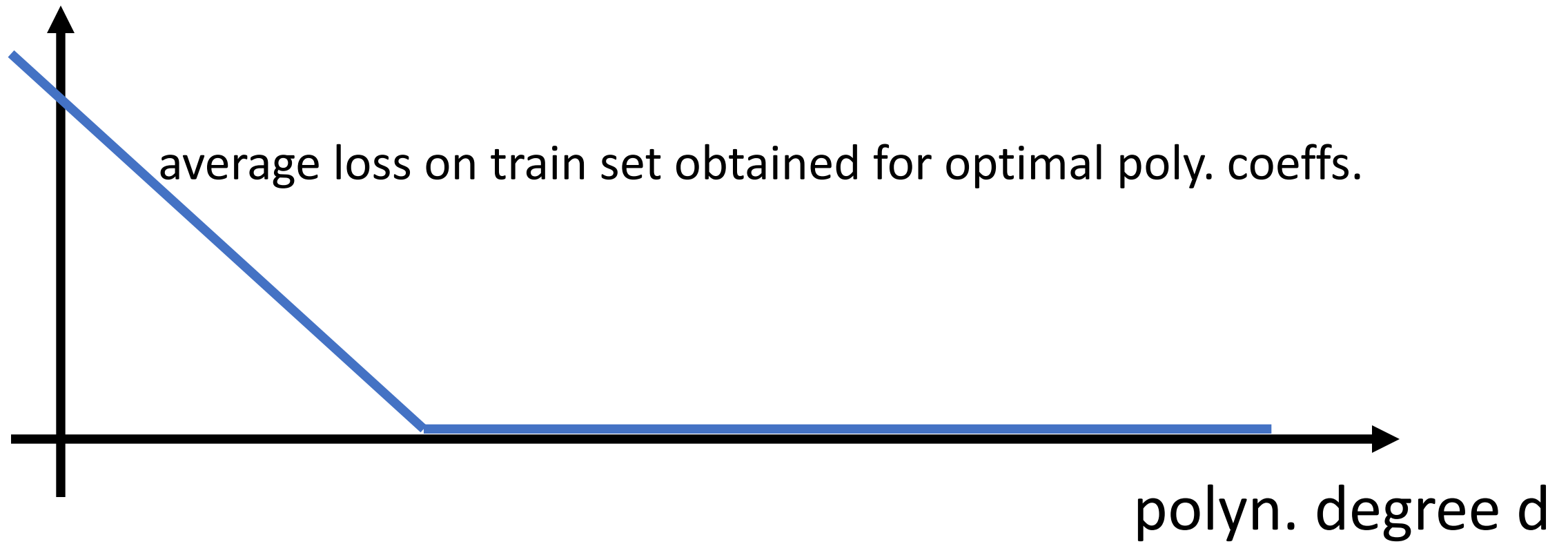
model 1  
linear predictors



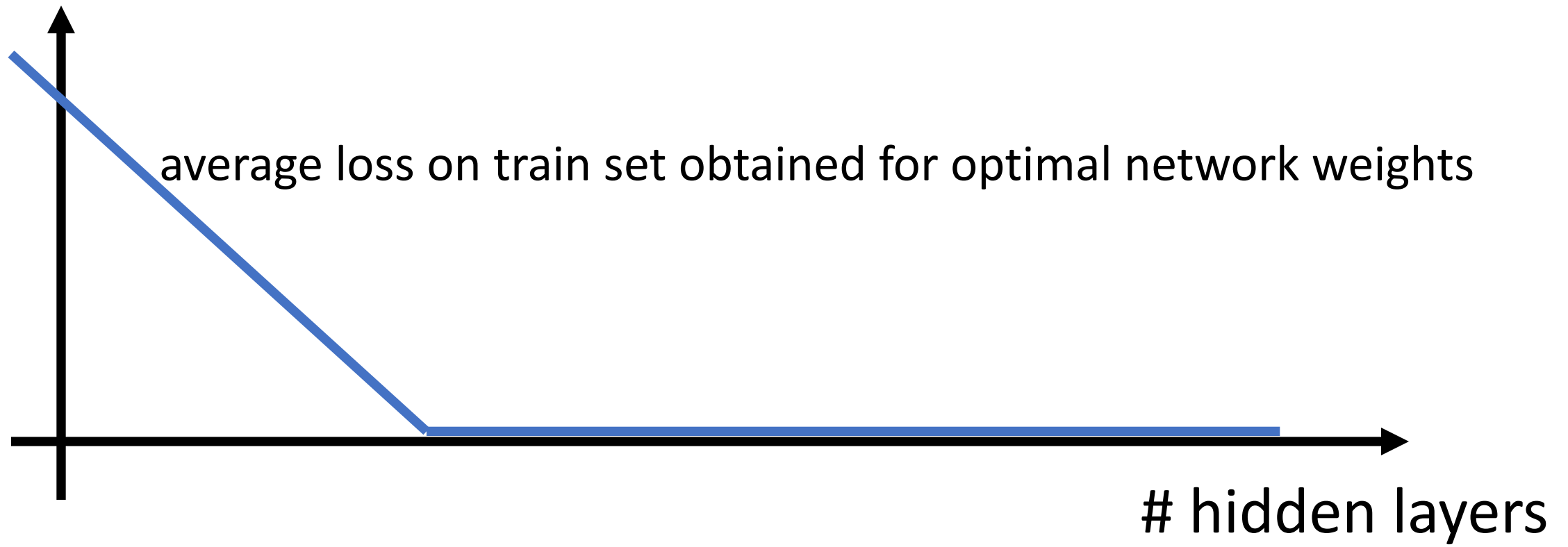
model 2:  
degree 3 polyn.



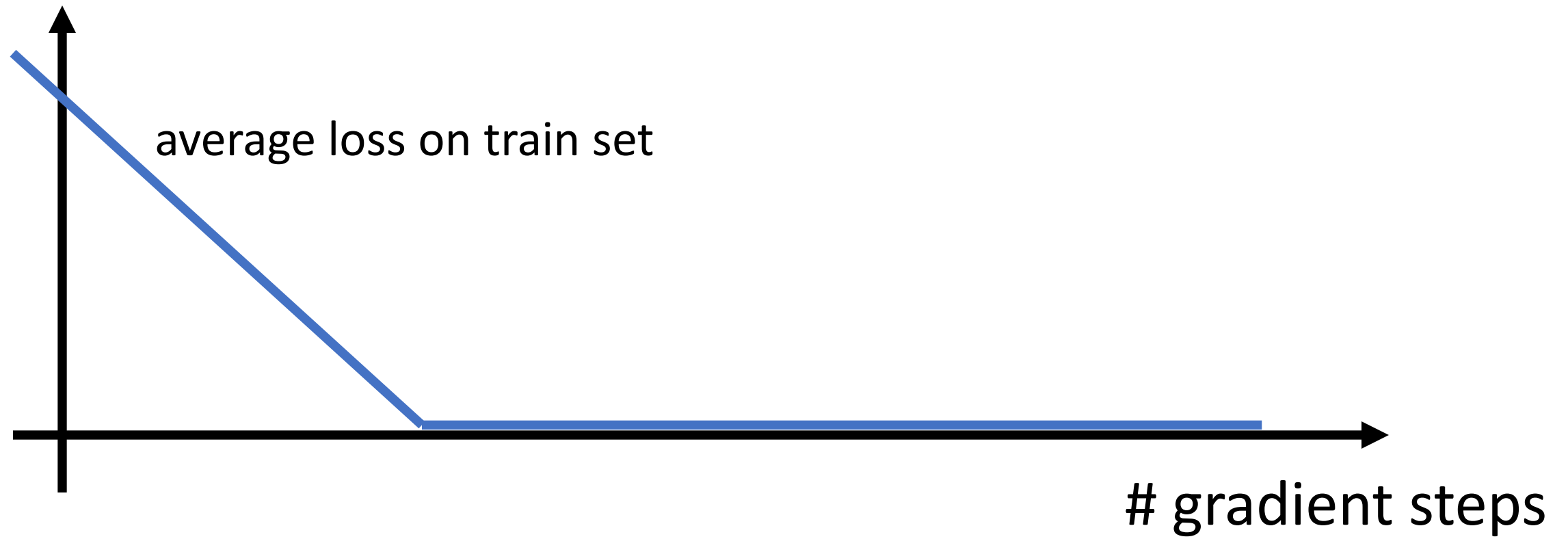
# Train Error vs. Degree



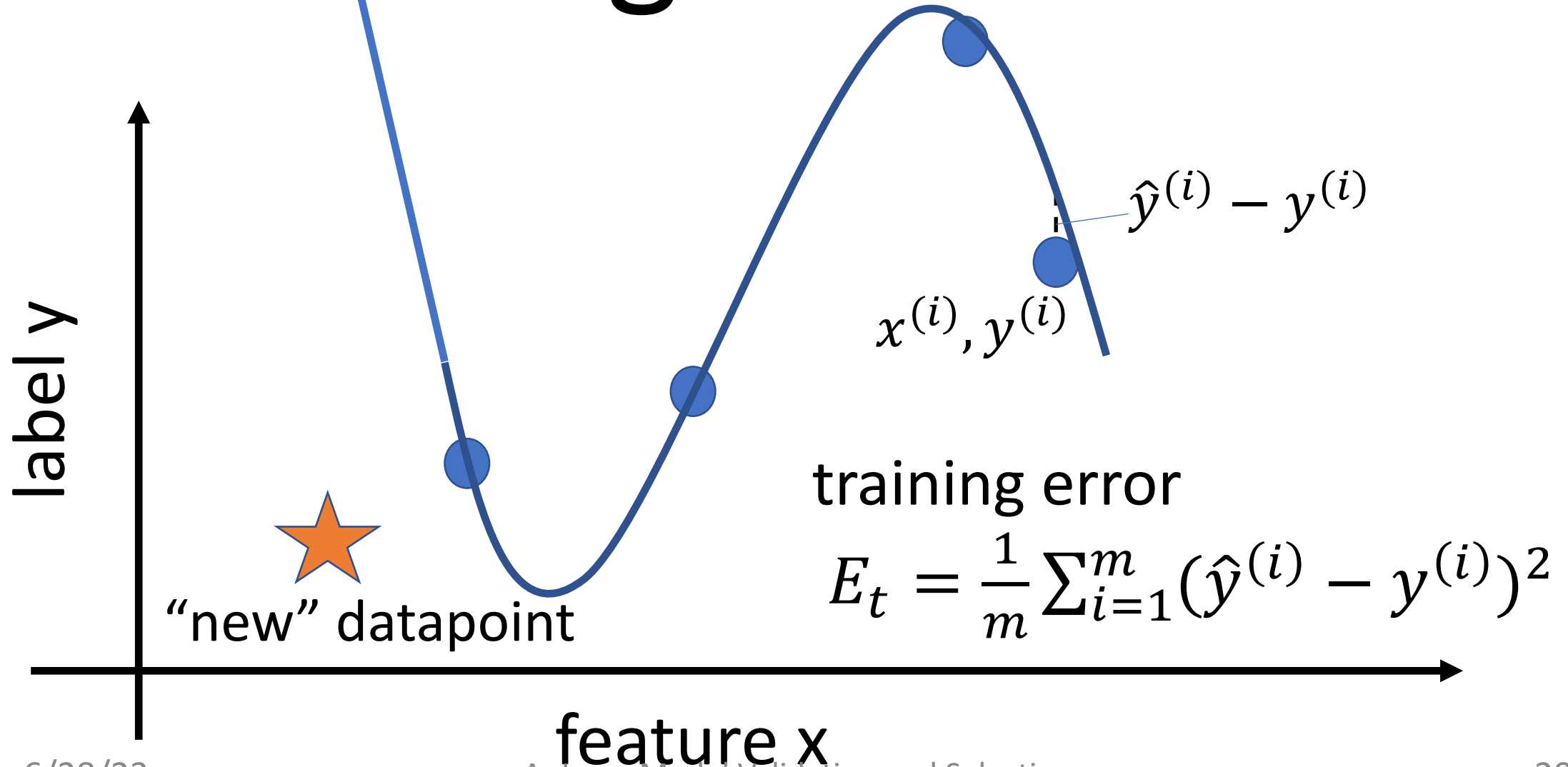
# Train Error vs. ANN Layers



# Train Error vs. Gradient Steps



# Overfitting



small training error does not imply good  
performance on new data points!

