Machine Learning I
80-629

Apprentissage Automatique I
80-629

Machine Learning fundamentals
— Week #2

Today: what's a machine learning problem

- Core concepts
  - Modeling and parameters
  - Bias/Variance
  - Overfitting
  - Representing uncertainty
- Types of learning problems
  - Supervised learning (generative and discriminative models)
  - Unsupervised learning
  - Reinforcement learning

Capsules

1. Machine Learning problem
2. Types of Learning problems
3. A first Supervised Model
4. Model Evaluation
5. Regularization
6. Bias/Variance

- I will follow the exposition of Chapter 5 in “Deep Learning”.
  - “Operational” approach vs. a decision-theoretic/probabilistic approach

The components of a learning problem

- I will follow the exposition of Chapter 5 in “Deep Learning”
  - “Operational” approach vs. a decision-theoretic/ probabilistic approach

Three main components

- **Task (T)**
- **Performance measure (P)**
- **Experience (E)**

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, measured by P, improves with experience E."

–Tom Mitchell (1997)

Task (T)

- The end goal(s). The question you are answering.
- For example:
  - Self-driving
  - Differentiate cats from dogs
  - Recommend movies of interest to users
  - Select a good portfolio of stocks

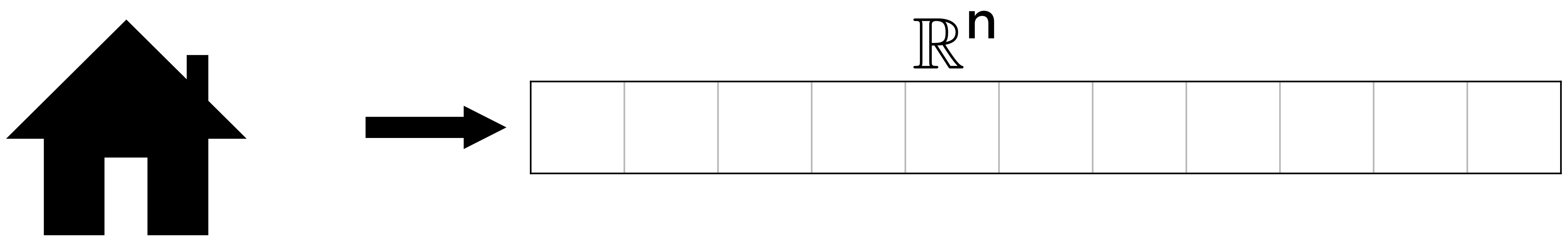
Task (T)

Task (T)

- **Determine the price of houses?**

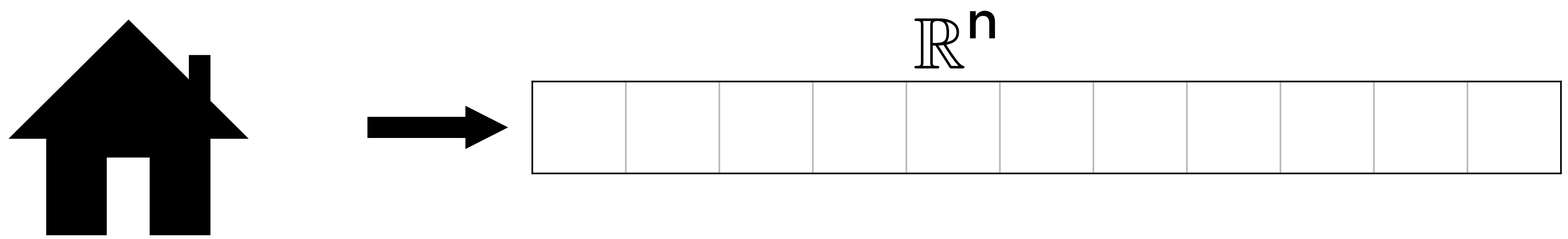
Task (T)

- Determine the price of houses?
  - Encode houses into a set of features
    - area, number of rooms (bedrooms, bathrooms), municipal evaluation, neighborhood, etc.

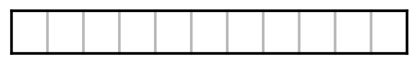


Task (T)

- Determine the price of houses?
  - Encode houses into a set of features
    - area, number of rooms (bedrooms, bathrooms), municipal evaluation, neighborhood, etc.



  - Function from feature to house price

f :  $\rightarrow$ price

$$f : \mathbb{R}^n \rightarrow \mathbb{R}^+$$

Example tasks

- **Regression: Assign a real value to an example**

$$f : \mathbb{R}^n \to \mathbb{R}$$

- **Classification: Classify instances in one of k classes**

$$f : \mathbb{R}^n \to \{1, \ldots, k\}$$

- **Clustering: Assign each instance to a cluster**

$$f : \mathbb{R}^n \to \{1, \ldots, k\}$$

More examples

- **Transcription (e.g., document classification)**

$$f : \mathbb{R}^{n \times m} \to \mathbb{R}^{k}$$

- **Multi-label classification (e.g., tag prediction)**

$$f : \mathbb{R}^{n} \to \{0, 1\}^{m}$$

- **Translation (e.g., sentence from French to English)**

$$f : \mathbb{R}^{n} \to \mathbb{R}^{m}$$

Model

- functions f are examples of models
  - Model is a simpler representation of the world
    - Has parameters (w)

https://www.istockphoto.com/ca/photos/toy-car

Model #1: $f(x; w) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$

$f : \mathbb{R}^n \rightarrow \mathbb{R}$

Model #2: $f(x; w) = w_1 x_1 + w_2 x_1^2 + w_3 x_2 + \ldots + w_{n+1} x_n$

The performance measure (P)

- **How to quantify the "goodness" of f? How to compare models?**

The performance measure (P)

- How to quantify the “goodness” of f? How to compare models?
    - Specific to the task at hand
        - E.g., linear regression: squared error

The performance measure (P)

- How to quantify the “goodness” of f? How to compare models?
  - Specific to the task at hand
    - E.g., linear regression: squared error

$$\frac{1}{P} := \sum_i \left(f(x_i) - y_i\right)^2$$

f_1 is better than f_2 if $P_1 > P_2$

The performance measure (P)

- How to quantify the “goodness” of f? How to compare models?
  - Specific to the task at hand
    - E.g., linear regression: squared error

$$\frac{1}{P} := \sum_i \left(f(x_i) - y_i\right)^2$$

$$f_1 \text{ is better than } f_2 \text{ if } P_1 > P_2$$

  - Encodes knowledge of what’s important
    - A model with perfect performance behaves perfectly
- Examples: accuracy, error rate, log-probability, F score.

Experience (E)

- What data does f experience?
  - (Focus on algorithms that experience whole datasets)
  - Unsupervised. Examples alone.

$$\{x_i\}_{i=0}^{n}$$

  - Supervised. Examples come with labels.

$$\{(x_i, y_i)\}_{i=0}^{n}$$

Different types of experience

1. Unsupervised Learning
2. Supervised Learning
3. Reinforcement Learning

Recall

Experience (E)

- What data does f experience?
  - (Focus on algorithms that experience whole datasets)
  - Unsupervised. Examples alone.

$$\{x_i\}_{i=0}^{n}$$

  - Supervised. Examples come with labels.

$$\{(x_i, y_i)\}_{i=0}^{n}$$

1. Unsupervised

1. Unsupervised

- **Experience examples alone**

$$\{\mathbf{x}_i\}_{i=0}^{n}$$

1. Unsupervised

- Experience examples alone

$$\{x_i\}_{i=0}^{n}$$

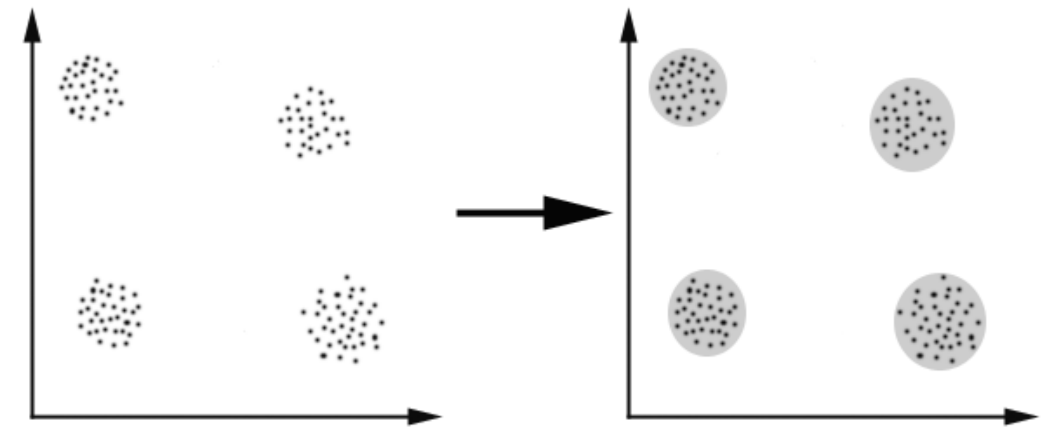
- Learn "useful properties of the structure of the data"

