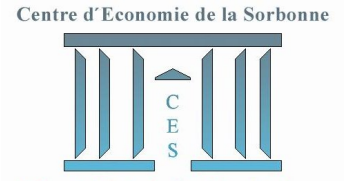
A Compact Introduction to Machine Learning

Amir SANI, PhD

Université Paris 1 Pathéon-Sorbonne,
Centre d'Économie de la Sorbonne,
CNRS and Paris School of Economics

Université Paris II Panthéon-Assas
ISF Masters 2
Paris, France
January 5, 2017



Welcome!

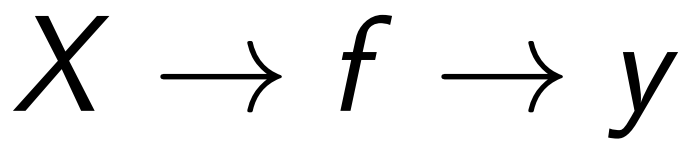
Please go to:
ML4EF.github.io

Click on “Information Sheet” under Class 1

Today's Class

What is Machine Learning?


Input → Predictor → Output


$X \rightarrow f \rightarrow y$

Binary Classification

$$\text{“Labels”}\ y \in \{-1, 1\}$$

(or equivalently $y \in \{0, 1\}$)

Regression

Labels $y \in \mathbb{R}$

Supervised Machine Learning

Supervised Machine Learning: Step 1

Given examples,

$$X^{Examples}$$

learn a function mapping to labels

$$y^{Examples}$$

Supervised Machine Learning: Step 1

Learn a function mapping,

$$X^{Examples} \rightarrow \hat{f} \rightarrow y^{Examples}$$

Supervised Machine Learning:Step 2

Use

$$\hat{f}$$

to map new data

$$X^{Out-of-Sample}$$

to predicted labels

$$y^{Predicted}$$

Supervised Machine Learning:Step 2

Map new data,

$$X^{Out-of-Sample} \rightarrow \hat{f} \rightarrow y^{Predicted}$$

Hypothesis Class

The Feature Map

$$\phi(X) \in \mathbb{R}^d$$

maps

$$X$$

to the d-dimensional space,

$$\phi(X) = [\phi_1(X), \ldots, \phi_d(X)]$$

Your Hypothesis Class is defined by,

$$\phi(X) = [\phi_1(X), \ldots, \phi_d(X)]$$

"Learning" aims to locate the weight vector

$$w \in \mathbb{R}^d$$

over the feature mapping

$$\phi(X) \in \mathbb{R}^d$$

where

$$w\phi(X) = \sum_{j=1}^{d} w_j \phi_j(X)$$

Performance is measured according to a loss,

$$l(w, x, \phi(\cdot), y),$$

Standard Loss functions include,

$$\text{Square Loss: } (w\phi(X) - y)^2$$
$$\text{Absolute Loss: } |w\phi(X) - y|$$

Learning is the optimization defined by,

$$\min_{w \in \mathbb{R}^d} loss(w, X^{Examples}, \phi(\cdot), y^{Examples})$$

Classification vs. Regression

	Classification	**Regression**
Predictor $f_w(\cdot)$	$\text{sign}(y^{\text{Predicted}})$	$y^{Predicted}$
Distance to y^{True}	$\text{margin}(y^{\text{Predicted}})$	$\text{residual}(y^{\text{Predicted}})$
Loss functions	zero-one hinge logistic	square absolute

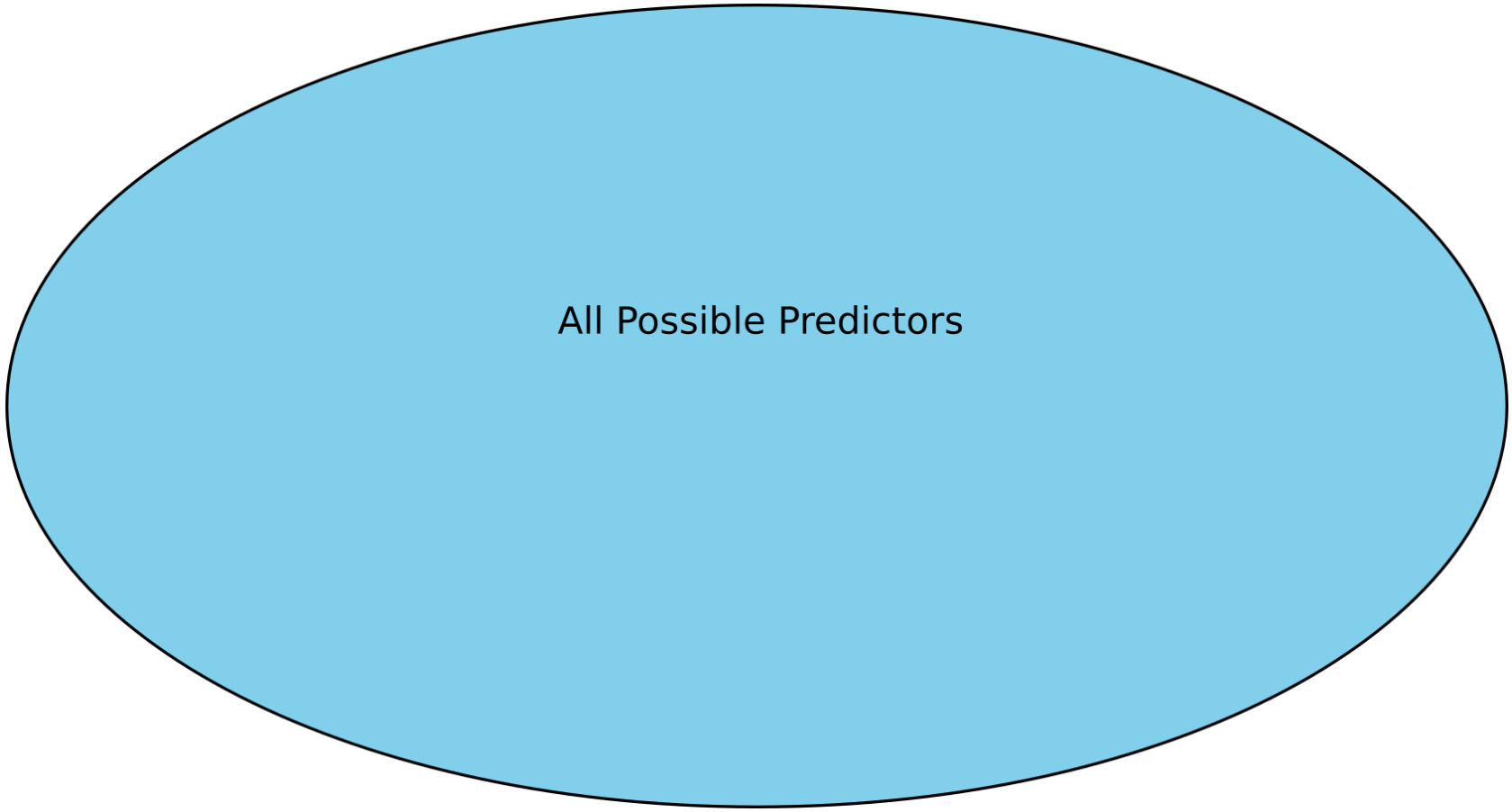
What is Expressiveness?