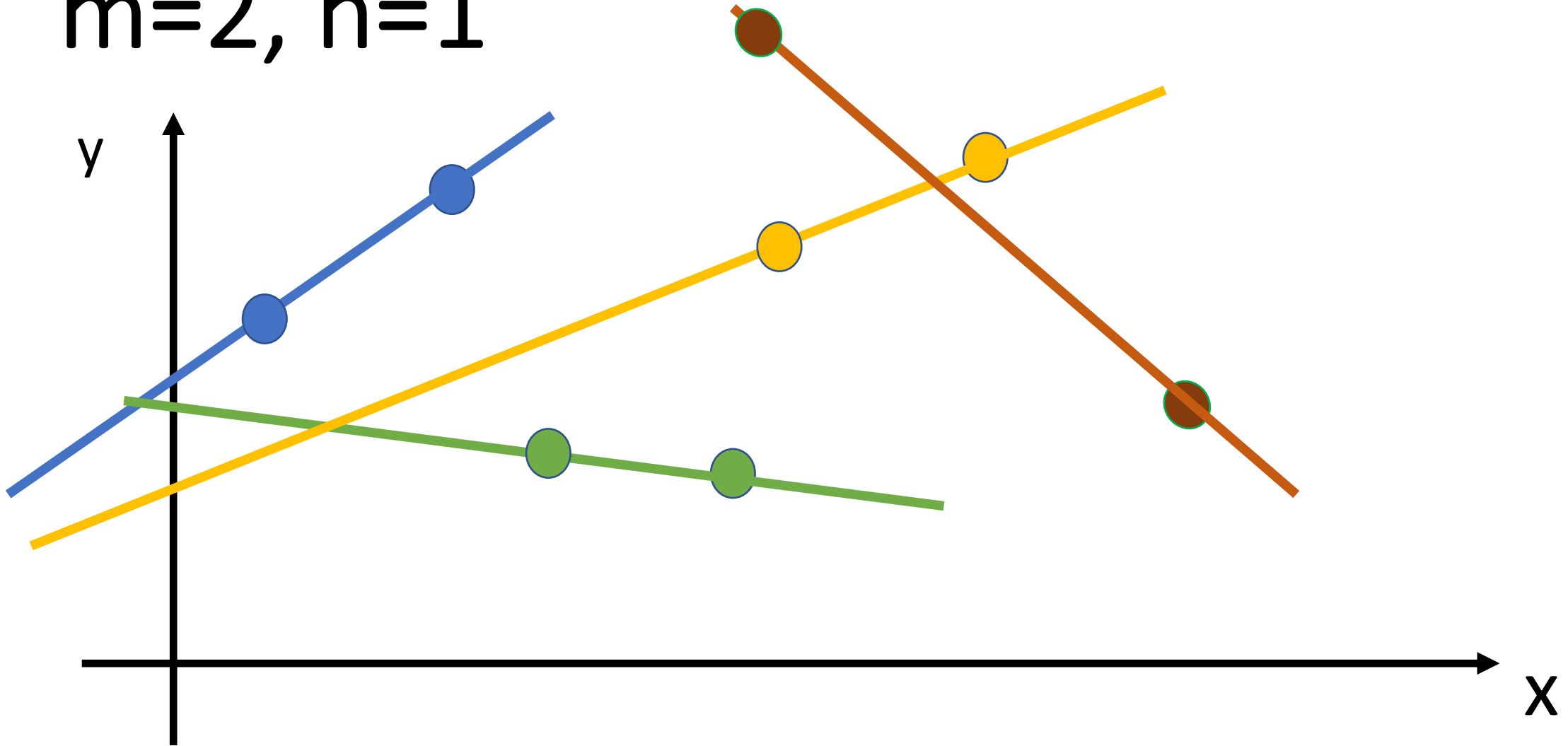
small training error merely indicates  
that training algorithm has been  
implemented correctly

# A Case in Point

we can perfectly fit (almost) any  $m$  data points using polynomials of degree  $n$  as soon as

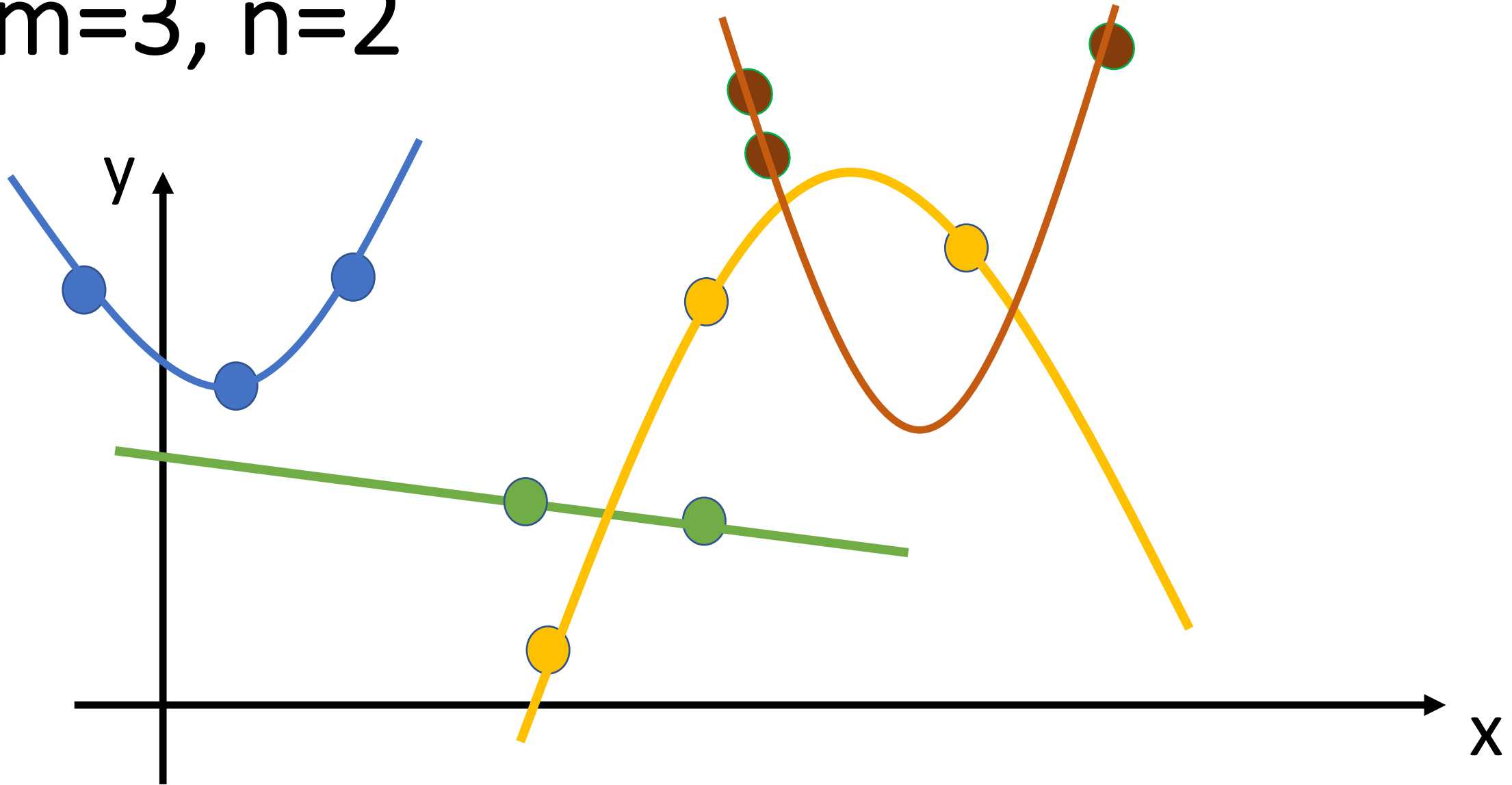
$$n \geq m-1$$

$m=2, n=1$





$m=3, n=2$



# Reminder: Probabilistic Model

- data points are realizations of RVs
- joint pdf  $p(x,y)$  of features and label
- training set is a RV
- learnt hypothesis  $h(.)$  is a RV
- prediction  $h(x)$  is a RV

# Why Can Train. Err. Mislead?

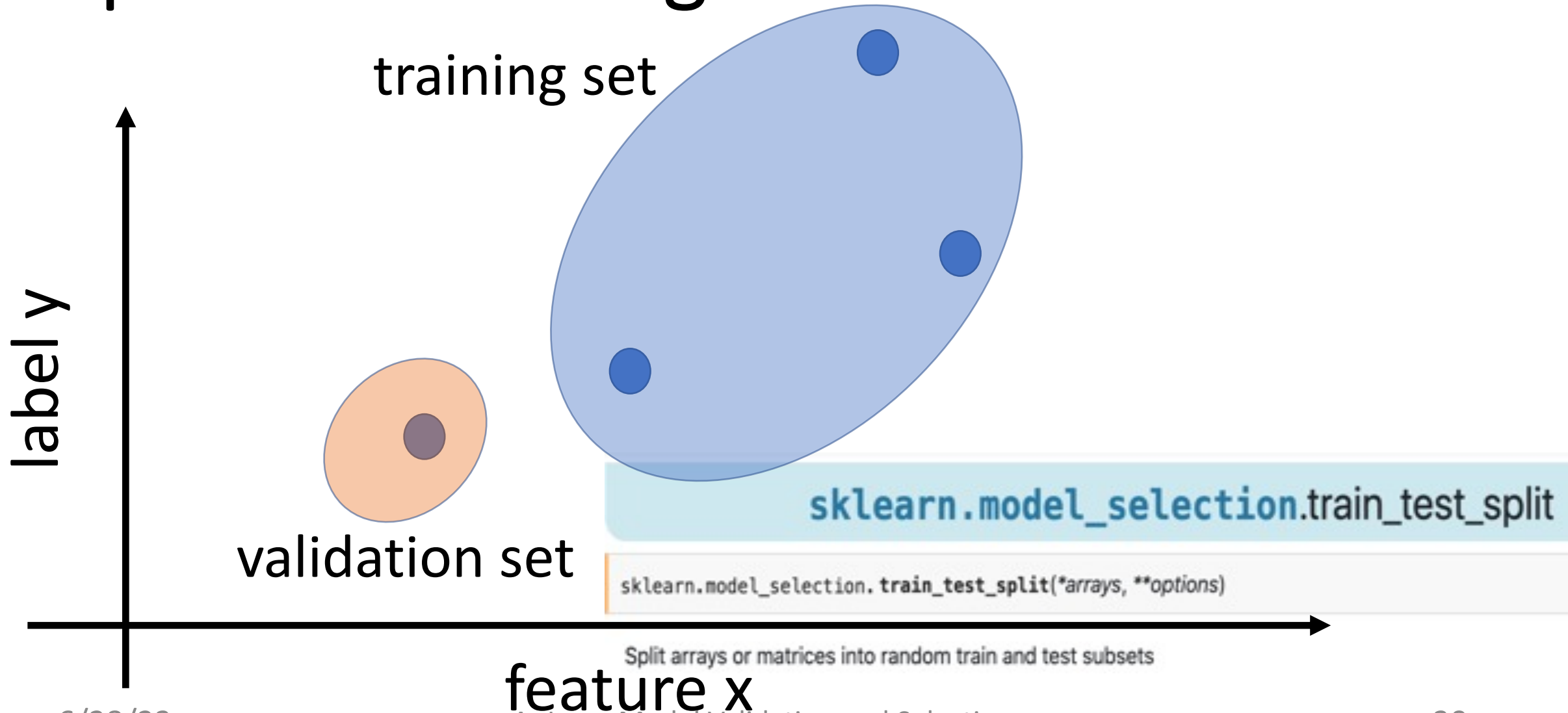
- consider expected loss of hypothesis
- estimate expectation using sample average
- this only works if hypothesis does not depends on data points used in average
- does not hold for training error

# Model Validation

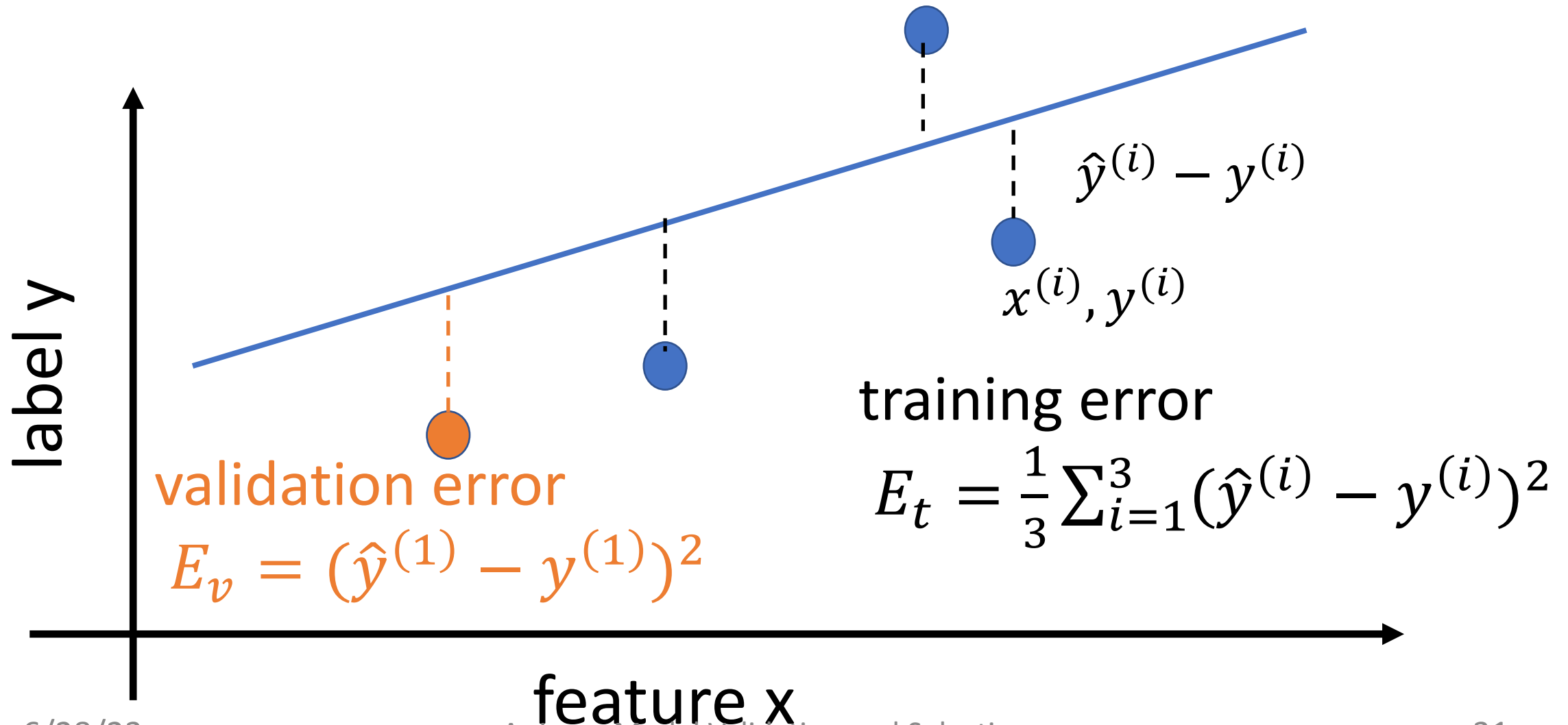
# Basic Idea of Validation

- divide data points into two subsets
- use **training set** to learn predictor
- use **validation set** to estimate loss