1. Unsupervised

- Experience examples alone

$$\{\mathbf{x}_i\}_{i=0}^{n}$$

- Learn "useful properties of the structure of the data"
  - E.g., Clustering



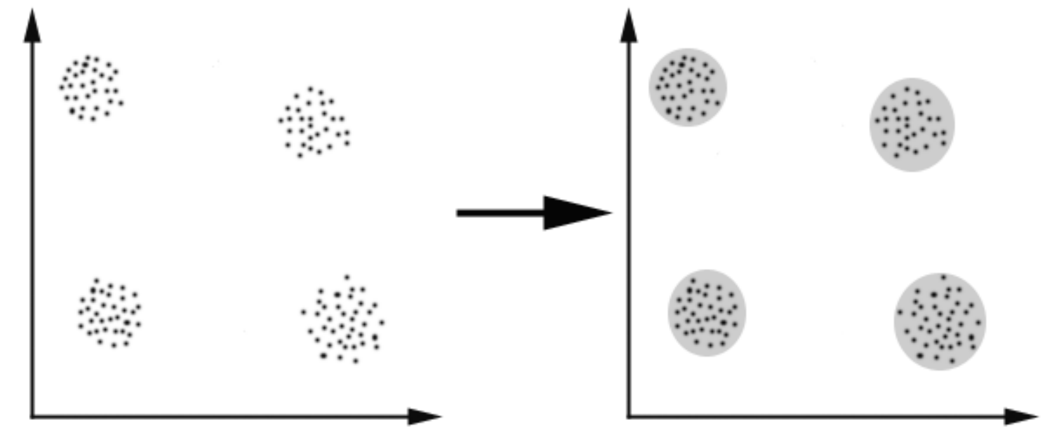
https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/

1. Unsupervised

- Experience examples alone

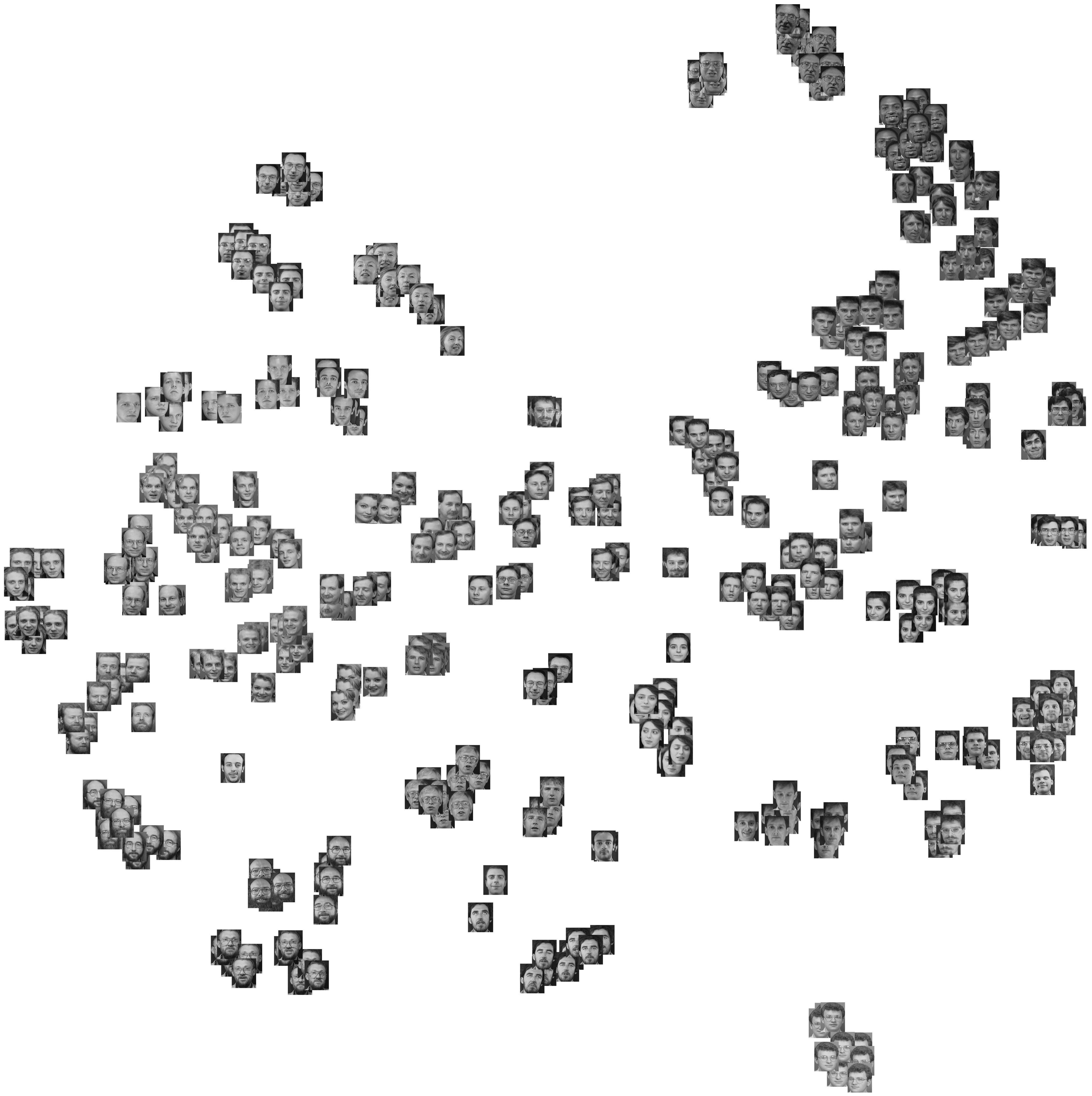
$$\{x_i\}_{i=0}^{n}$$

- Learn “useful properties of the structure of the data”
  - E.g., Clustering



https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/

- Probabilistic models
  - Density modeling p(x), PCA, FA.



Example from a non-linear dimensionality reduction technique (tsne)
https://lvdmaaten.github.io/tsne/

2. Supervised

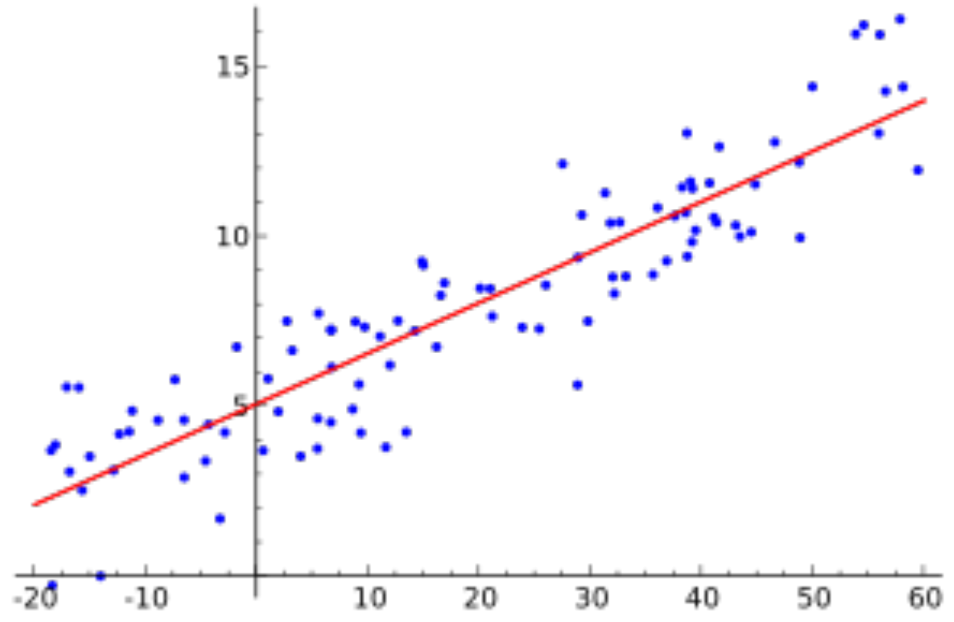
- Experience examples and their label(s)

$$\{(x_i, y_i)\}_{i=0}^{n}$$

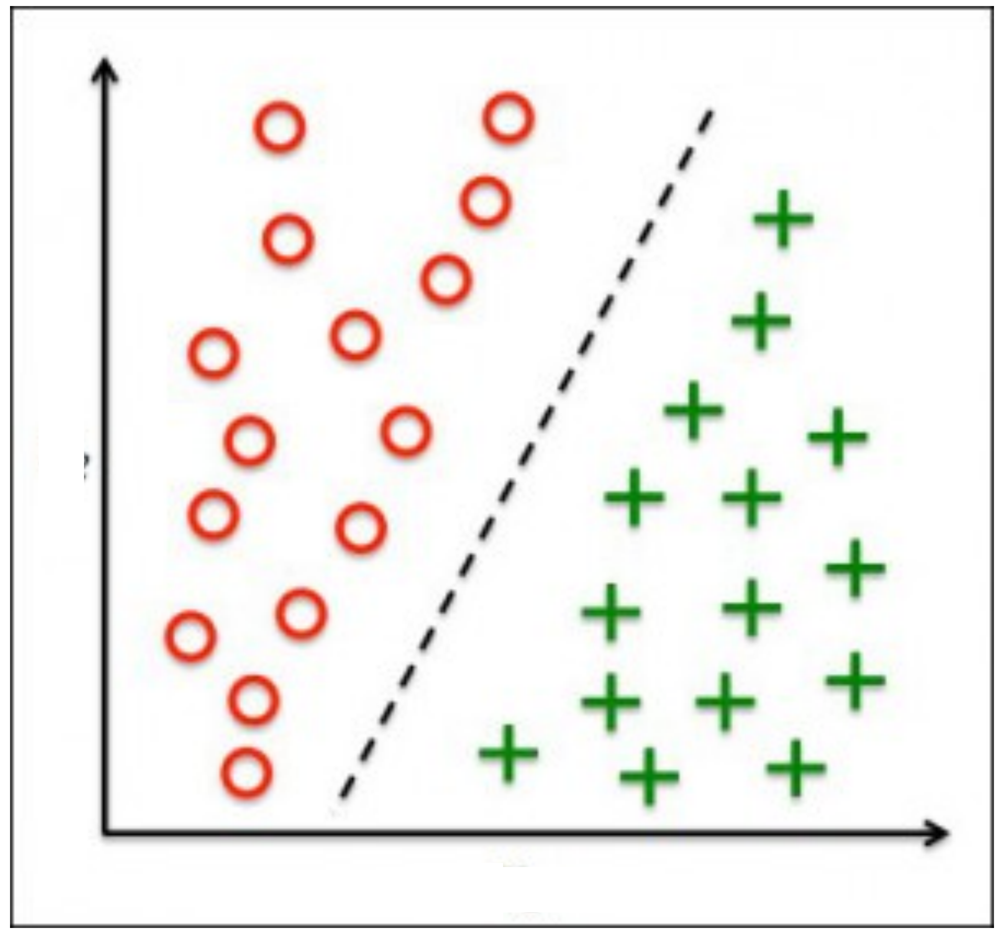
- Given an example (x) predict its label (y)

$$f : X \to Y \qquad p(y|x)$$

- E.g., regression, classification



[https://en.wikipedia.org/wiki/Regression_analysis]