Does the Hypothesis class,

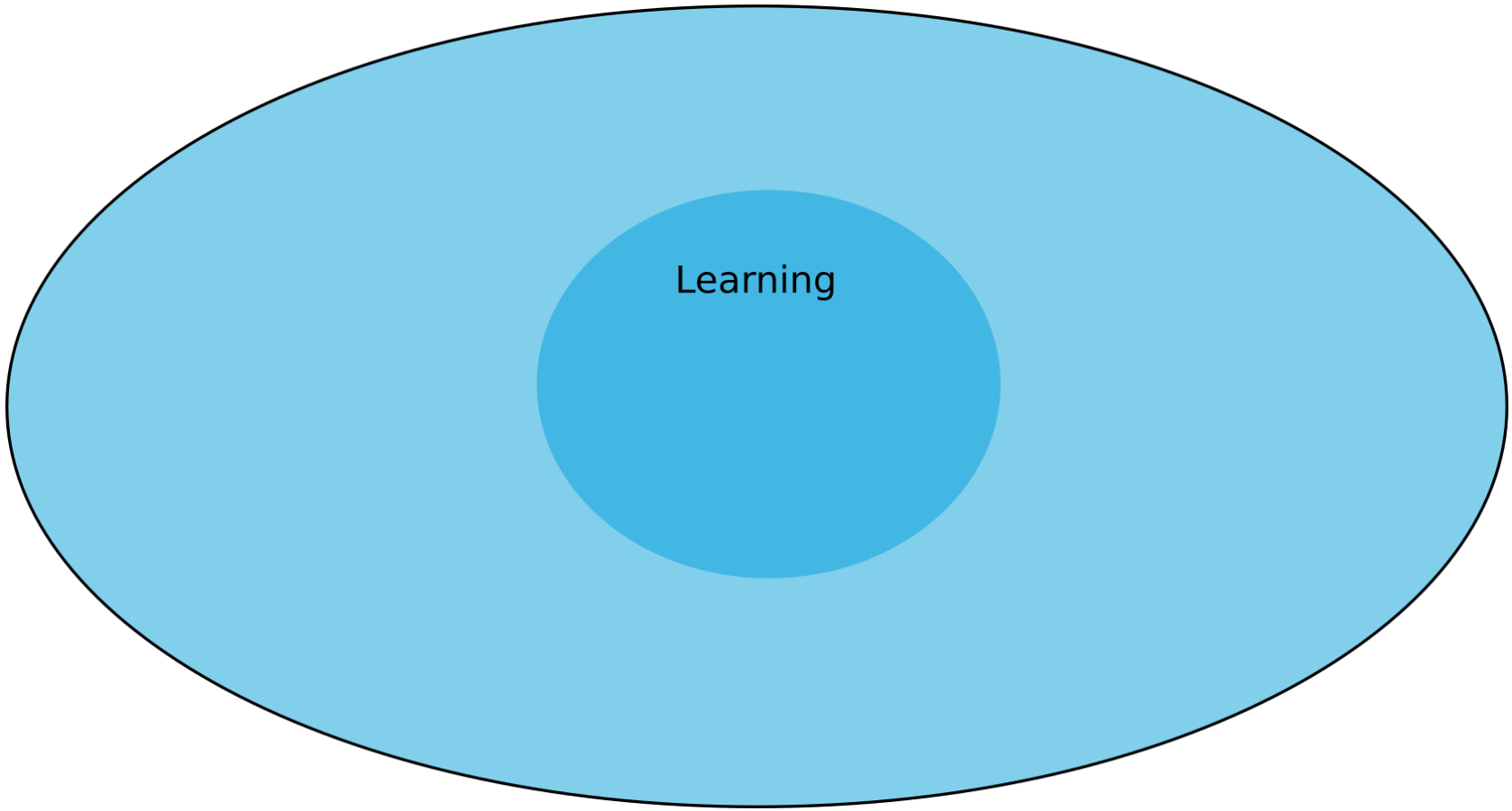
$$\mathcal{F} = \{f_w : w \in \mathbb{R}^d\}$$

contain a good predictor?