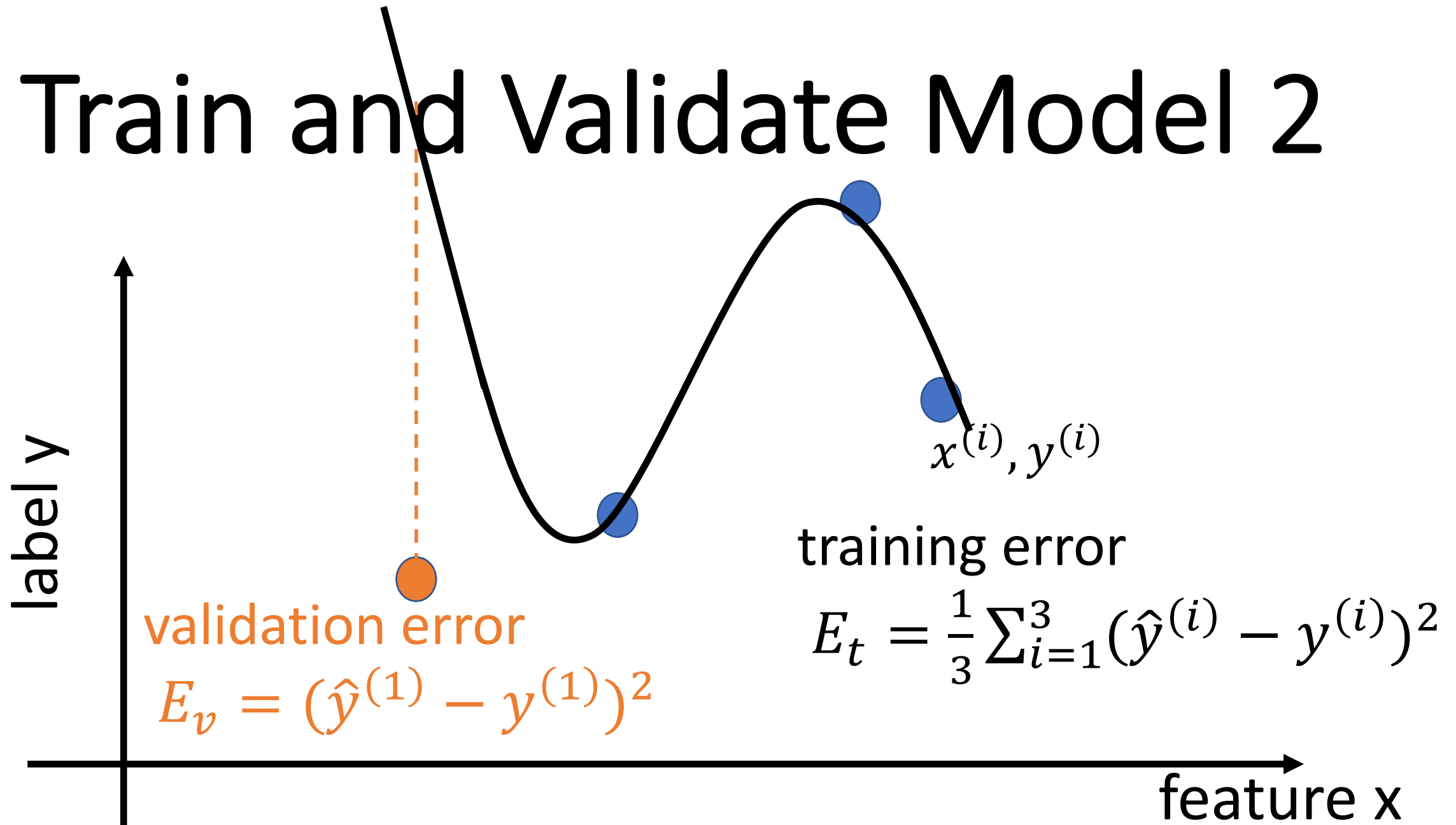
# Split into Training and Validation Set



# Train and Validate Model 1



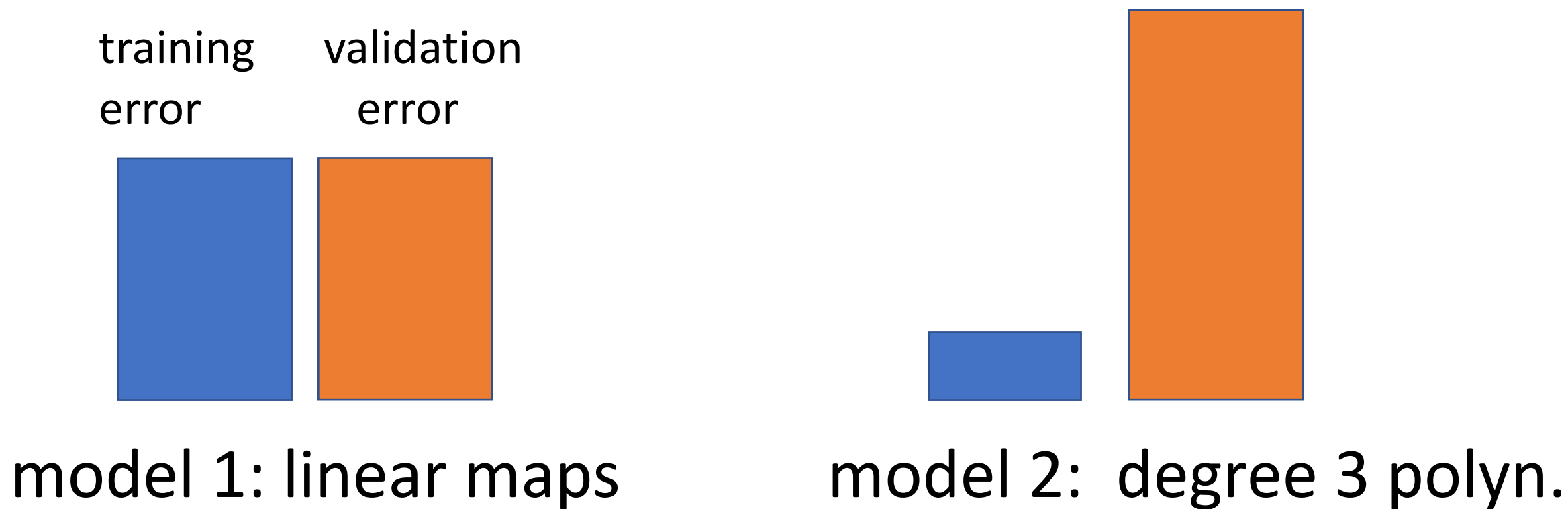
# Train and Validate Model 2





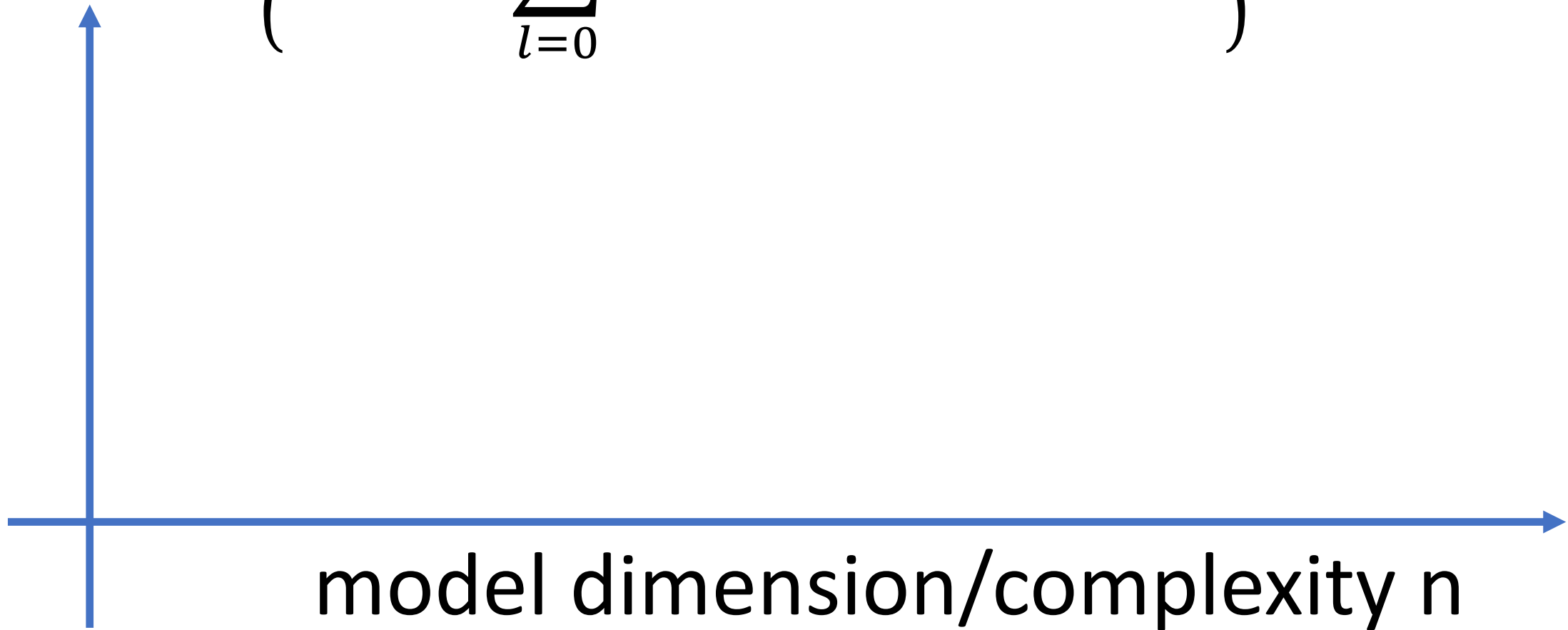
# Basic Idea of Model Selection

choose model via validation error

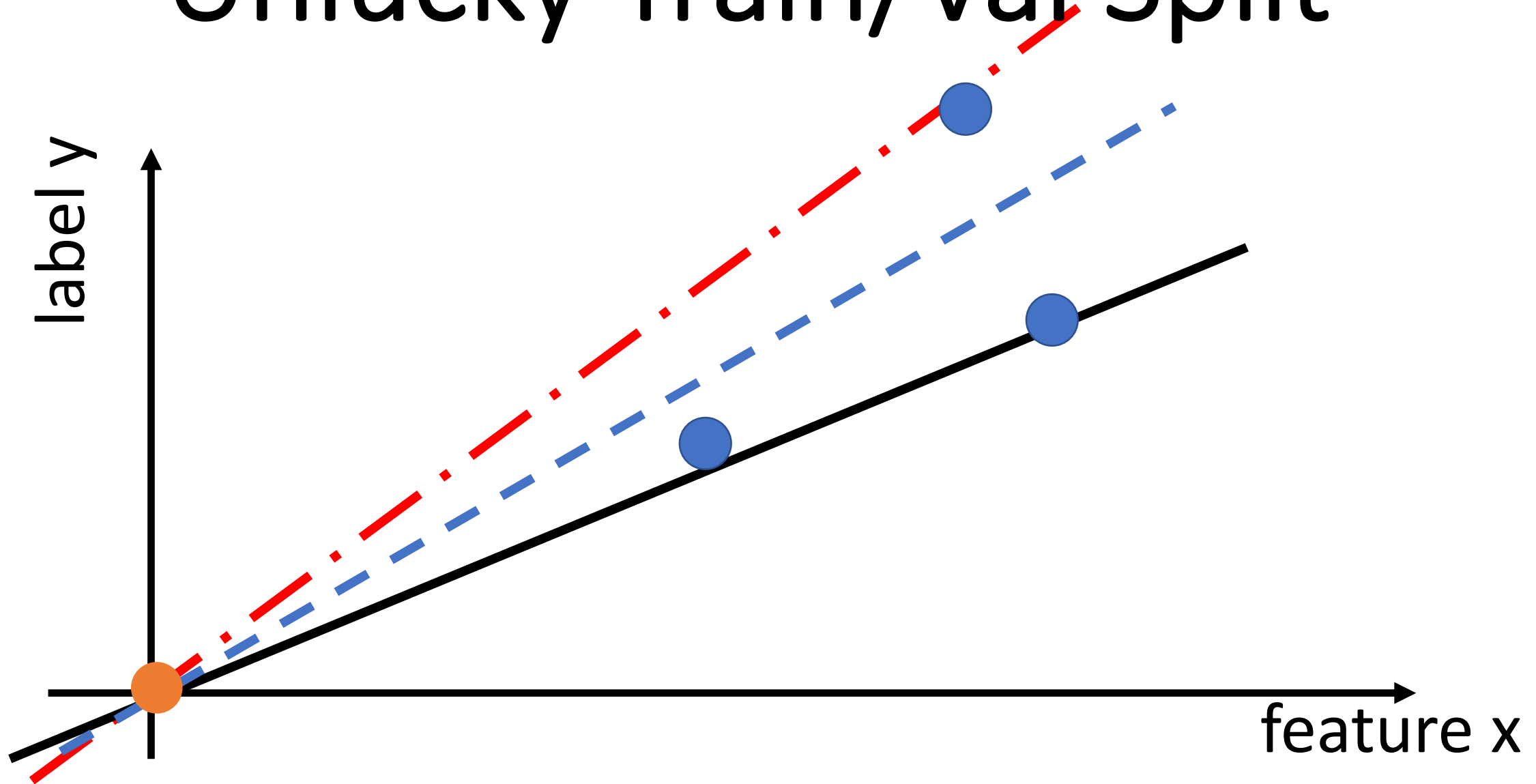


# Train/Val Error vs Model Complexity

$$\mathcal{H}^{(n)} = \left\{ h(x) = \sum_{l=0}^{n-1} w_l x^l \text{ with weights } w_l \right\}$$