[https://jaxenter.com/machine-learning-an-introduction-for-programmers-122135.html]

Distinction can be blurry

- **Supervised data modeled jointly:**

$$(x, y), \quad p(y \mid x) = \frac{p(x, y)}{\sum_{y'} P(x, y')}$$

- **Unsupervised data modeled as supervised data:**

$$x \in \mathbb{R}^n, \quad p(x) = \prod_{i=1}^{n} p(x_i \mid x_1, \ldots, x_{i-1}) P(x_0)$$

Conditional model: $P(y \mid x)$

Generative model: $P(y, x)$

Semi-supervised learning

- Idea: Can we augment a supervised dataset with unsupervised data

$$(\{(x_i, y_i)\}_{i=0}^{n}, \{x_j\}_{j=0}^{m})$$

- Unlabelled data are cheap (images on the web). Labeled data are expensive

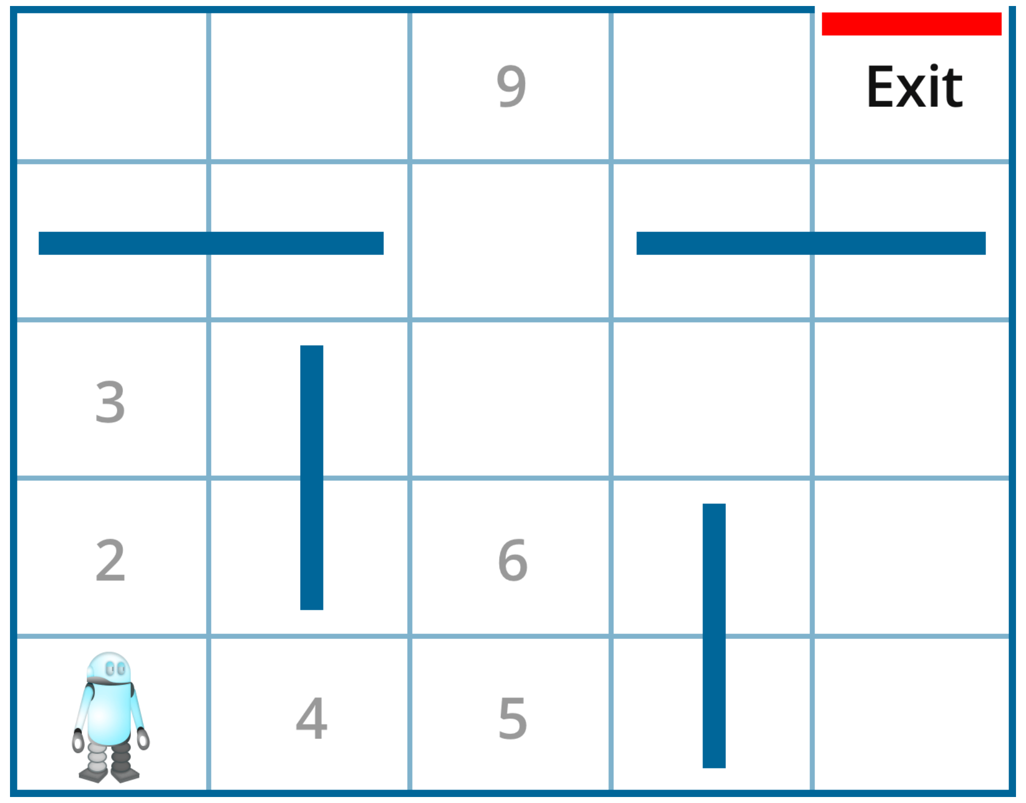
- Can the unlabelled data help model x and yield a better model for y?

[Dog]

[ ]

3. Reinforcement learning

- The algorithm interacts with the environment
  - The algorithm observes its environment. Prototypical example is a robot navigating a maze



[https://www.oreilly.com/ideas/reinforcement-learning-explained]

  - The resulting dataset depends on the algorithm's choices

A first supervised example

Dataset(s)

- Each instance: $\mathbf{x}_i$
- A dataset is a set of instances: $\{\mathbf{x}_i\}$
- Often denoted as a matrix X, (design matrix)
  - rows are instances
  - columns are features
  - (assumes that all instances can be encoded using a fixed-size vector)

	Nb.bed.	Area	Neigh.	·	·		Price
x_0	1	0	0	0	0	y_0	125000
x_1	1	100	1	.2	.5	y_1	150000
x_2	3	200	0	.1	.2	y_2	350000
x_3	1	150	1	.4	.1	y_3	275000
x_4	2	210	2	.5	1.1	y_4	225000

Concrete full example: Linear regression

$$y_i = w_0 x_{i0} + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_p x_{ip} \qquad w \in \mathbb{R}^p$$

$$= \sum_{j=0}^{p} w_j x_{ij} = w^\top x_i$$

- w is a vector of parameters (weights)
  - w_j represents the effect of feature j on y
- Task: predict y from x using $w^\top x_i$

Concrete full example: Linear regression

$$y = w^\top x$$

- Task: predict y_i from x_i using $w^\top x_i$
- Performance: mean squared error

$$\begin{aligned} \text{MSE} &:= \frac{1}{n}\sum_i (y_i - w^\top x_i)^2 \\ &= \frac{1}{n}\sum_i (y_i - \hat{y}_i)^2 \end{aligned}$$

- Experience: $\{(x, y)\}$

How do you find optimal parameters?

- Optimize the MSE with respect to the parameters of the model

$$L(\mathbf{w}) = \frac{1}{n}\sum_i (y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

- Take the gradient of the MSE. Set equation to 0 and solve.

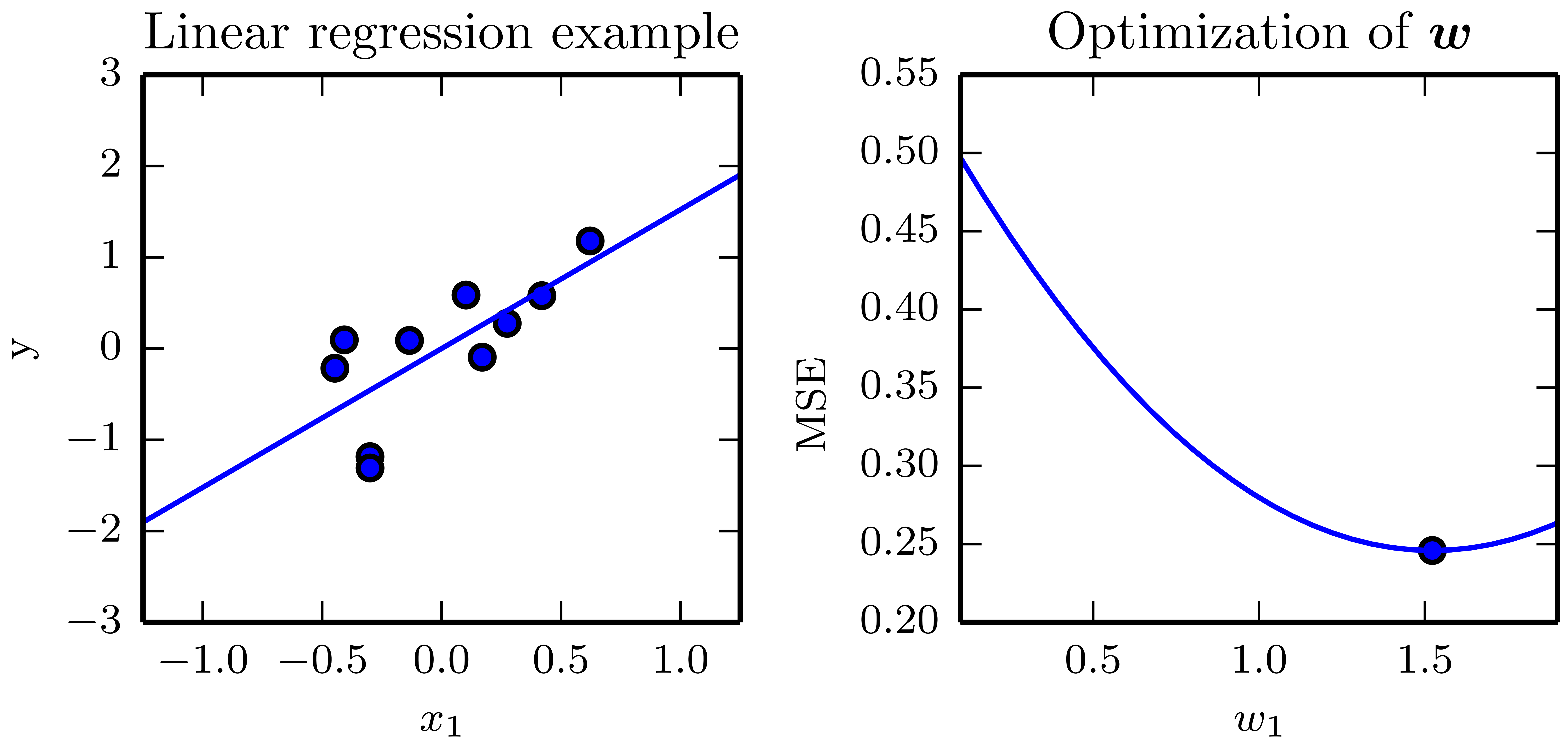
Take gradient of the Loss wrt its parameters

$$\nabla_w L(w) = \nabla_w \frac{1}{n}\sum_i (y_i - w^\top x_i)^2$$
$$= \frac{1}{n}\sum_i \nabla_w (y_i - w^\top x_i)^2$$
$$= \frac{1}{n}\sum_i 2(y_i - w^\top x_i)(-x_i)$$