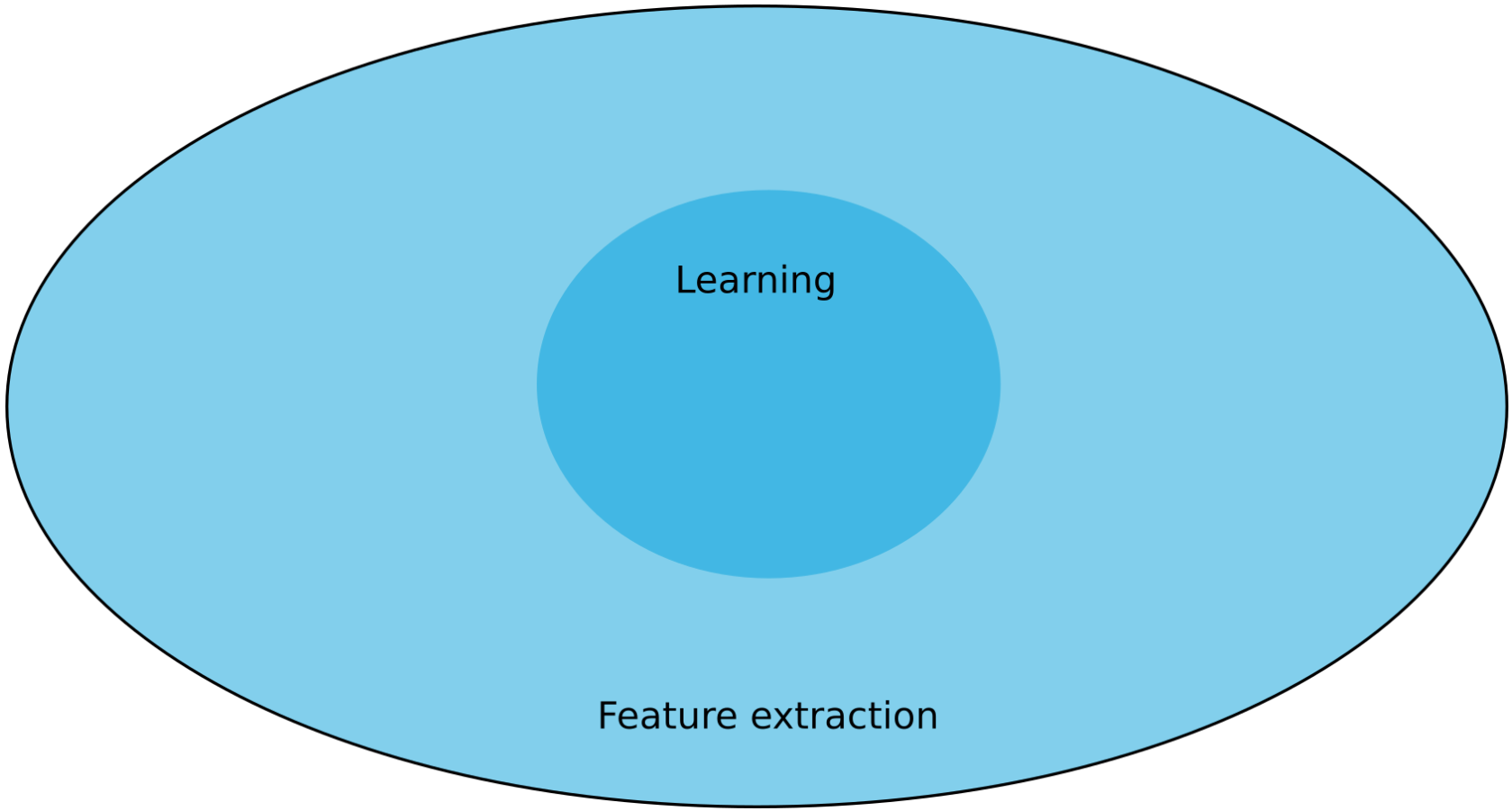
Consider,



Learning can find predictors defined by a fixed $\theta(X)$ and varying weights w



Feature Extraction can potentially access better predictors



What are some common ways to extract features?

Feature Extraction

- Domain Knowledge
- Feature Interactions
- Polynomial Expansions
- Kernel Maps
- (Unsupervised) Feature Learning (e.g. Deep Learning)
- Semi-supervised (Manifold) Learning
- Cluster Analysis (e.g. T-SNE, MDS)
- etc.

How can I tell how well my algorithm will perform out of sample?