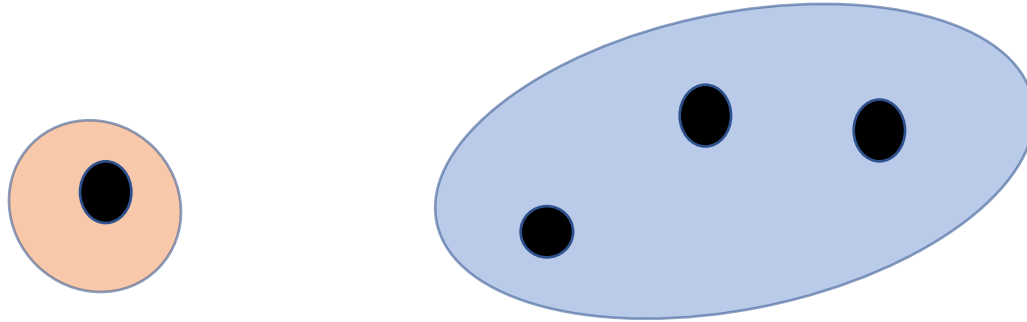
# Unlucky Train/Val Split



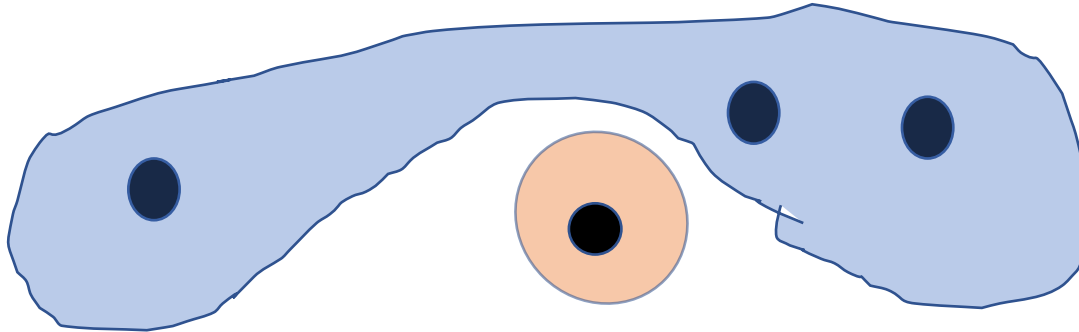
# k-Fold Cross Validation

- might be unlucky with train/val split
- problematic for small datasets
- IDEA: randomly split several times
- “average out” unlucky splits

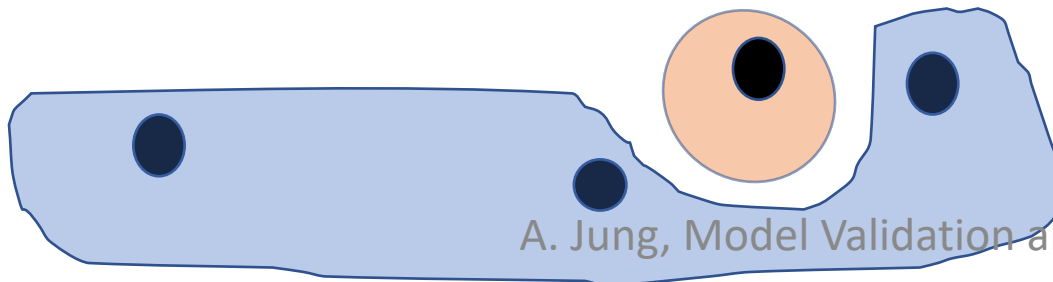
# K-Fold Cross Validation



fold 1



fold 2



fold 3

# k-Fold Cross Validation

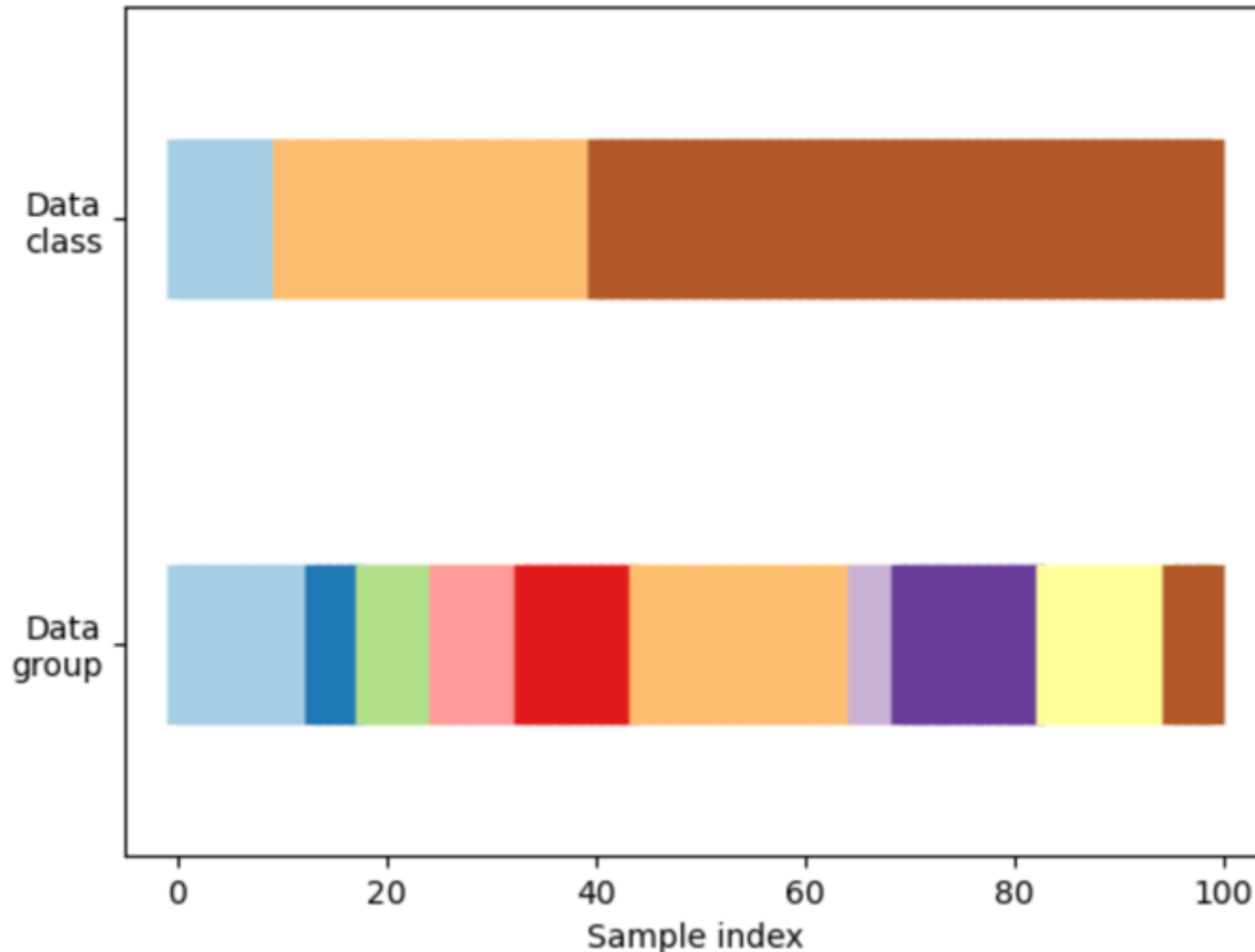
how to choose nr of folds (the “k” in k-fold CV) ?

- train fold should be sufficiently large (avoid overfitting)
- val folds should be sufficiently large (to get reliable estimate of generalization)

# CAUTION!

- k-fold CV requires a method to split into folds
- most basic method: evenly divide into k folds
- works if data is i.i.d. (“order of data points is arbitrary”)
- fails if data points are grouped or ordered

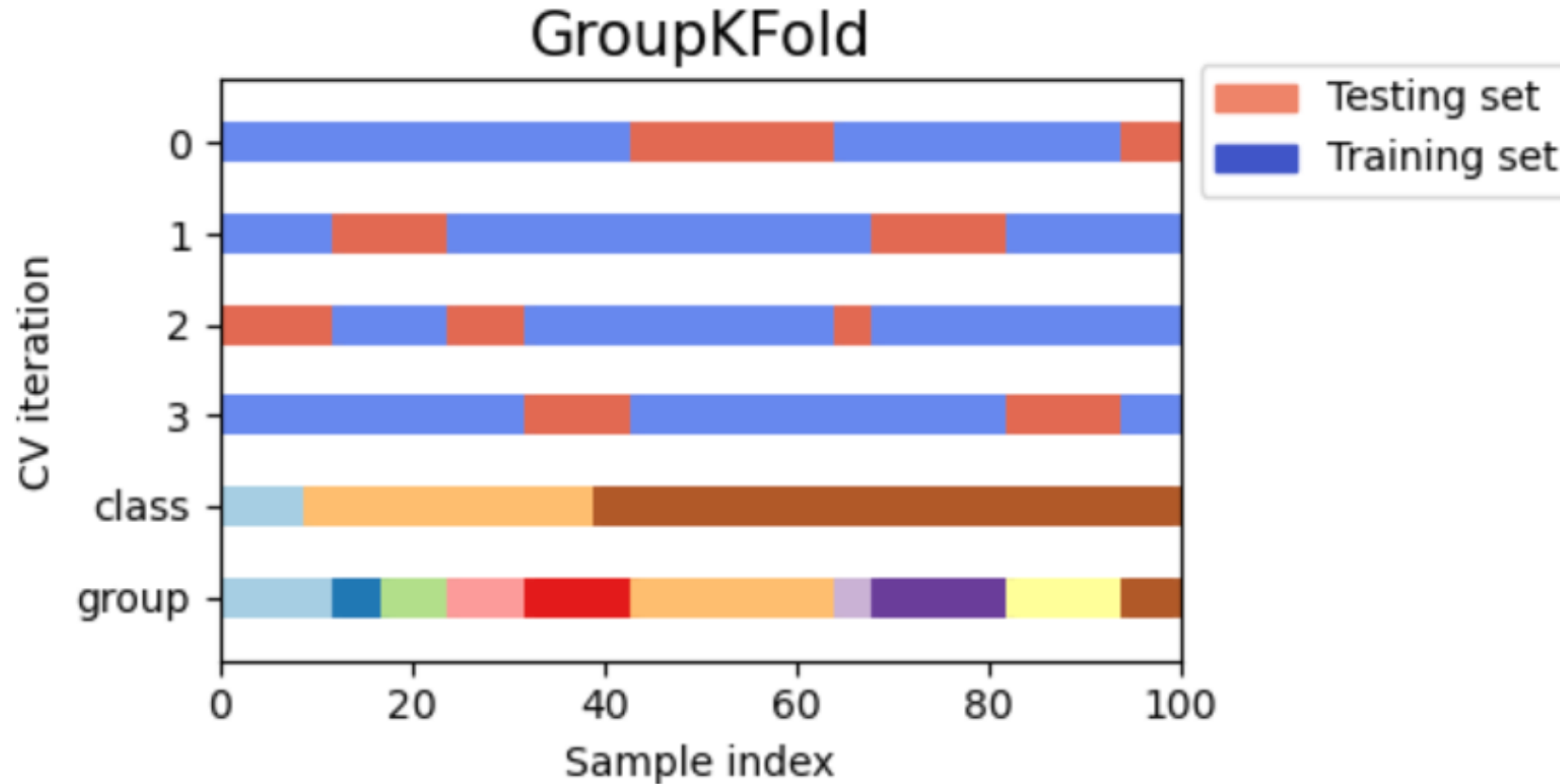
# Imbalanced Classes and Group Structure



- e.g. data points with same label are contiguous blocks
- or data points are obtained at consecutive time instants ( $\rightarrow$  correlations)