Set gradient to 0.

$$0 = \frac{1}{n}\sum_i (-2x_i y_i + 2(w^\top x_i)x_i)$$
$$\Rightarrow \sum_i 2(w^\top x_i)x_i = \sum_i 2x_i y_i$$
$$\Rightarrow w(X^\top X) = X^\top Y$$
$$\Rightarrow w = (X^\top X)^{-1} X^\top Y$$

$\mathbf{y} = \mathbf{w_1 x_1}$



[Figure 5.1, Chapter 5, Deep Learning]

Model evaluation

The goal of ML

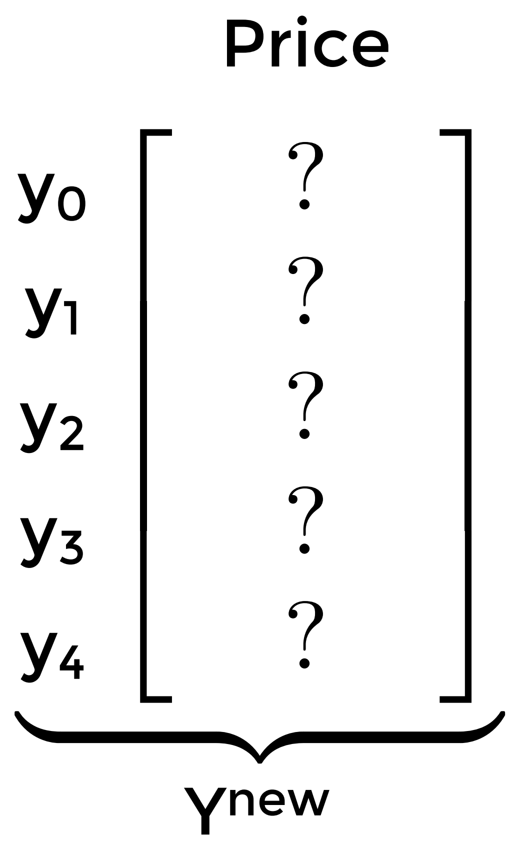
- Predict on new inputs (X^{new})
  - Given the price of houses this year (X, Y), I want to predict the price of houses next year (X^{new})

$$\underbrace{\begin{matrix} & \text{Nb.bed.} & \text{Area} & \text{Neigh.} & \cdot & \cdot \\ x_0 & 1 & 0 & 0 & 0 & 0 \\ x_1 & 1 & 100 & 1 & .2 & .5 \\ x_2 & 3 & 200 & 0 & .1 & .2 \\ x_3 & 1 & 150 & 1 & .4 & .1 \\ x_4 & 2 & 210 & 2 & .5 & 1.1 \end{matrix}}_{X} \quad \underbrace{\begin{matrix} & \text{Price} \\ y_0 & 125000 \\ y_1 & 150000 \\ y_2 & 350000 \\ y_3 & 275000 \\ y_4 & 225000 \end{matrix}}_{Y}$$

$$\underbrace{\begin{matrix} & \text{Nb.bed.} & \text{Area} & \text{Neigh.} & \cdot & \cdot \\ x_0 & 1 & 0 & 0 & 0 & 0 \\ x_1 & 2 & 50 & 1 & .3 & .8 \\ x_2 & 1 & 100 & 1 & .5 & 1.4 \\ x_3 & 4 & 170 & 0 & .7 & .4 \\ x_4 & 1 & 120 & 3 & .9 & .5 \end{matrix}}_{X^{new}} \quad \underbrace{\begin{matrix} & \text{Price} \\ y_0 & ? \\ y_1 & ? \\ y_2 & ? \\ y_3 & ? \\ y_4 & ? \end{matrix}}_{Y^{new}}$$

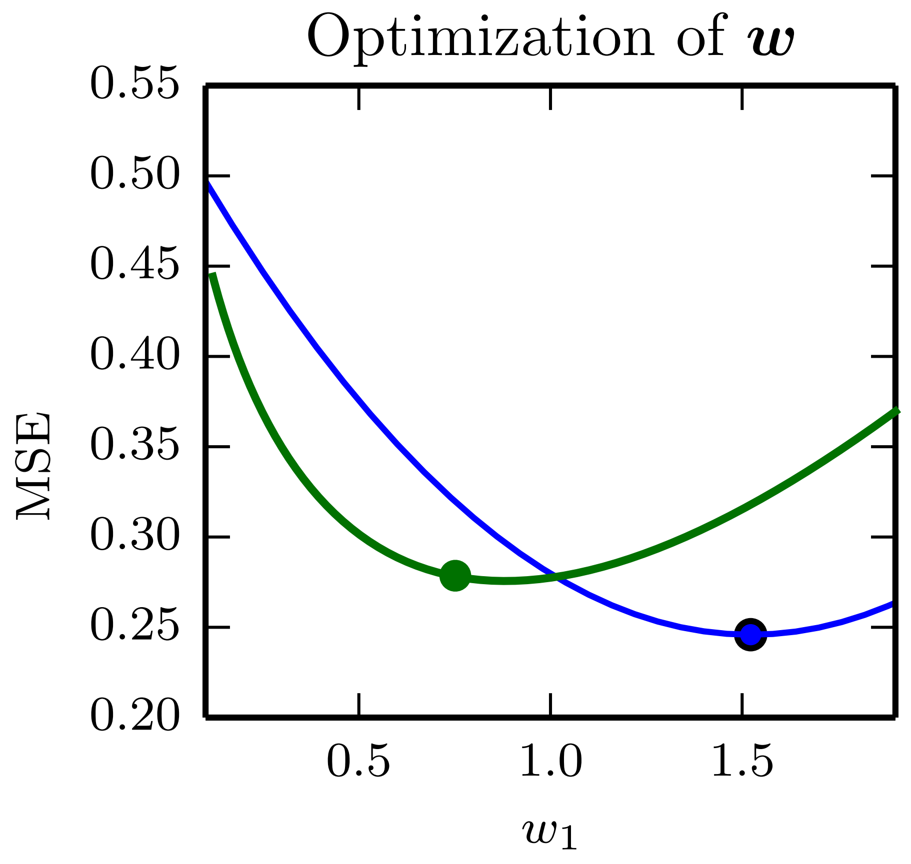
- We can use our estimated ($\widehat{w}$): $Y^{new} = \widehat{w}^\top X^{new}$