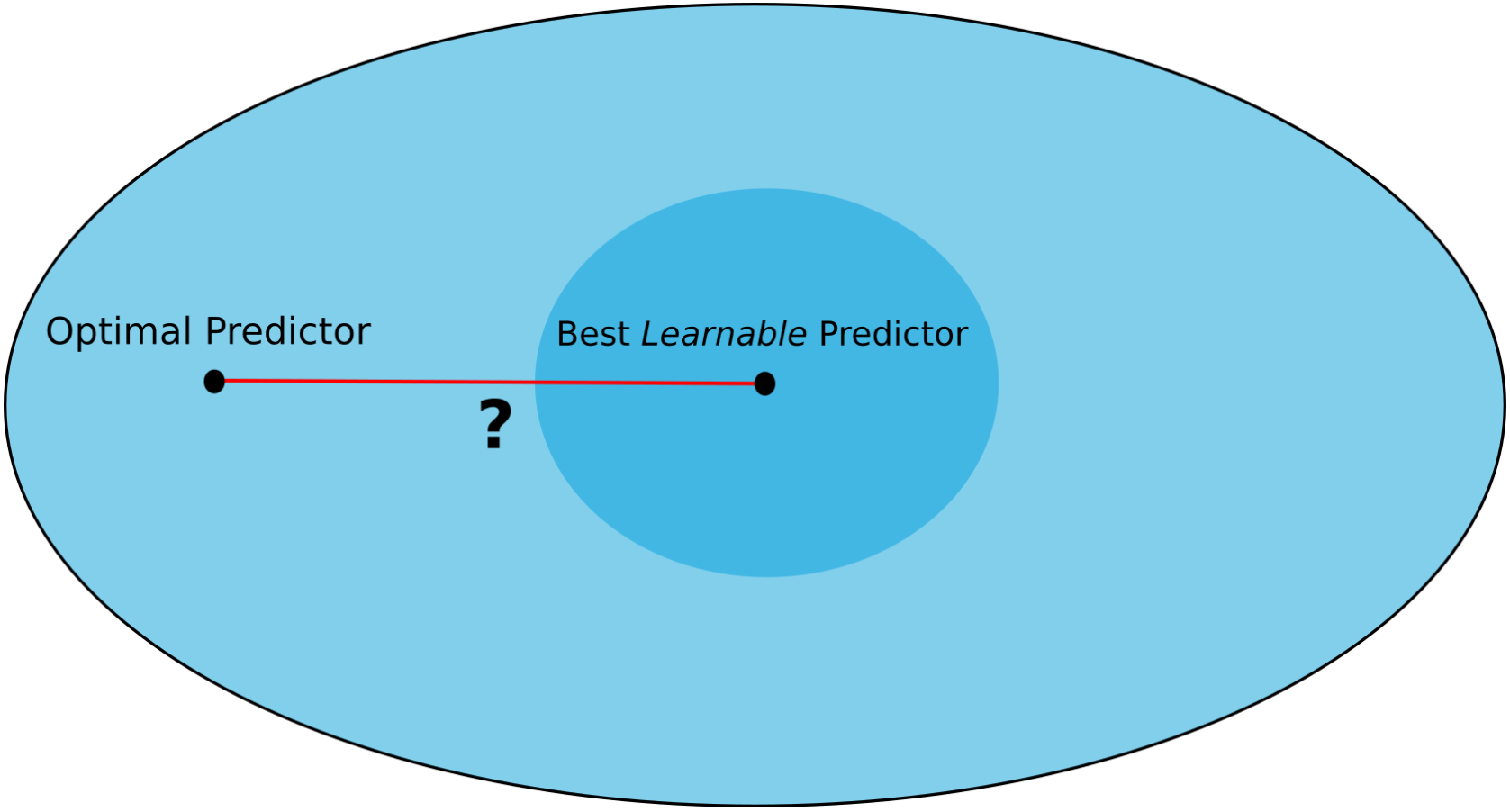
Generalization

How far is the best predictor (in the smaller circle),

$$B = \arg\min_{f \in \mathcal{F}} \mathit{Error}(f),$$

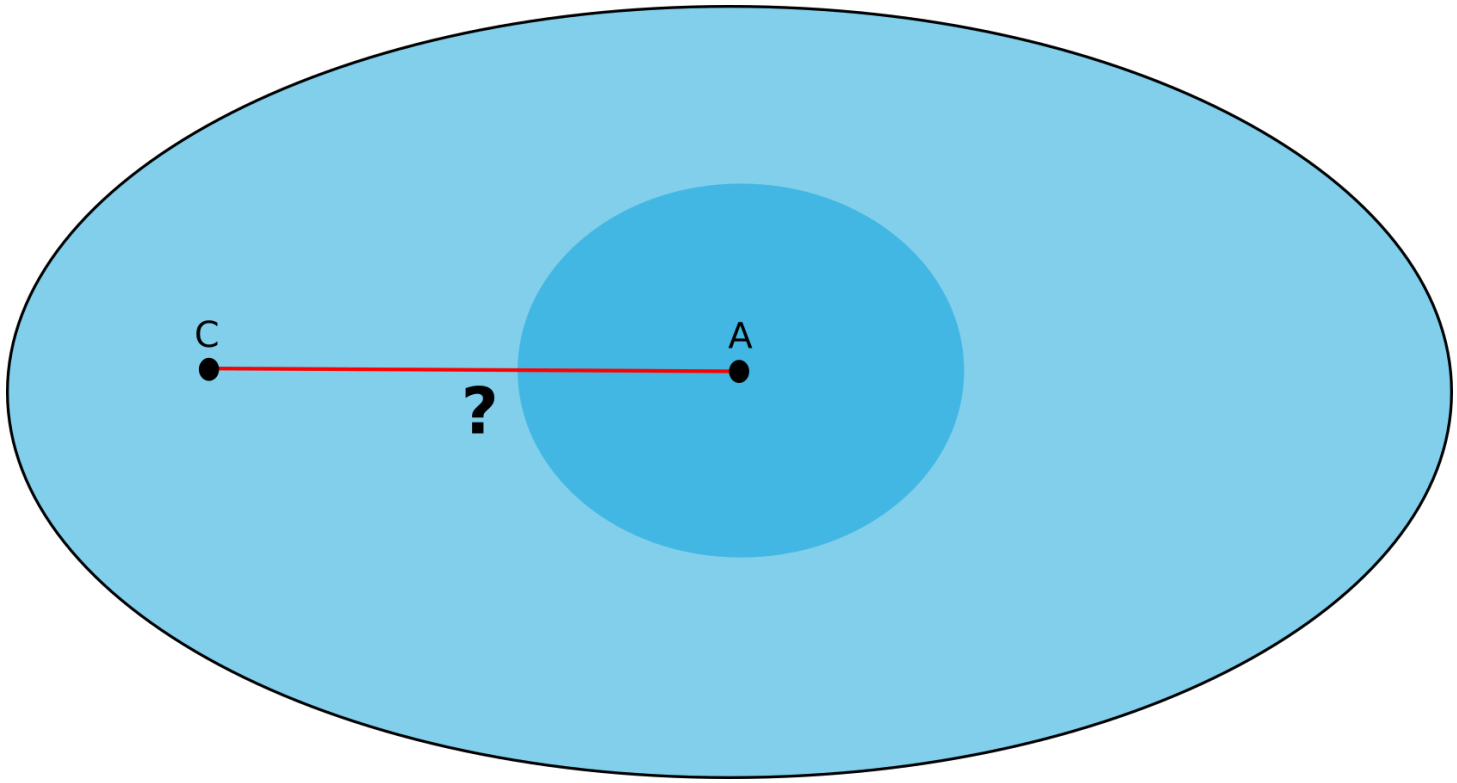
from the best possible predictor
(in the larger circle)?