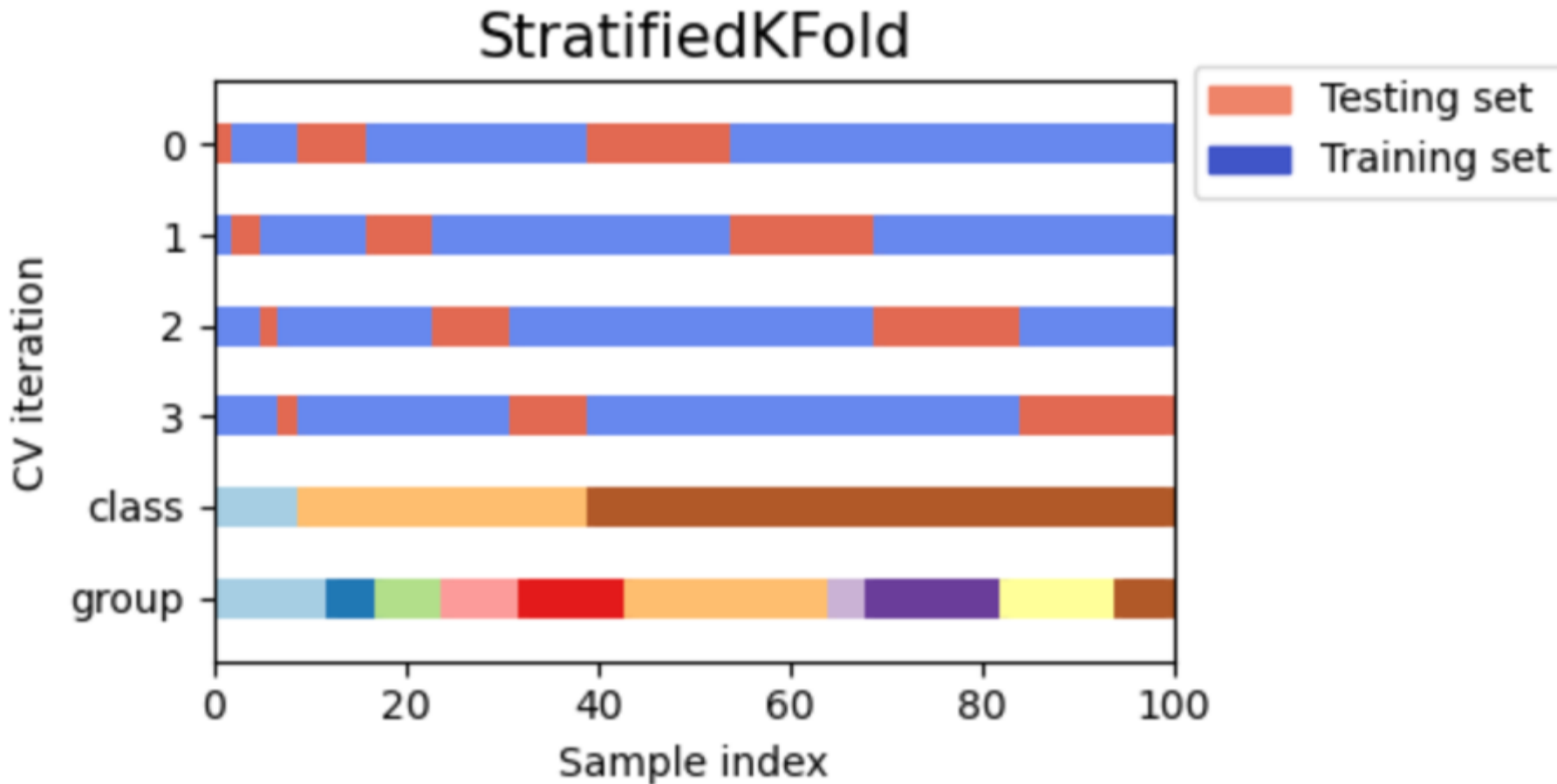


# Group-Preserving Splitting

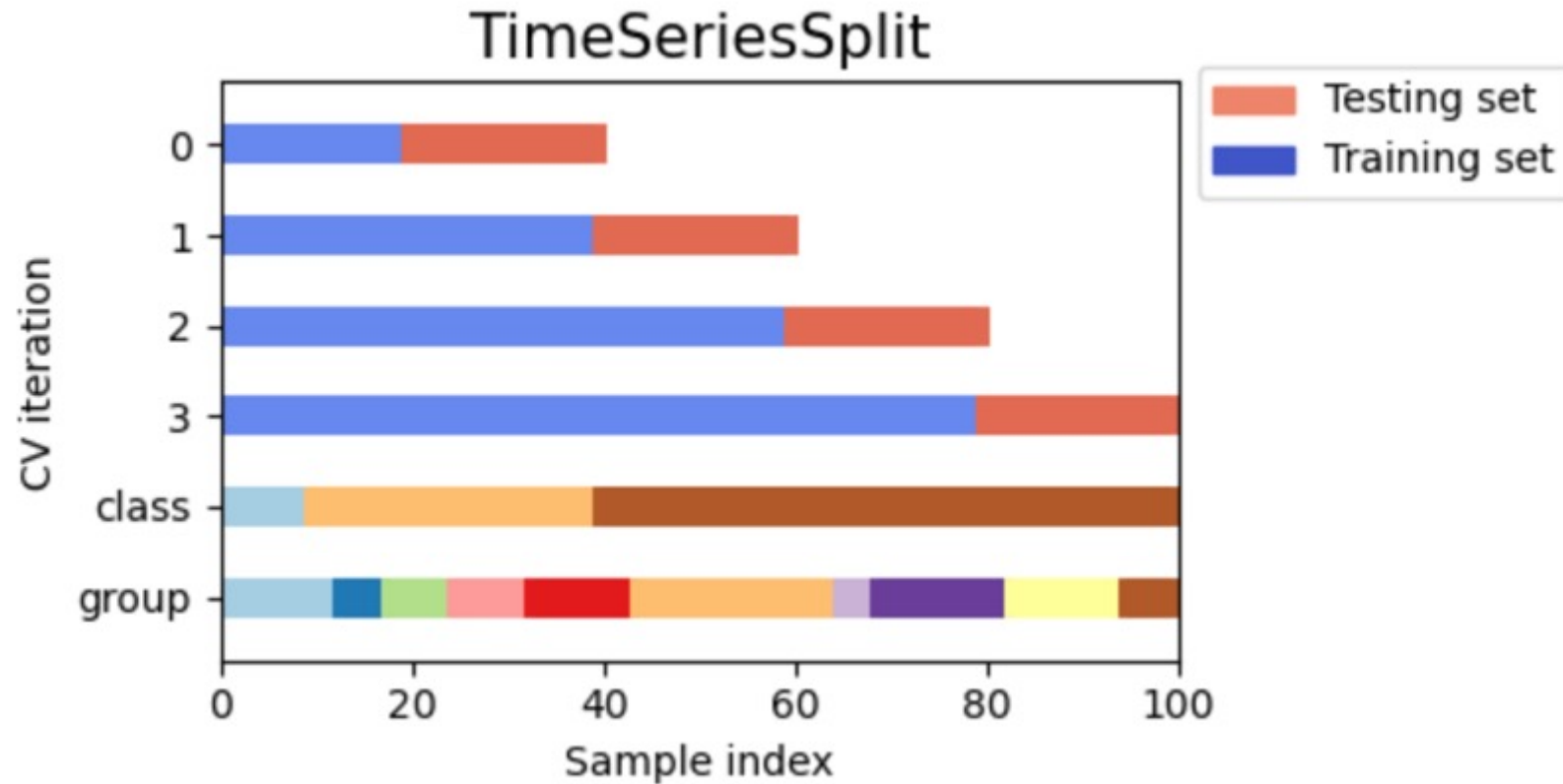


[https://scikit-learn.org/stable/modules/generated/sklearn.model\\_selection.GroupKFold.html](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GroupKFold.html)

# Class-Ratio Preserving Splitting



# Temporal Successive Splitting



source: <https://scikit-learn.org/stable/>

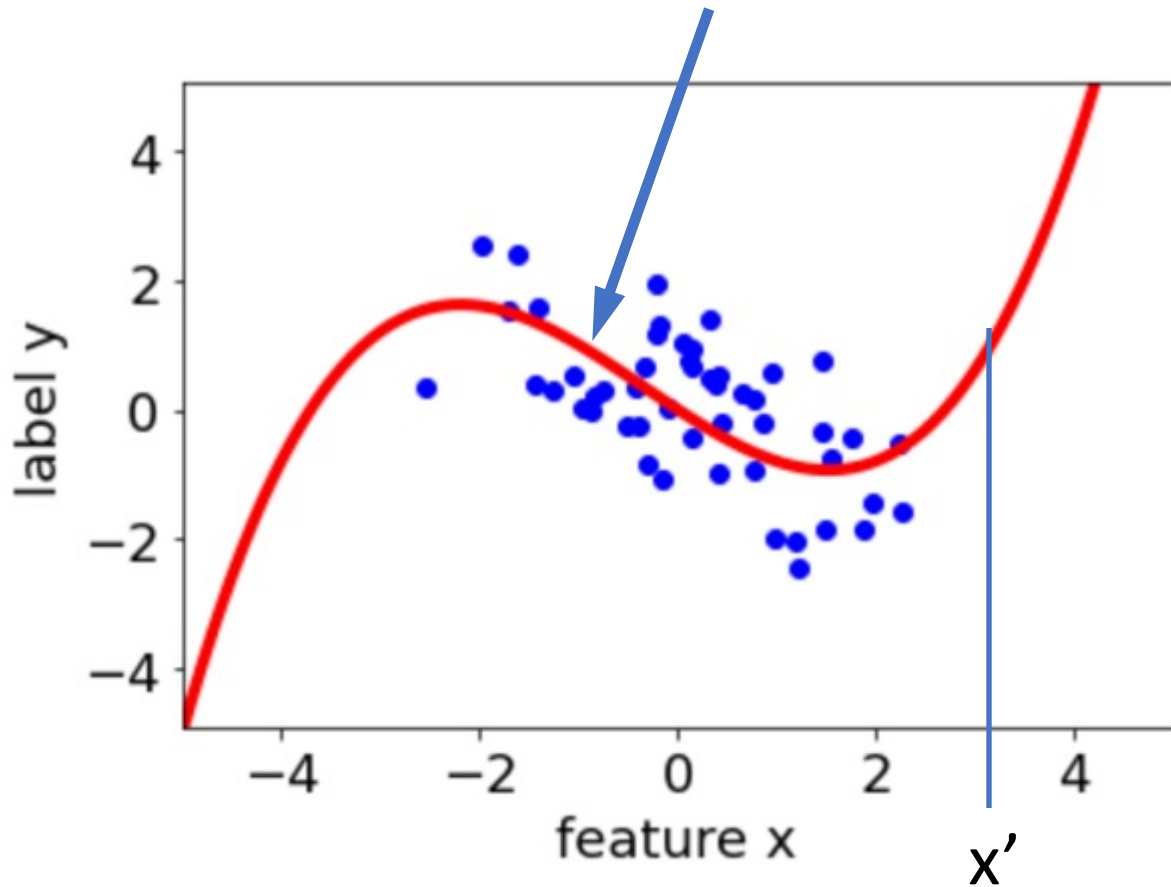
# Bias and Variance Decomposition

“Bias” reflects error due to  
model being too small

“Variance” reflects error due to dataset being too small

# Toy Data

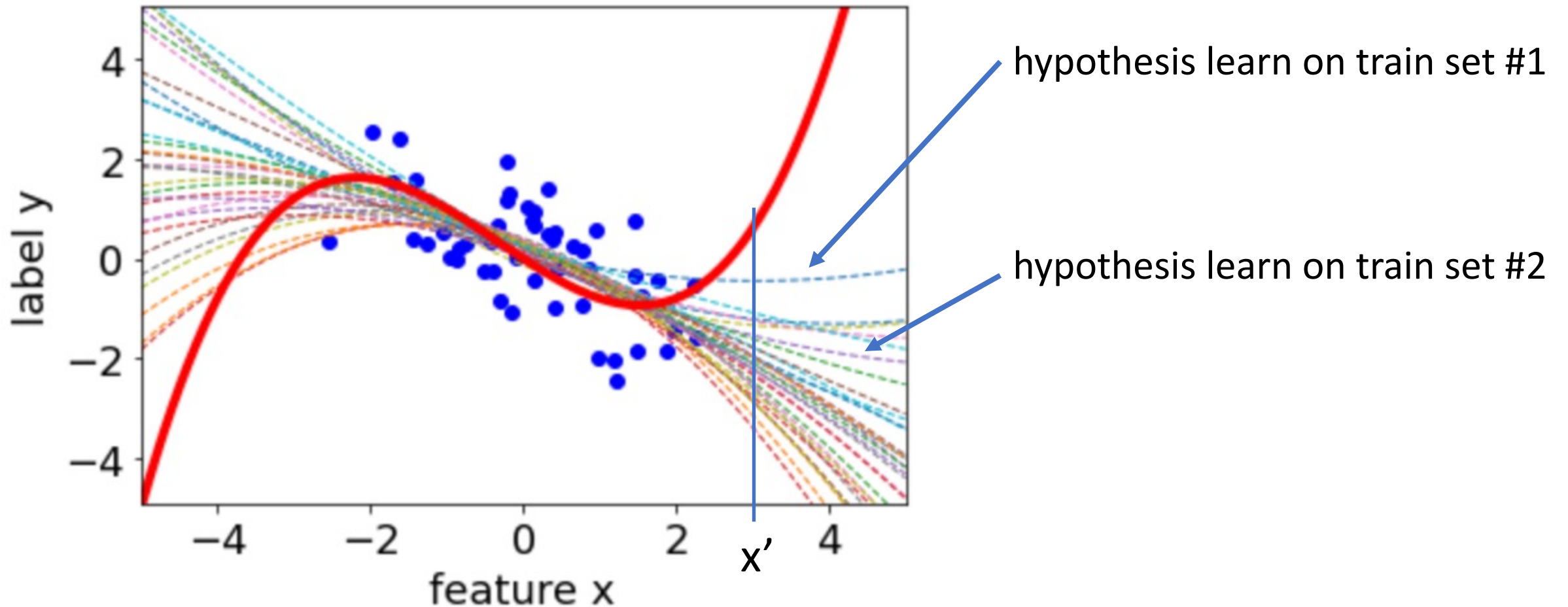
$$y = g(x) + \text{"noise"}$$



learn hypothesis  $h(.)$  using a randomly selected training set

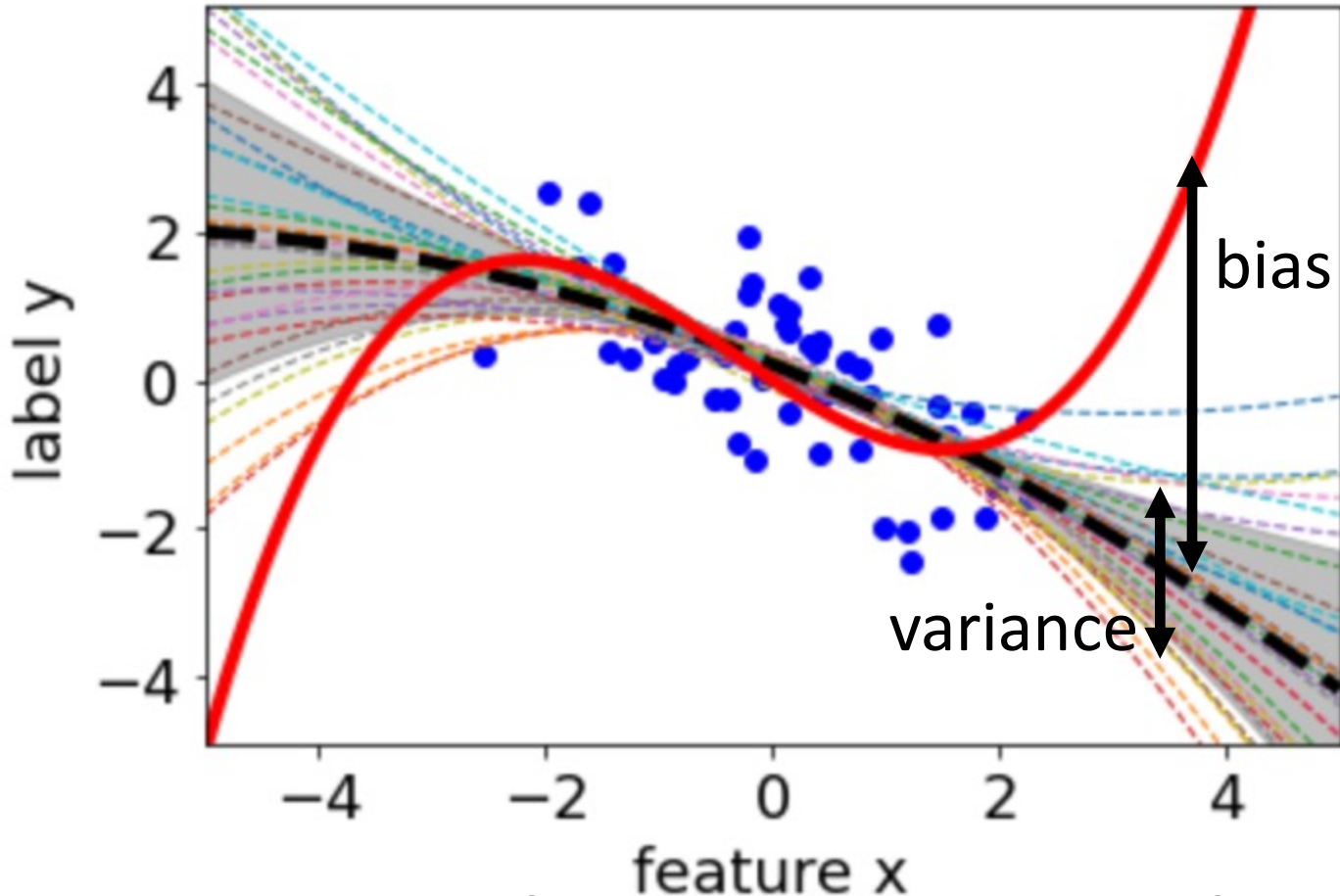
compute prediction  $h(x')$  for a fixed feature value  $x'$