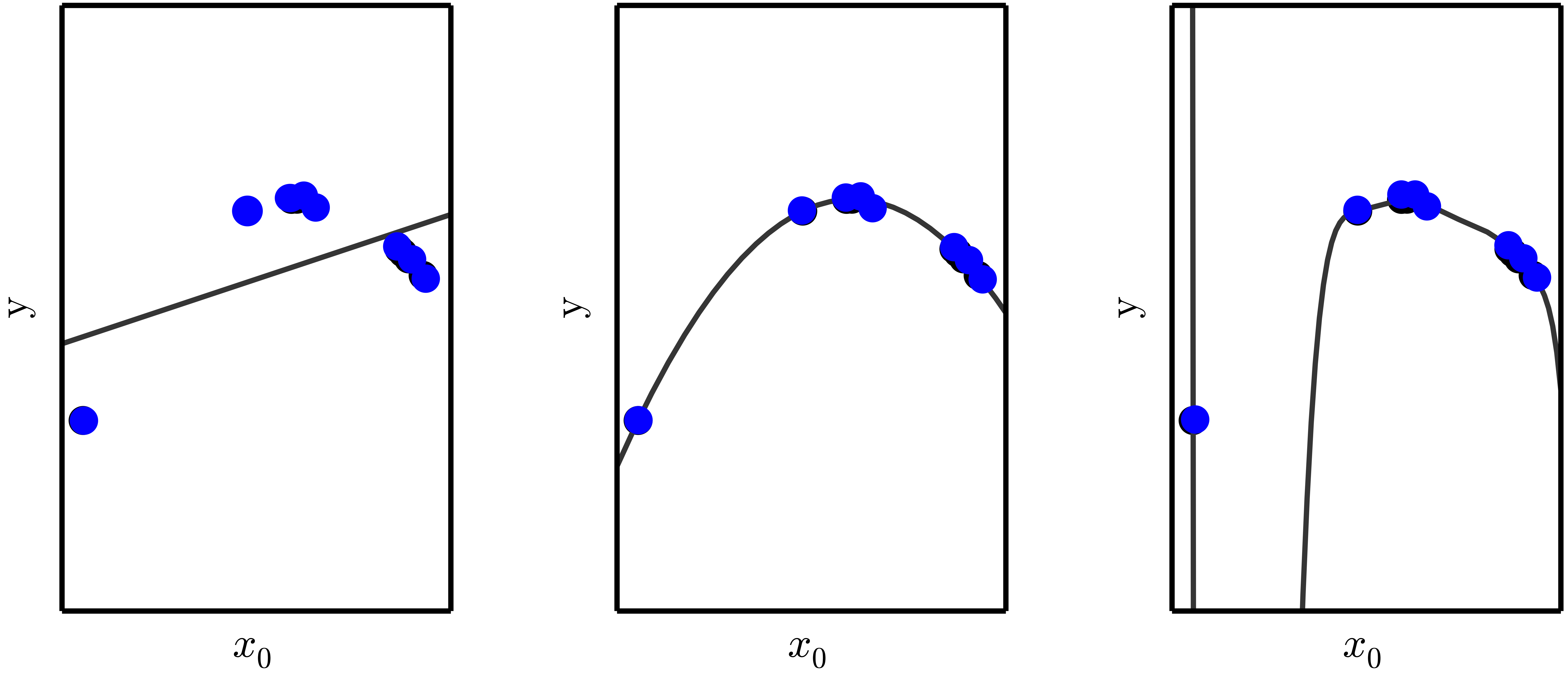
Generalization



- $\text{Loss}^{(X,Y)}$: The one you can evaluate
- $\text{Loss}^{(X^{new},Y^{new})}$: The one that you care about
- In general minimizing the former will not yield the best loss on the latter:

$$\arg\min_{\mathbf{w}} \text{Loss}^{(X,Y)} \neq \arg\min_{\mathbf{w}'} \text{Loss}^{(X^{new},Y^{new})}$$

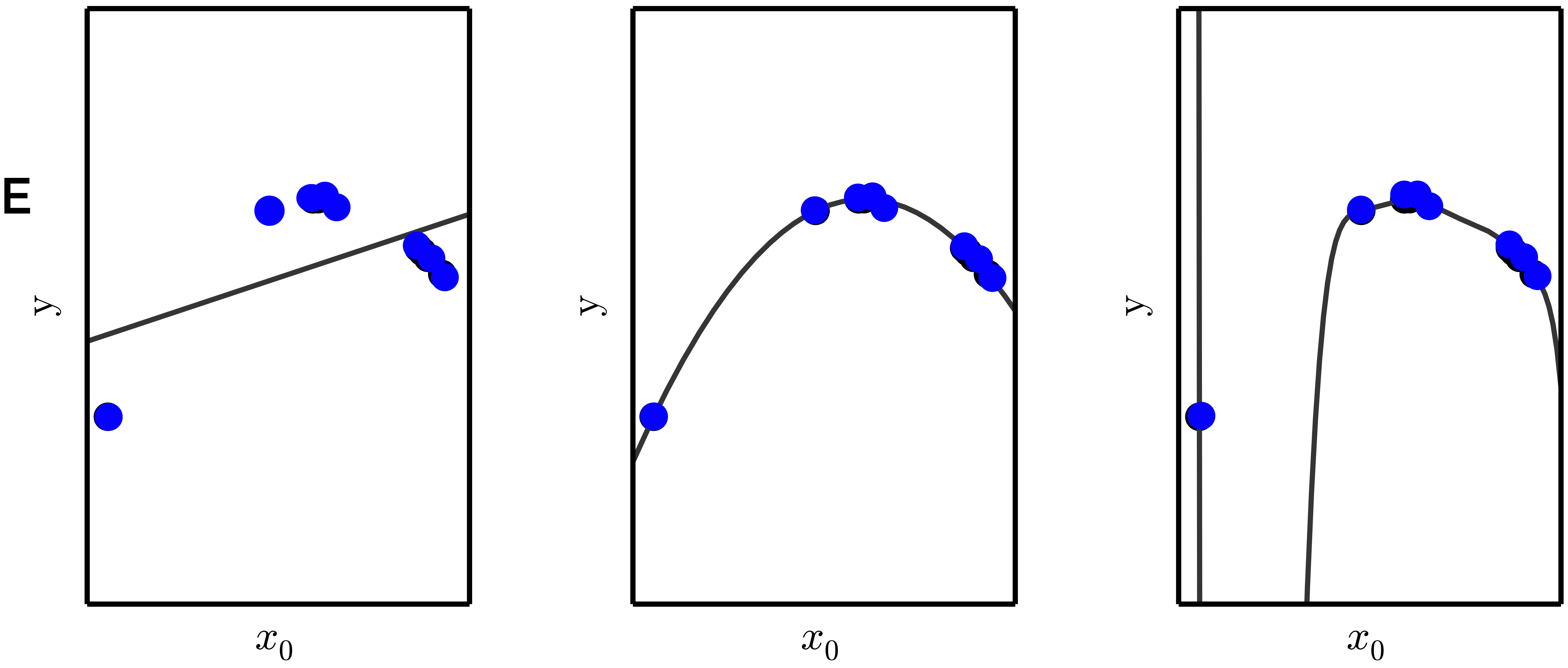




[Figure 5.2, Chapter 5, Deep Learning]

$$\text{Loss}_1^{(X,Y)} > \text{Loss}_2^{(X,Y)} > \text{Loss}_3^{(X,Y)}$$

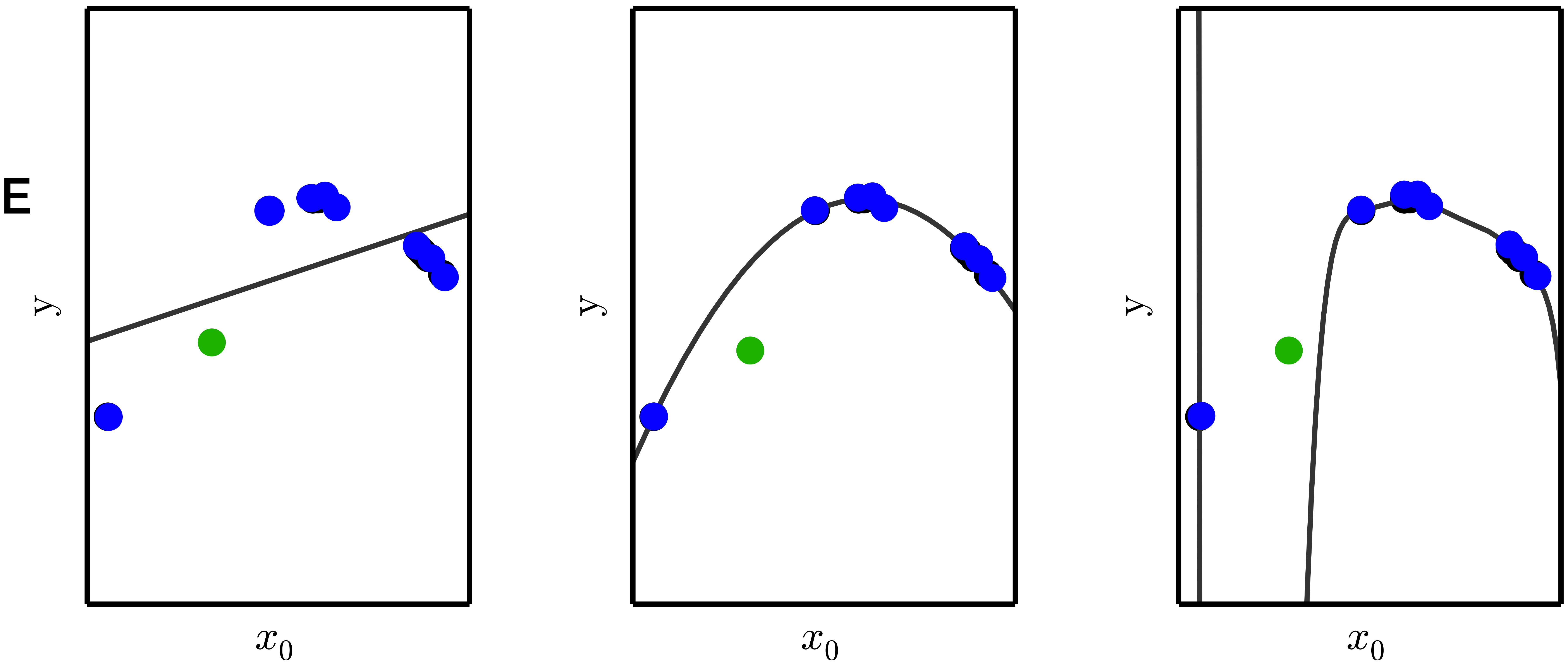
$\text{Loss} \equiv \text{MSE}$



[Figure 5.2, Chapter 5, Deep Learning]