Generalization Performance



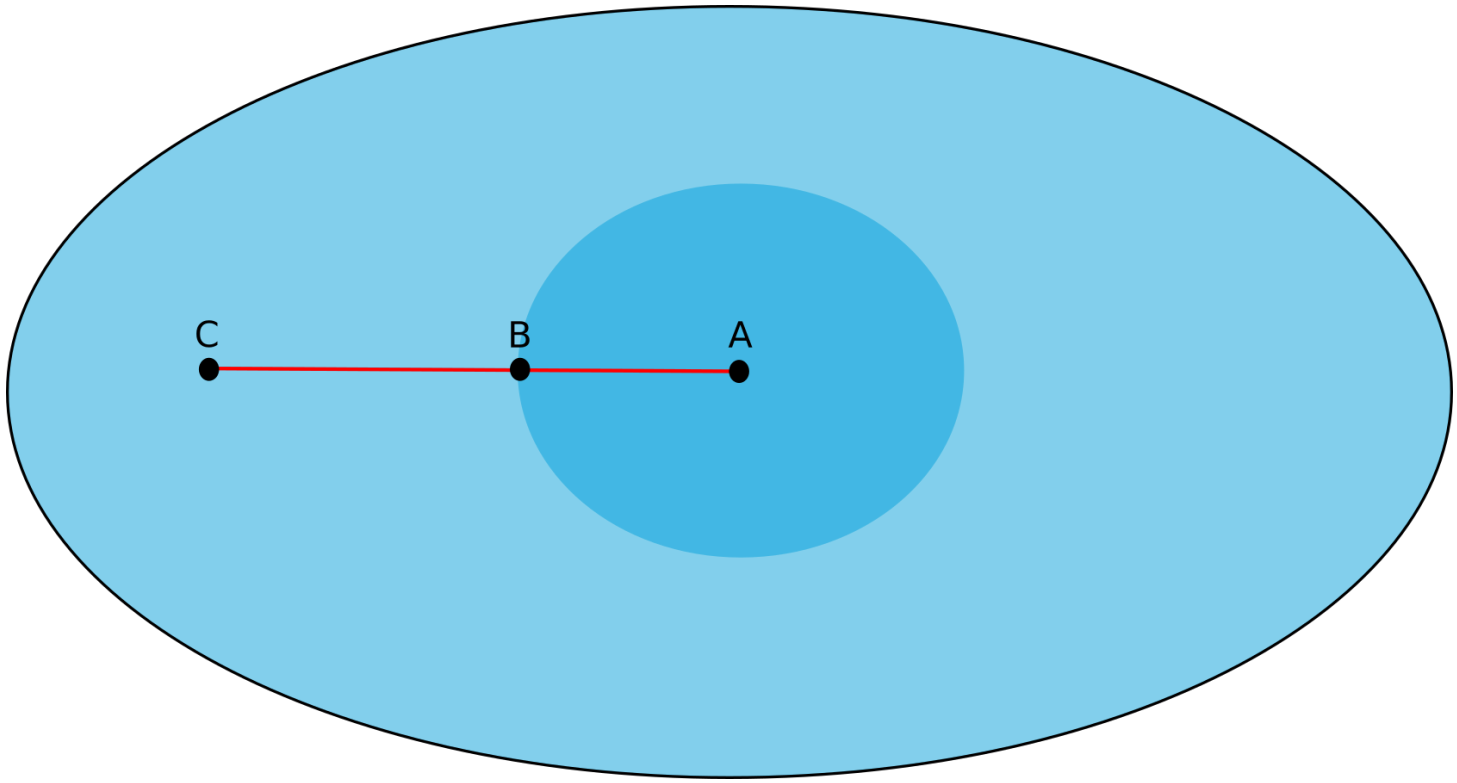
Generalization Error

$$Error[A] - Error[C]$$



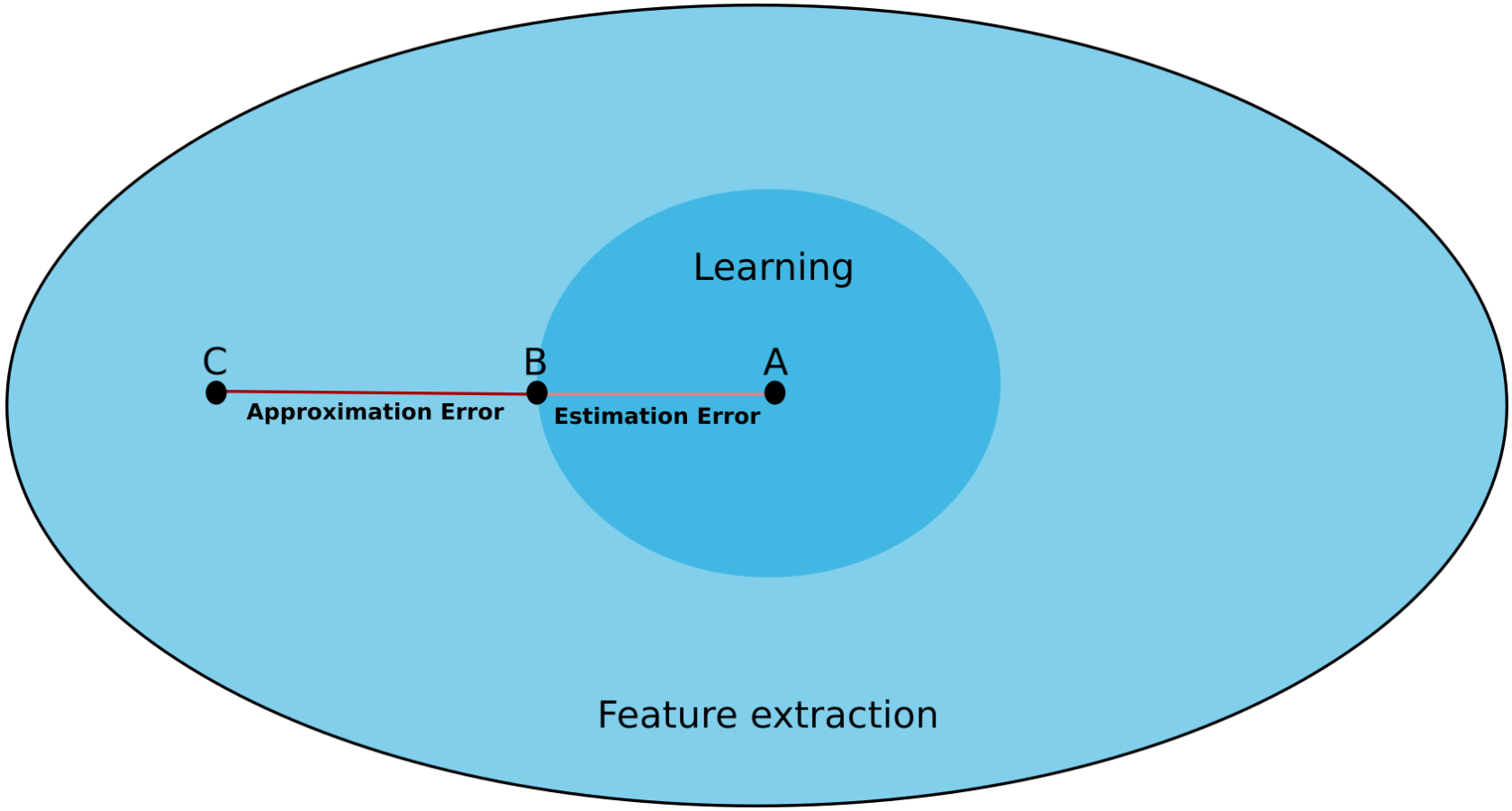
Generalization Error

$$Error[A] - Error[B] + Error[B] - Error[C]$$



Generalization Error

Approximation vs. Estimation Error



Approximation Error

How far is the Hypothesis Class from the Optimal Predictor C?

$$Error(B) - Error(C)$$

Approximation Error

Determined by Quality of Hypothesis Class

LARGER Hypothesis Class

$\downarrow$

smaller Approximation Error

Estimation Error

The learning algorithm's ability to locate the best "learnable" predictor.