# Ensemble of Learnt Hypotheses





# Bias and Variance

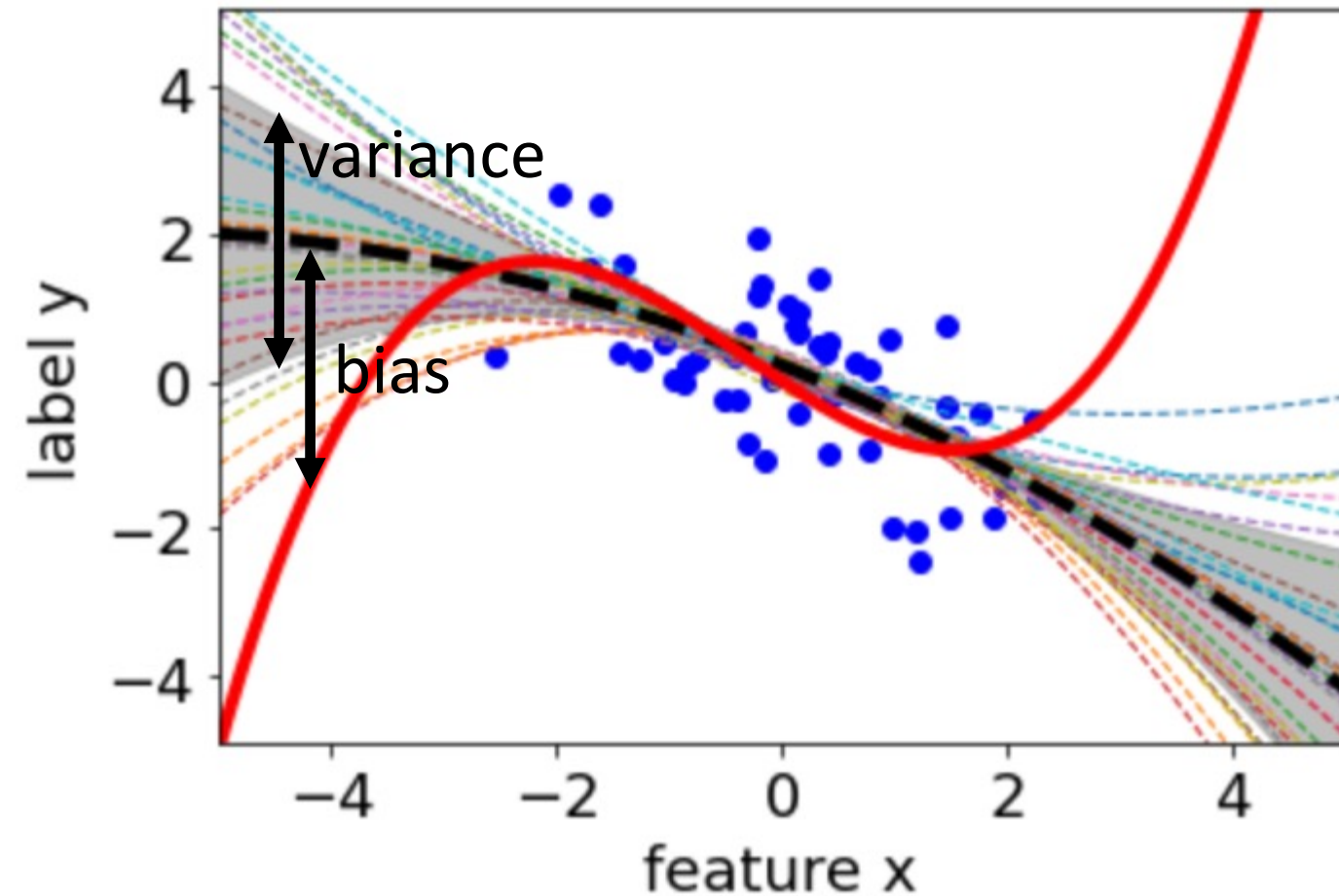


$$\hat{y} = h(x')$$

RV since obtained  
from a randomly  
selected training set

$$\mathbb{E}\{(\hat{y} - y)^2\} = (\mathbb{E}\{\hat{y}\} - y)^2 + \mathbb{E}\{(\hat{y} - \mathbb{E}\{\hat{y}\})^2\}$$

# Bias and Variance Tradeoff

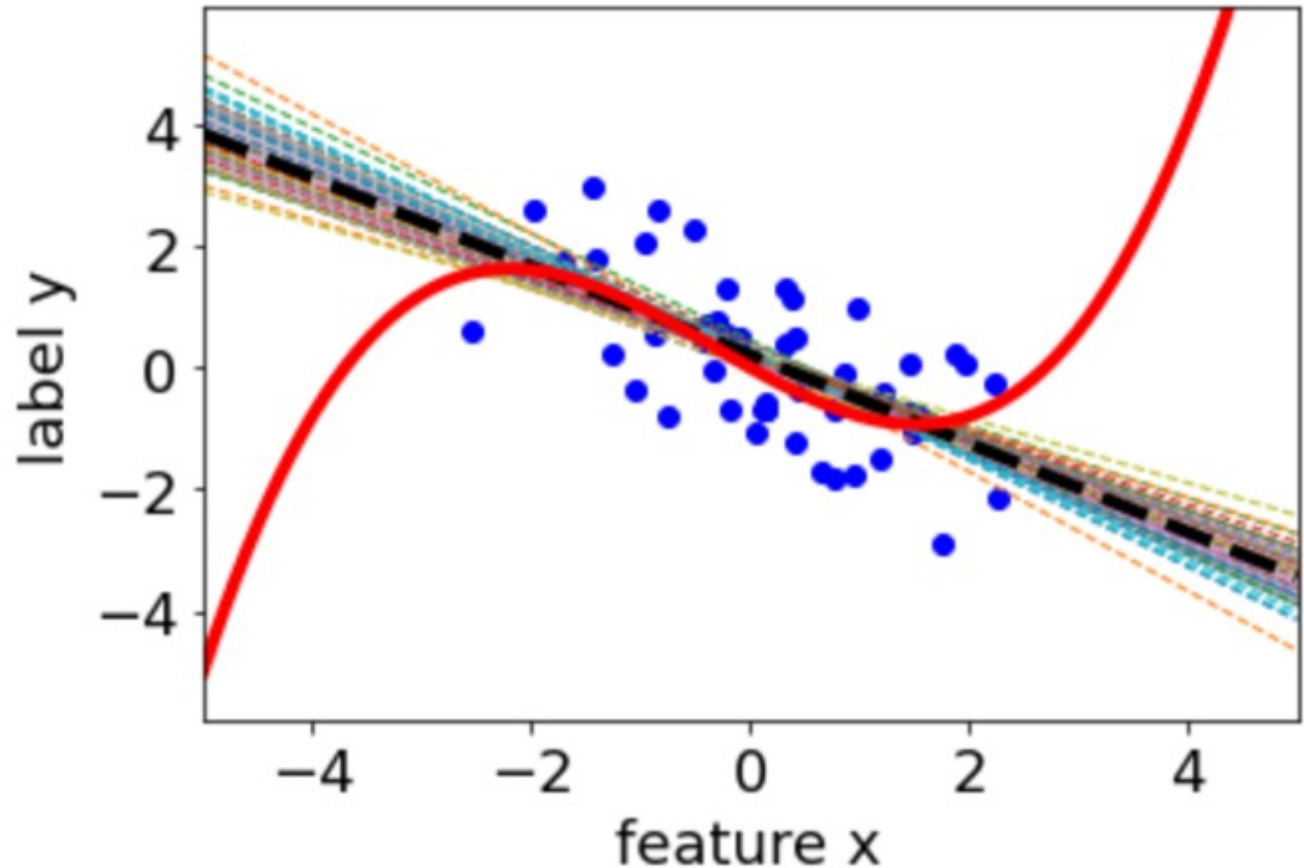


"Prediction Error = Bias + Variance"

bias reduction typically incurs variance increase and vice versa

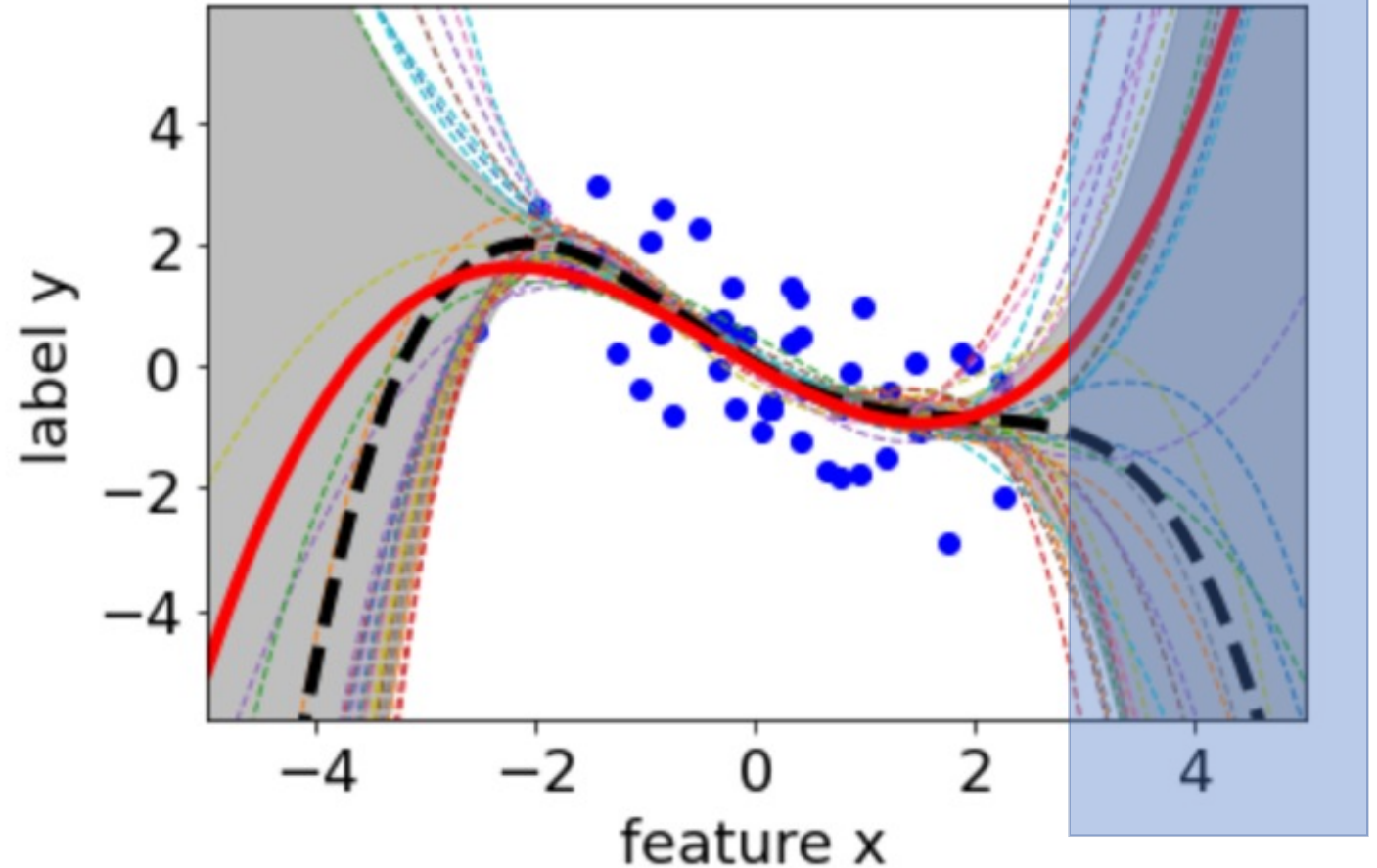
# Smaller Model (Poly.Degree)