$\text{Loss}_1^{(X,Y)} > \text{Loss}_2^{(X,Y)} > \text{Loss}_3^{(X,Y)}$

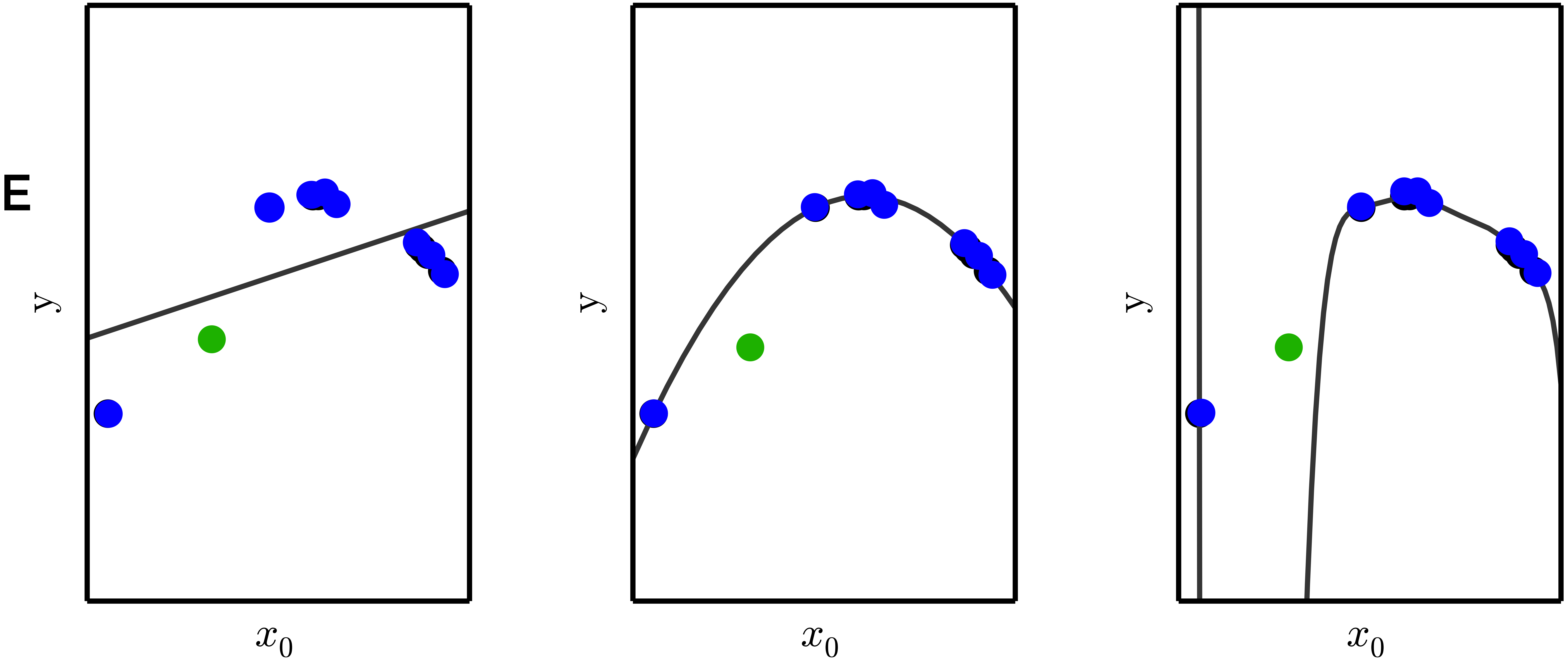
Loss $\equiv$ MSE



[Figure 5.2, Chapter 5, Deep Learning]

$$\text{Loss}_1^{(X,Y)} > \text{Loss}_2^{(X,Y)} > \text{Loss}_3^{(X,Y)}$$

$\text{Loss} \equiv \text{MSE}$



$$\text{Loss}_1^{(X^{new},Y^{new})} < \text{Loss}_2^{(X^{new},Y^{new})} < \text{Loss}_3^{(X^{new},Y^{new})}$$

[Figure 5.2, Chapter 5, Deep Learning]

Some Terminology

Some Terminology

- (X,Y) : training set
- (X^{new}, Y^{new}) : test set
- $\text{Loss}^{(X,Y)}$: train loss (error)
- $\text{Loss}^{(X^{new},Y^{new})}$: test/generalization loss (error)

Some Terminology

- (X,Y) : training set
- $(\mathsf{X}^{\text{new}}, \mathsf{Y}^{\text{new}})$: test set
- $\text{Loss}^{(\mathsf{X},\mathsf{Y})}$: train loss (error)
- $\text{Loss}^{(\mathsf{X}^{\text{new}},\mathsf{Y}^{\text{new}})}$: test/generalization loss (error)

Our goal in ML is (to train the model) to obtain small generalization error

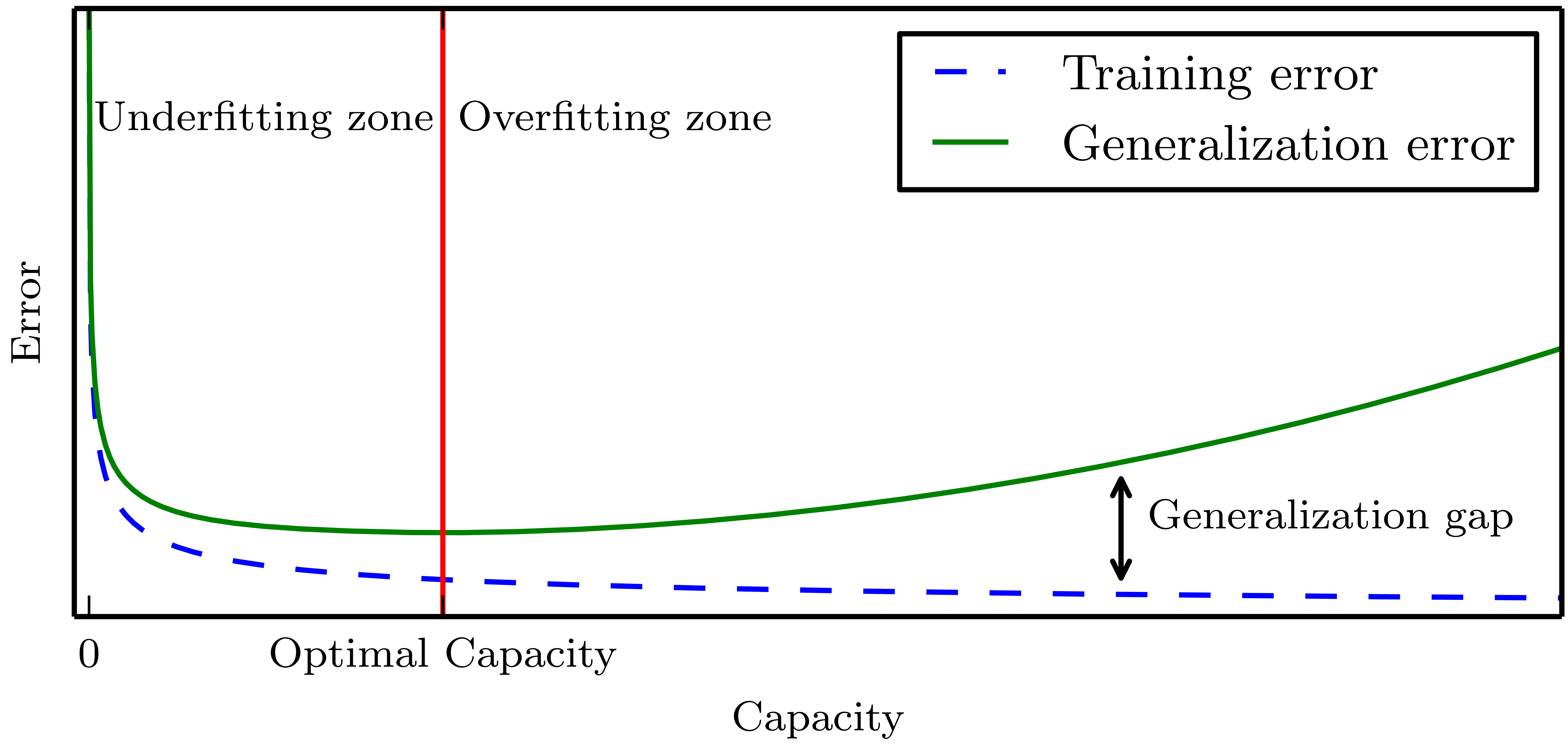
Some Terminology

- (X,Y) : training set
- (X^{new}, Y^{new}) : test set
- $\text{Loss}^{(X,Y)}$: train loss (error)
- $\text{Loss}^{(X^{new}, Y^{new})}$: test/generalization loss (error)

Our goal in ML is (to train the model) to obtain small generalization error

- Capacity: "The ability of a model to fit a variety of functions" [DL]