$$Error(A) - Error(B)$$

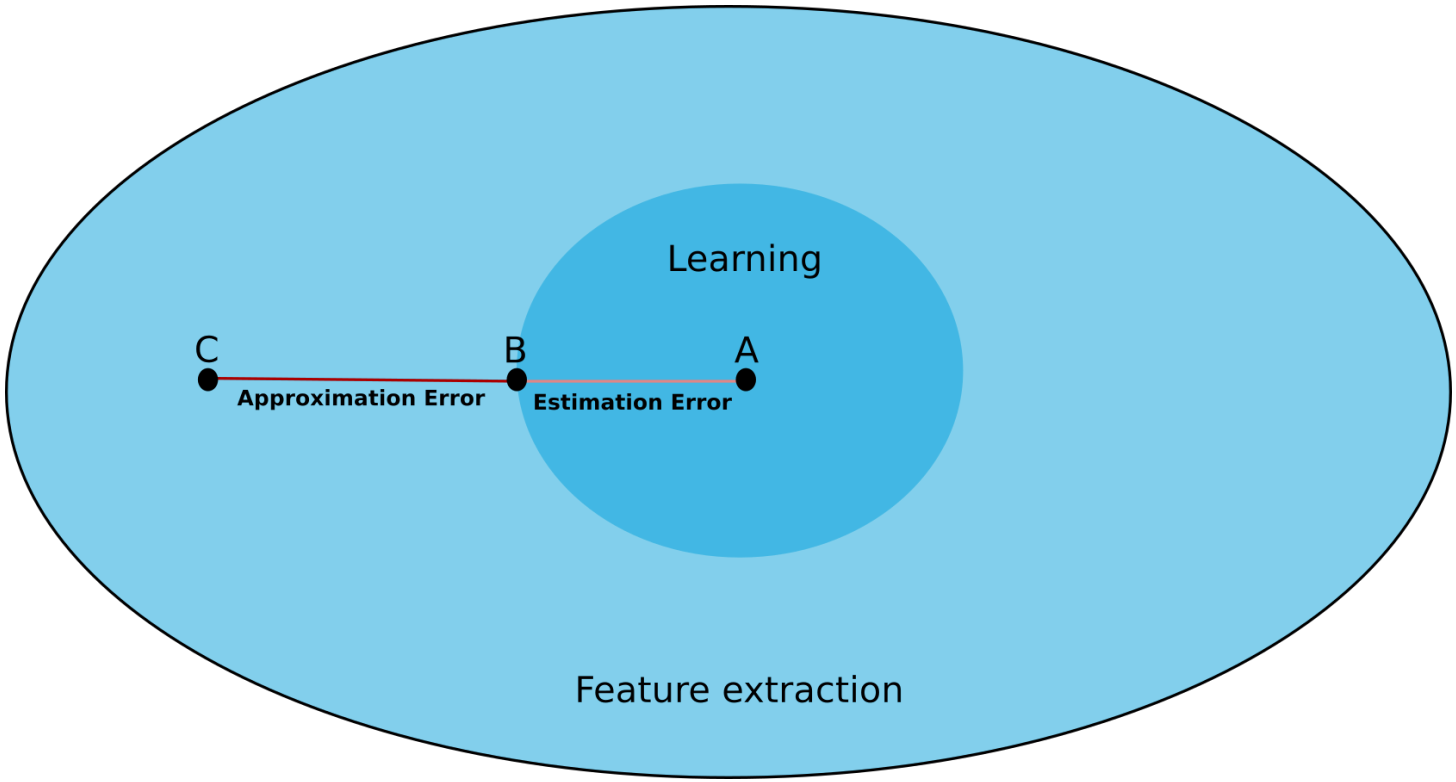
LARGER Hypothesis Class

$\downarrow$

LARGER Approximation Error

Generalization Error

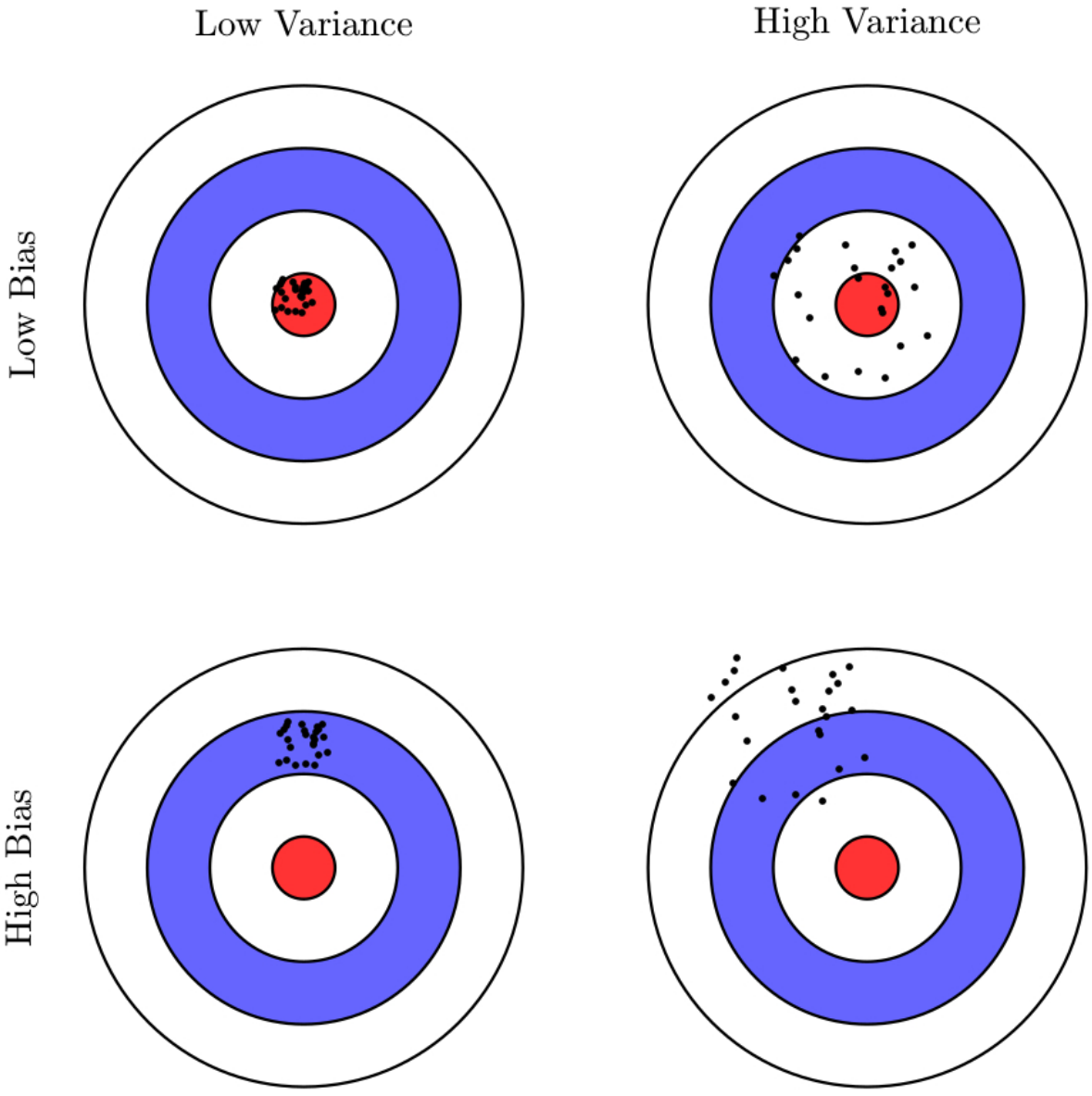
$$\underbrace{Error(A) - Error(B)}_{\text{Estimation Error}} + \underbrace{Error(B) - Error(C)}_{\text{Approximation Error}}$$