- small variance
- large bias

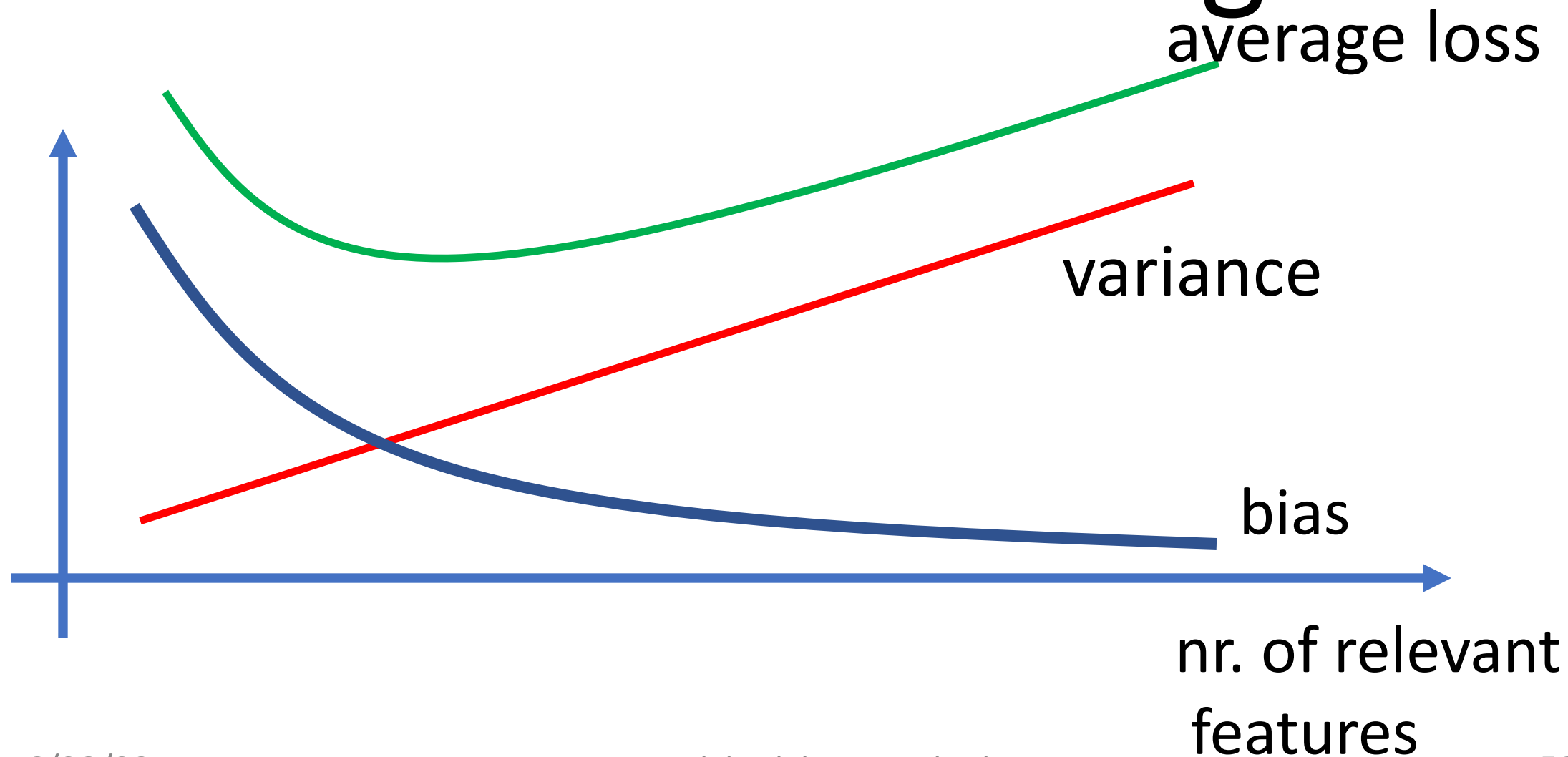


# Larger Model (Poly. Degree)

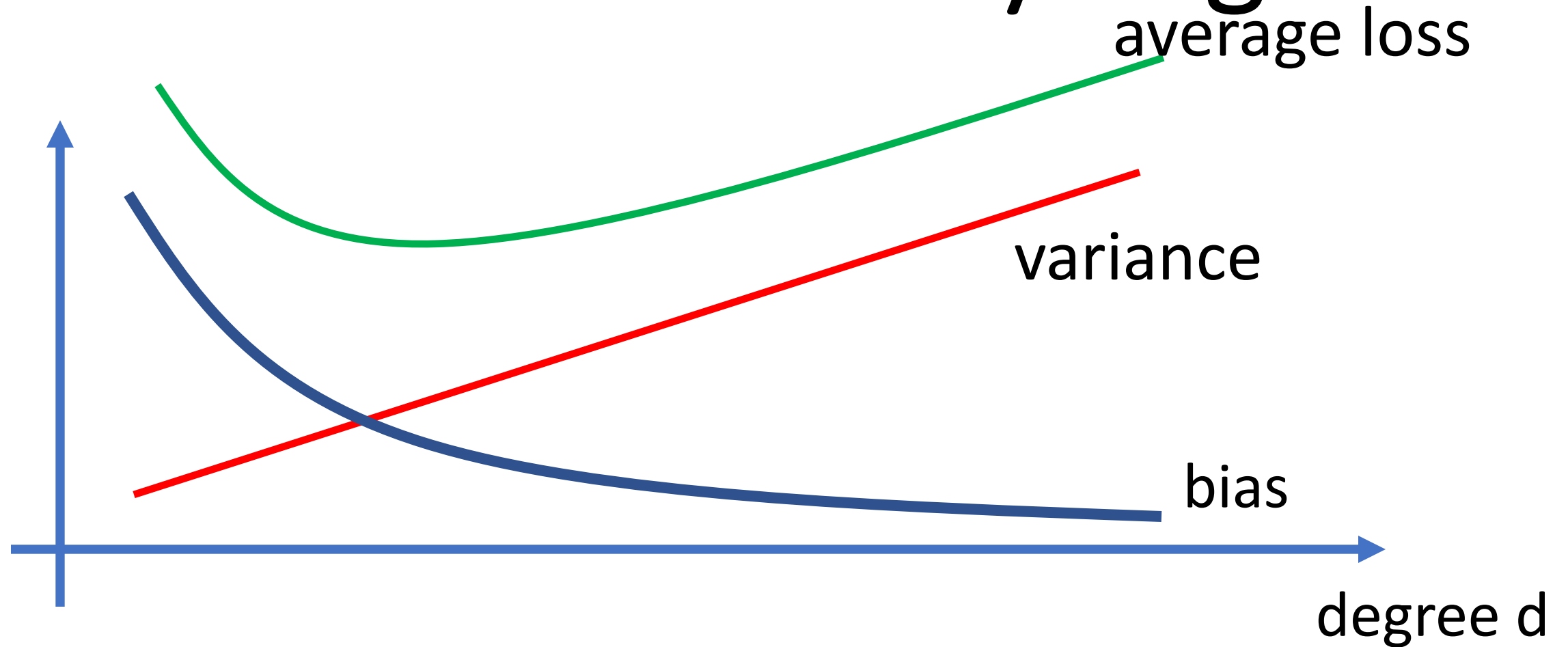
- large variance
- small bias



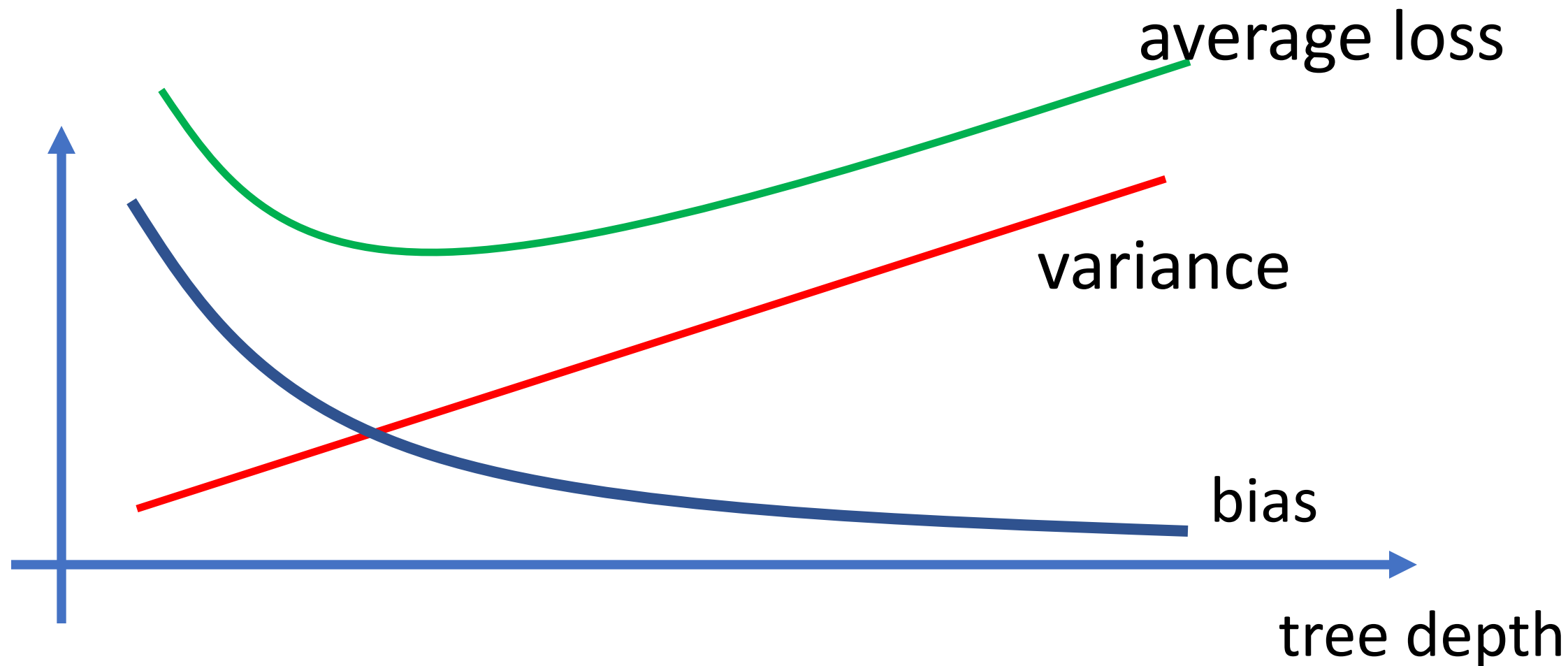
# Bias vs. Variance Lin.Reg.



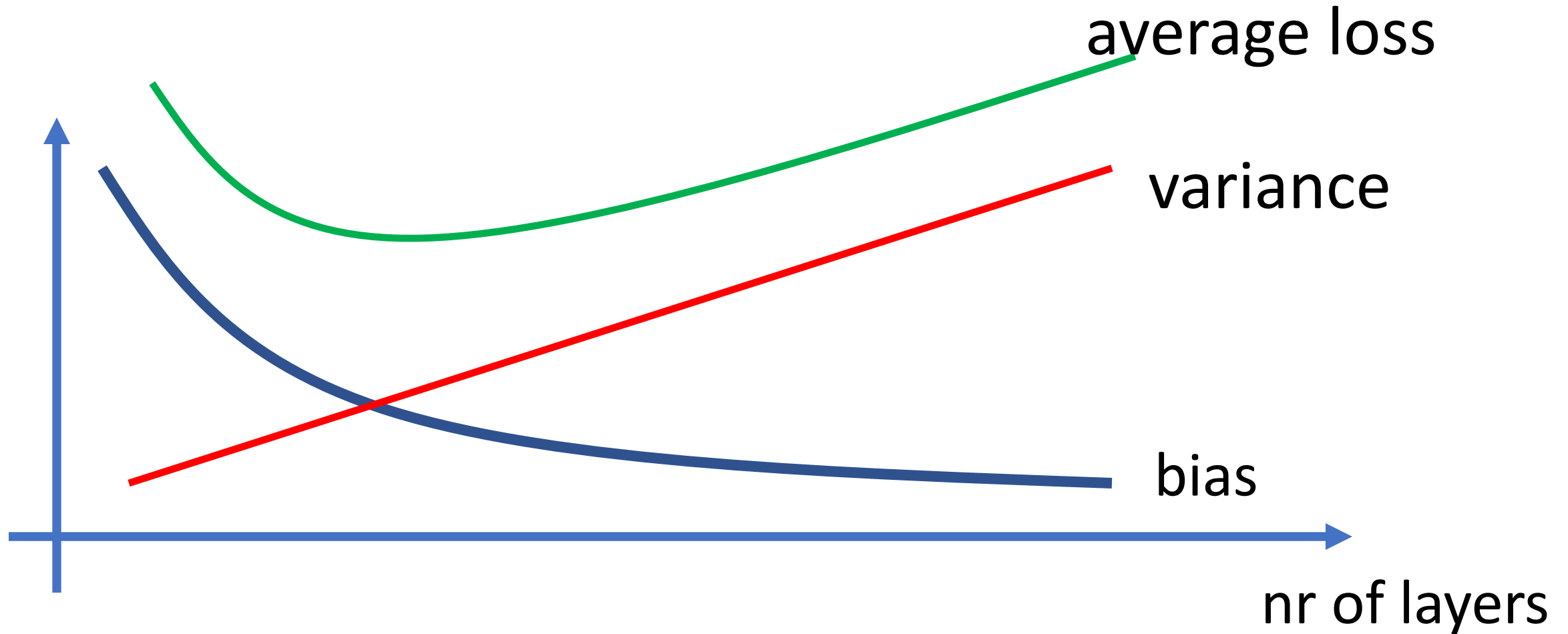
# Bias vs. Variance Poly.Reg.