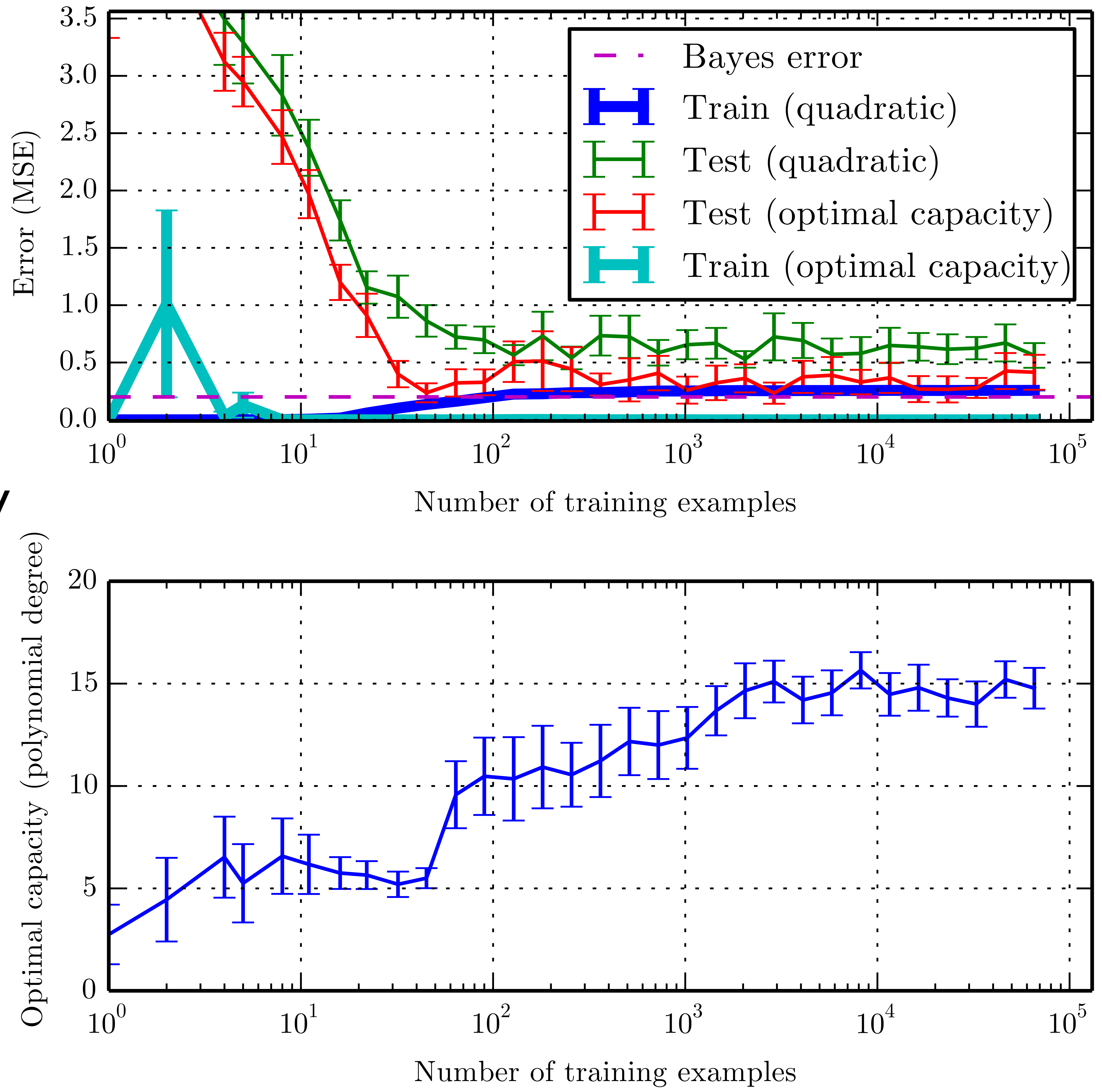
$$\text{capacity}(w_2^\top x^2 + w_1^\top x) > \text{capacity}(w_1^\top x)$$



[Figure 5.3, Chapter 5, Deep Learning]

Synthetic data is generated using a degree 5 polynomial $y = w_5x^5 + w_4x^4 + w_3x^3 + w_2x^2 + w_1x^1$

Training set size also plays an important role in a model's capacity to generalize



[Figure 5.4, Chapter 5, Deep Learning]

Formal learning guarantees

- It is possible to bound the generalization gap
  - Bounds involve:
    - the size of the training set
    - the capacity of the learning model

Informally

- Larger datasets (train) are helpful
    - Allow you to better fit models and/or fit more complex models
- Larger capacity models can be better but (all being equal) they will require more data

Regularization

Regularization

- Can affect a model's effective capacity
  - Instead of changing the model (reminder: polynomials)
  - Focusses on particular (good) solutions

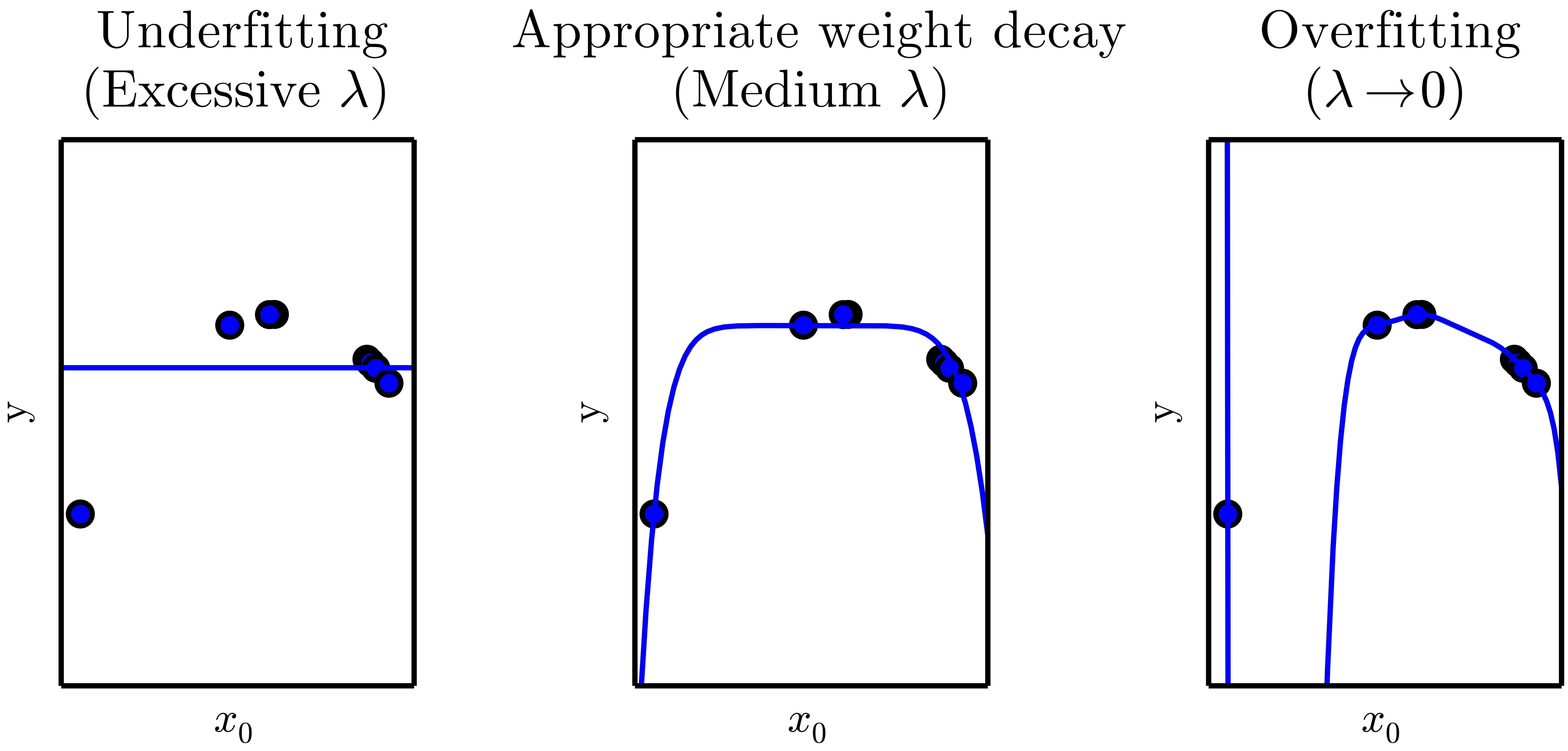
L2-regularization

- A popular form of regularization

$$\text{Loss} := \text{MSE}^{\text{train}} + \lambda \underbrace{\mathbf{w}^\top \mathbf{w}}_{\|\mathbf{w}\|_2}$$

  - Penalizes the size of the weights
    - Smaller weights means simpler models (next slide)

$$\text{Loss} := \text{MSE}^{\text{train}} + \lambda \mathbf{w}^\top \mathbf{w}$$



[Figure 5.5, Chapter 5, Deep Learning]

Validating a model

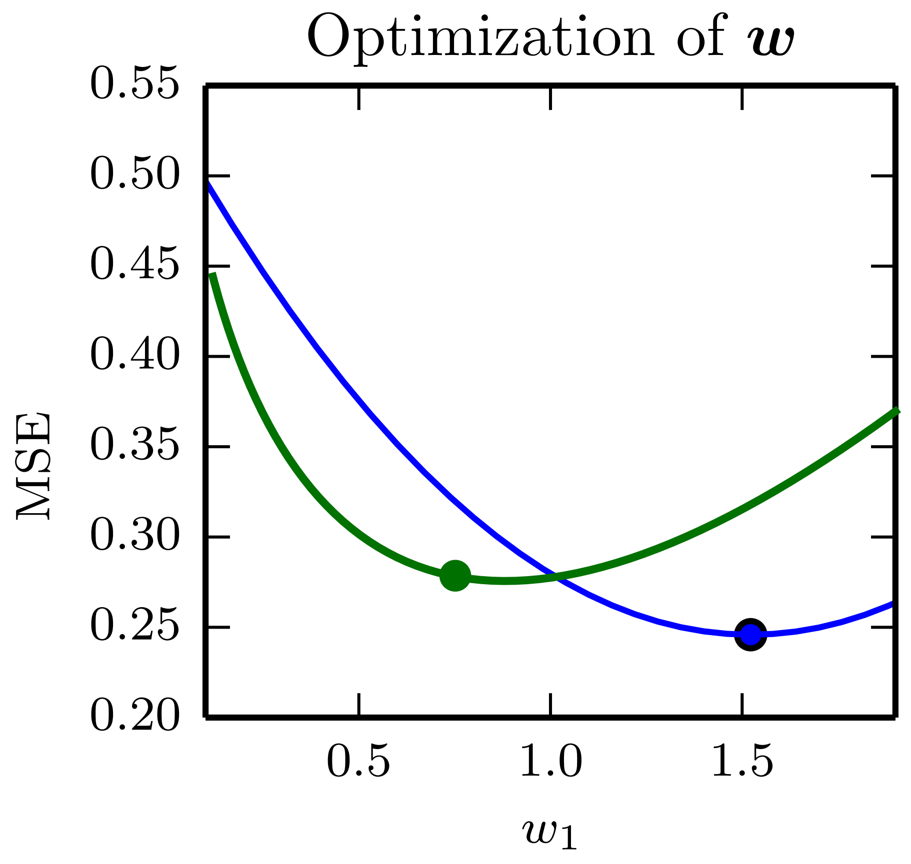
Generalization

Price

$$\underbrace{\begin{matrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{matrix} \begin{bmatrix} ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}}_{Y^{new}}$$