Learning Objective

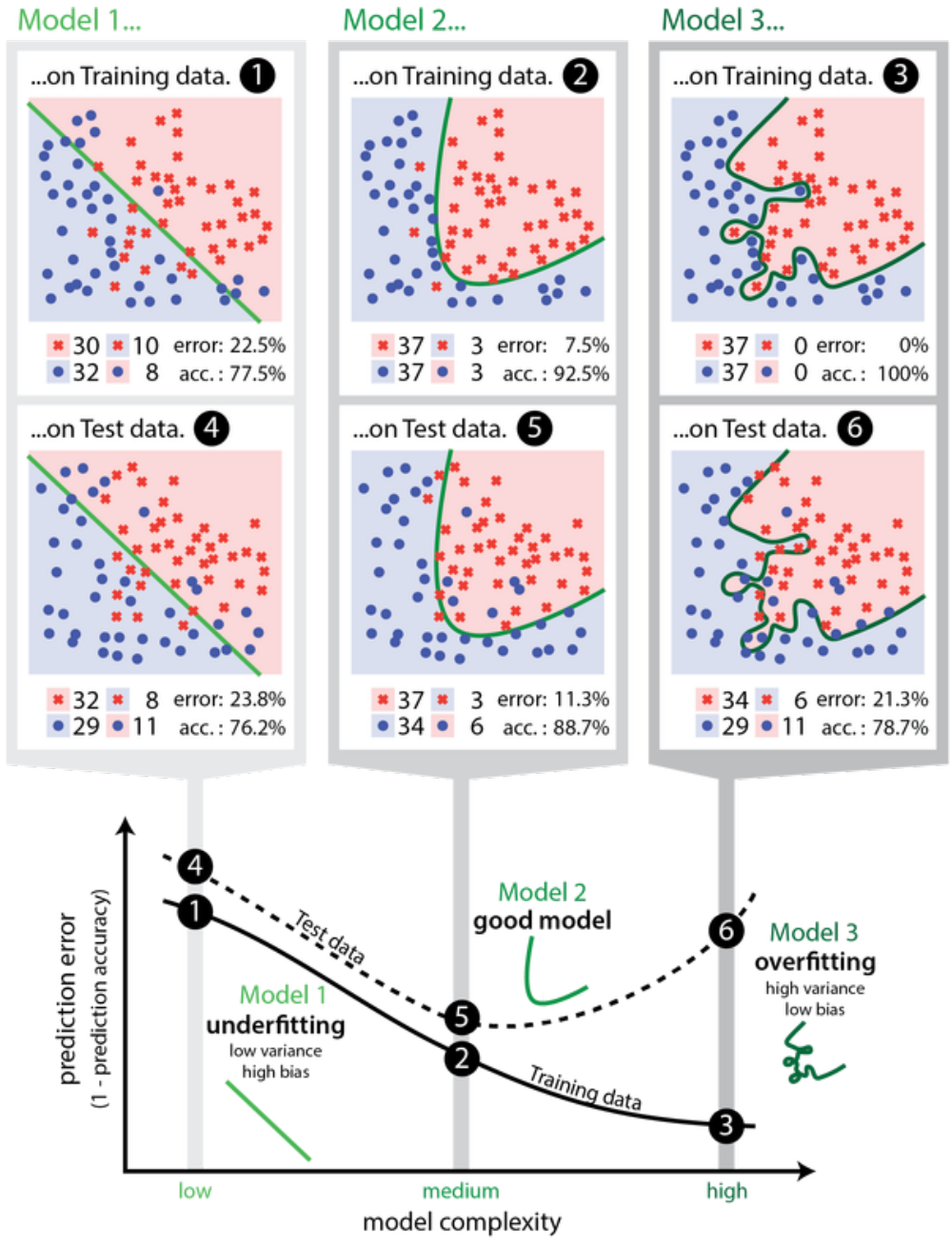
$$\text{Minimize} \left(\begin{array}{c} \text{Estimation Error} \\ \& \\ \text{Approximation Error} \end{array} \right)$$

“Bias–Variance Trade-off”



Bias–Variance Trade-off

$$\underbrace{Error(A) - Error(B)}_{\text{Variance}} + \underbrace{Error(B) - Error(C)}_{\text{Bias}}$$


Model 1...
Model 2...
Model 3...
...on Training data. 1
30 10 error: 22.5%
32 8 acc.: 77.5%
...on Training data. 2
37 3 error: 7.5%
37 3 acc.: 92.5%
...on Training data. 3
37 0 error: 0%
37 0 acc.: 100%
...on Test data. 4
32 8 error: 23.8%
29 11 acc.: 76.2%
...on Test data. 5
37 3 error: 11.3%
34 6 acc.: 88.7%
...on Test data. 6
34 6 error: 21.3%
29 11 acc.: 78.7%
prediction error
(1 - prediction accuracy)
Test data
Training data
Model 1
underfitting
low variance
high bias
Model 2
good model
Model 3
overfitting
high variance
low bias
low
medium
high
model complexity

Avoid Over/Under-Fitting with Feature Selection

- Remove Features
- Simplify Features
- Regularization[1]
- Avoid Over-Optimization (Data Mining)

[1] Penalize norm on w by adding additional penalty terms.

Feature Selection

- Forward Selection
- Backward Selection
- L1 (LASSO)
- Elastic Net
- Boosting
- Many many more

A lot of useful feature selection algorithms have already been implemented. Use them!

Thank you!