# Bias vs. Variance Dec. Tree.

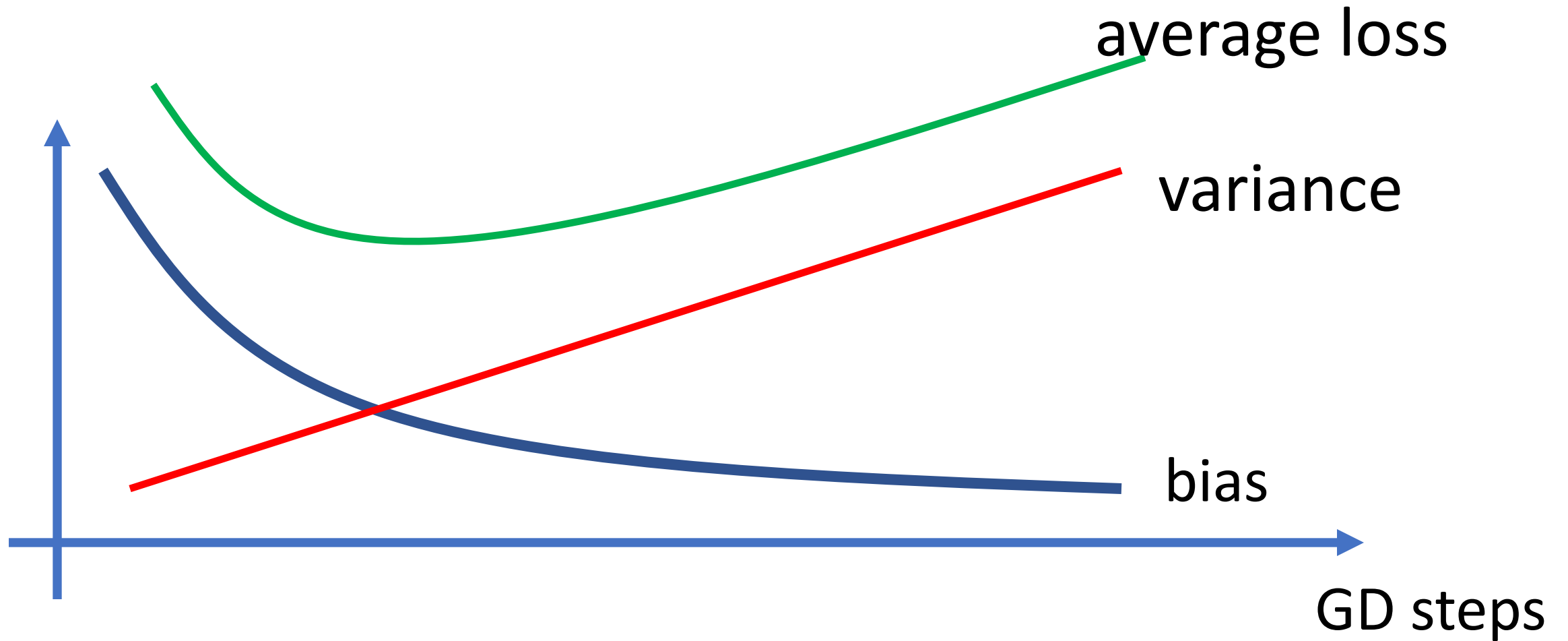


# Bias vs. Variance Deep Learning



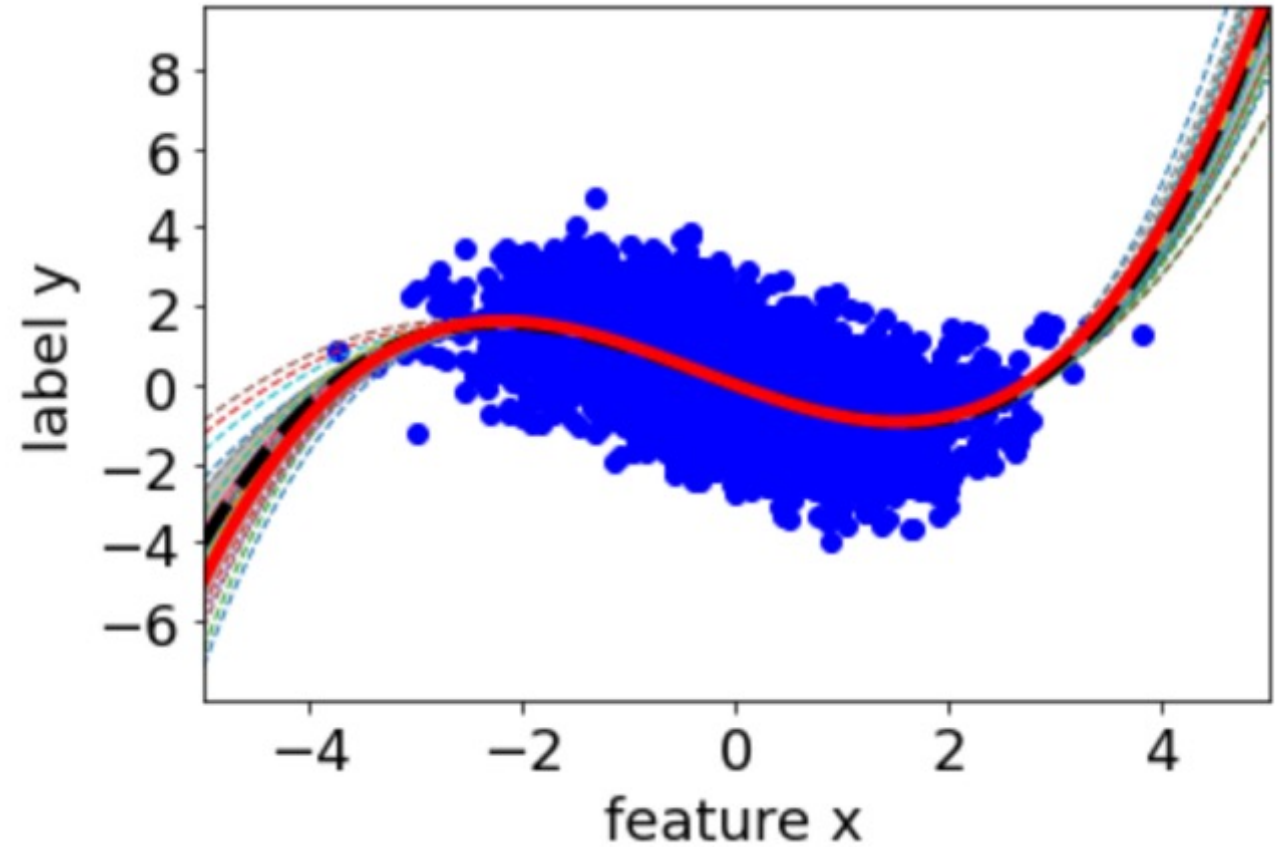


# Bias vs. Variance Grad. Desc.



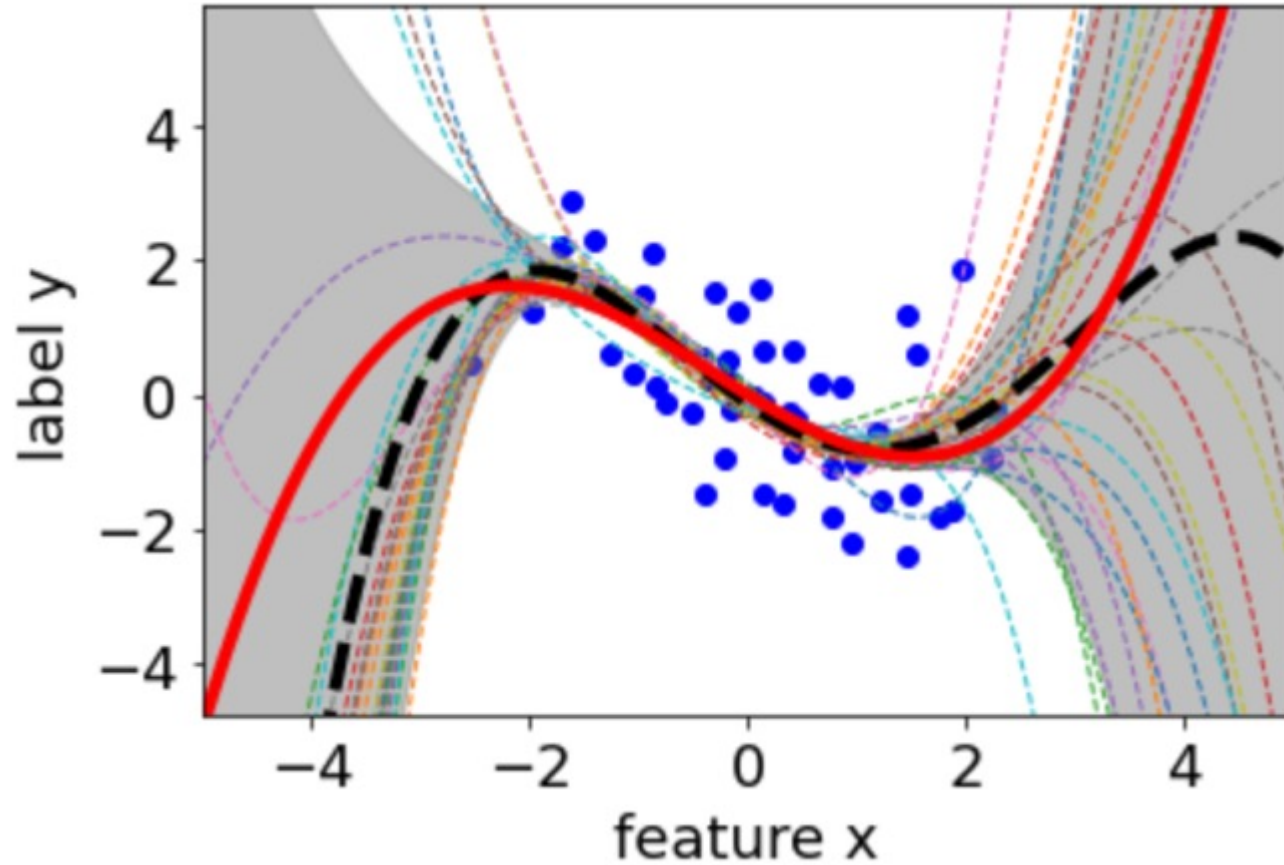
# More Data

-> smaller variance



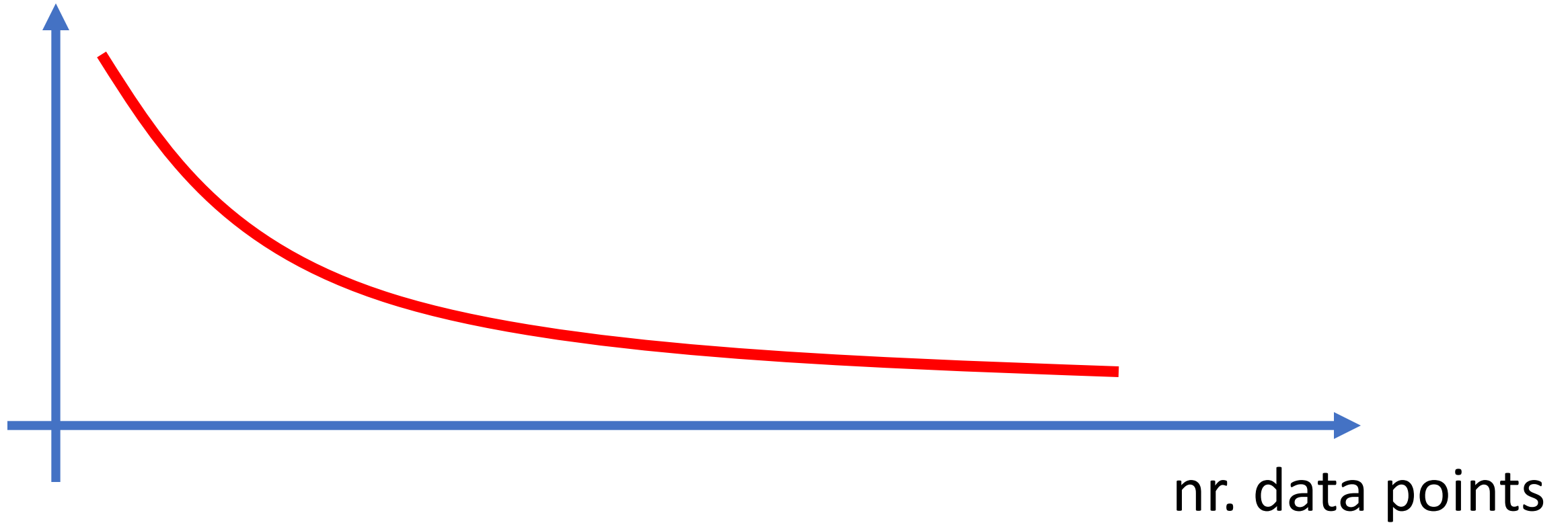
# Less Data

-> larger variance



# Learning Curve

variance



# Alex' Rule of Thumb

effective number of training data points

>

10 \* nr. tunable effective model parameters

stretch the term “effective” as much as possible !

# ML Diagnosis

# Simple Recipe

- consider ML method with some hypothesis space
- learn hypothesis by min. average loss on train.set
- training error = average loss of learnt hypothesis
- compute validation error
- compare val err, train err with a baseline

# Benchmark/Baseline

could be obtained from

- probabilistic models
- domain expertise
- existing ML methods
- human performance
- ...



training  
error

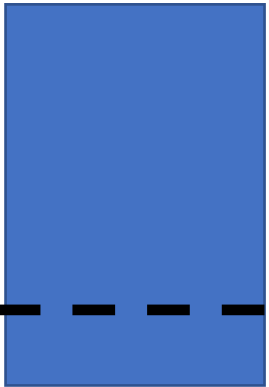
validation  
error

- small train error  $\rightarrow$  hypothesis space is large
- large val err  $\rightarrow$  overfitting
- Workaround ?

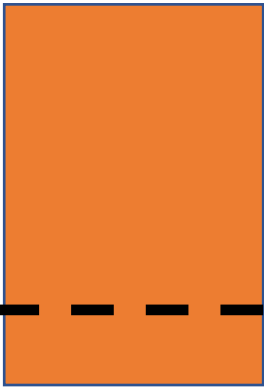
benchmark



training  
error



validation  
error



- large train error  $\rightarrow$  no good hypothesis found
- Workaround ?

training  
error

validation  
error



- Case Solved !

# Take Home Messages

- large models (e.g. deep nets) often overfit
- small training error does not mean much!
- diagnosis by comparing train/val err
- bias/variance analysis can guide model improvement

# Thank You !