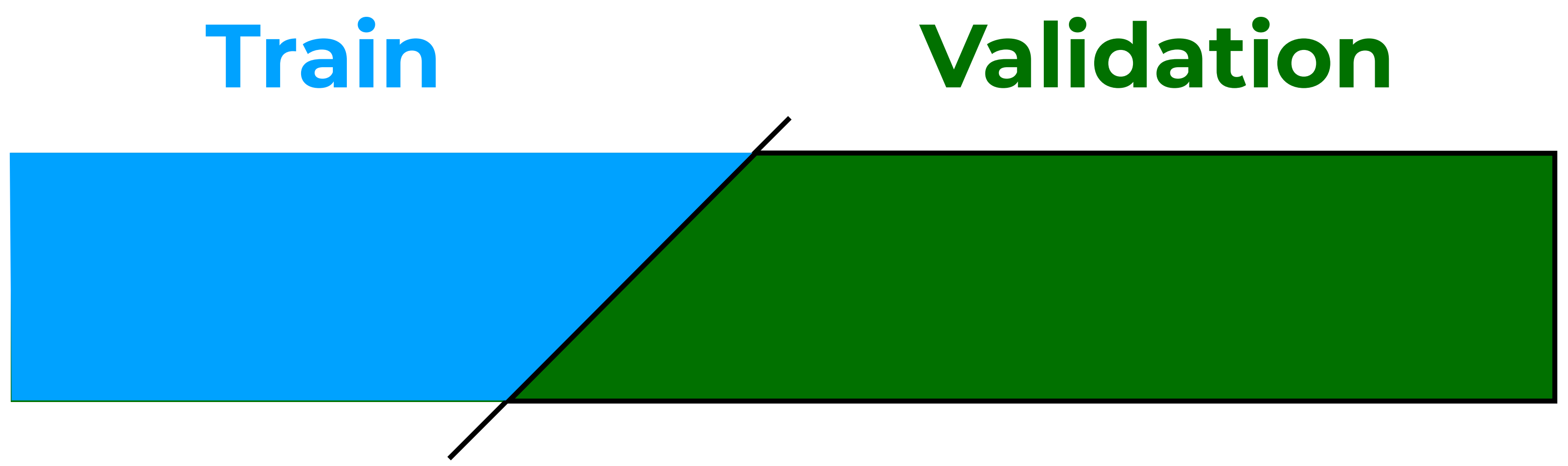
- $\text{Loss}^{(X,Y)}$: The one you can evaluate
- $\text{Loss}^{(X^{new},Y^{new})}$: The one that you care about
- In general minimizing the former will not yield the best loss on the latter:

$$\arg\min_{\mathbf{w}} \text{Loss}^{(X,Y)} \neq \arg\min_{\mathbf{w}'} \text{Loss}^{(X^{new},Y^{new})}$$



Validation set

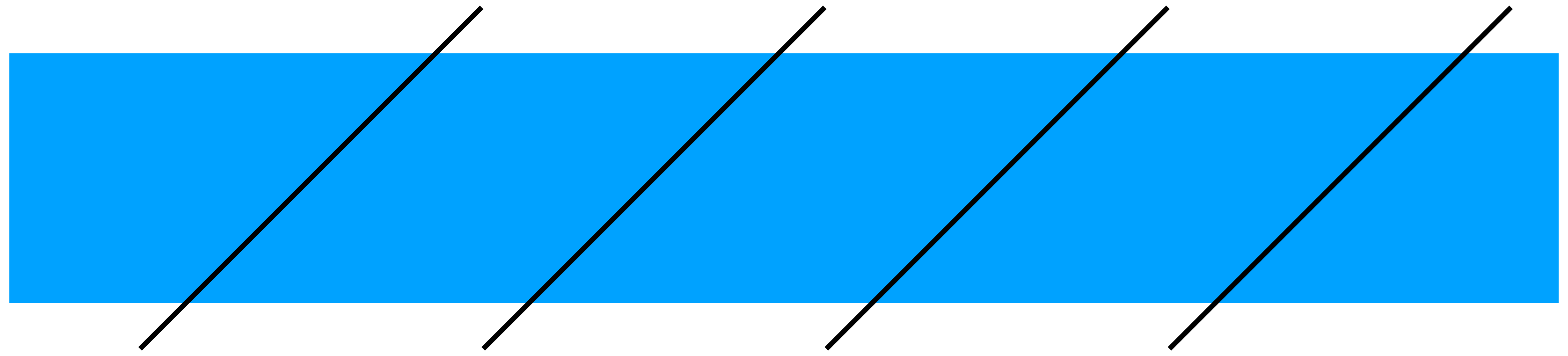
- How do we choose the right model and set its hyper parameters (e.g. λ)?
  - Use a validation set
    - Split the original data into two:
      1. Train set
      2. Validation set
         - Proxy to the test set



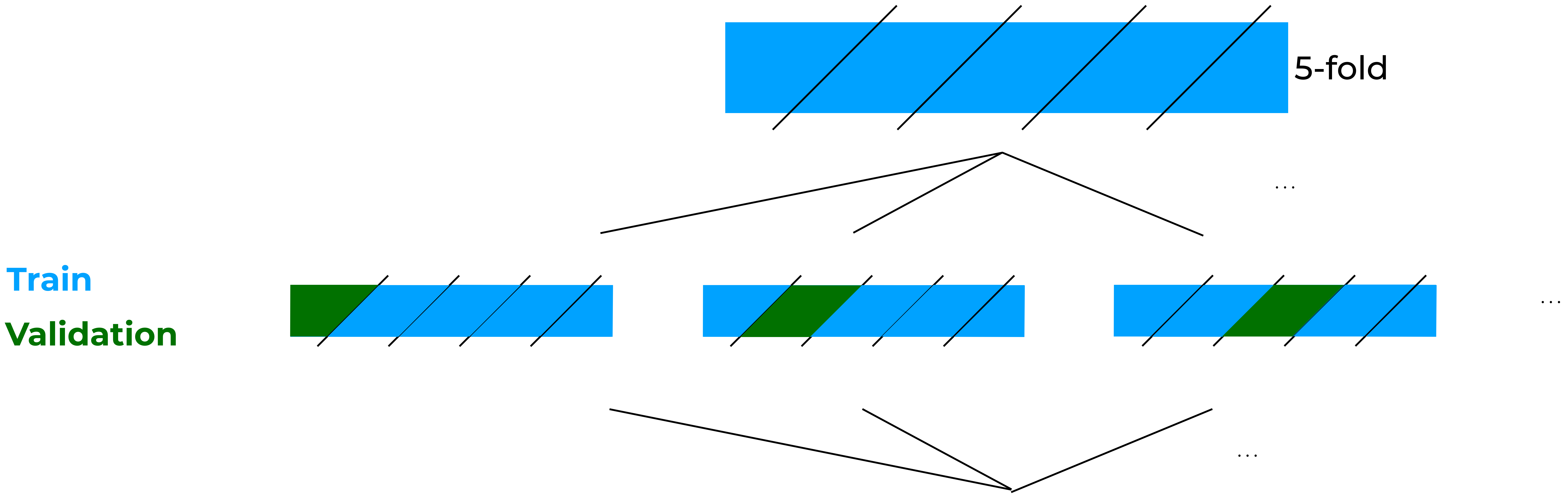
  - Train different models/hyperparameter settings on the train set
  - Pick the best according to their performance on the validation set

Cross-validation (CV)

- Splitting the data into train/validation can be detrimental
    - e.g., if data is small to begin with (small train and validation sets)
- K-fold CV: Split the data into k-folds



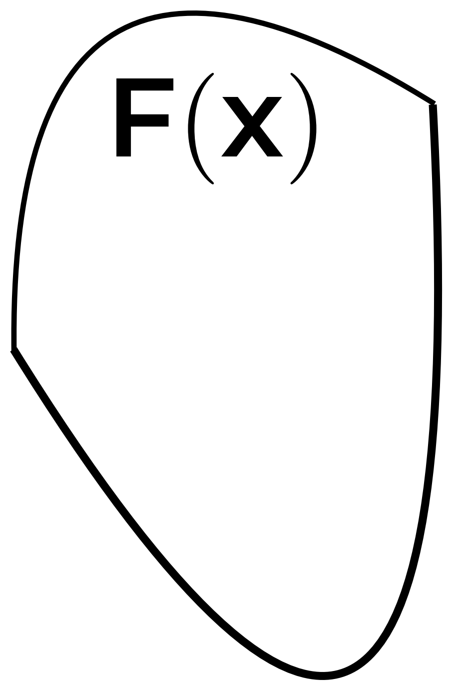
5-fold

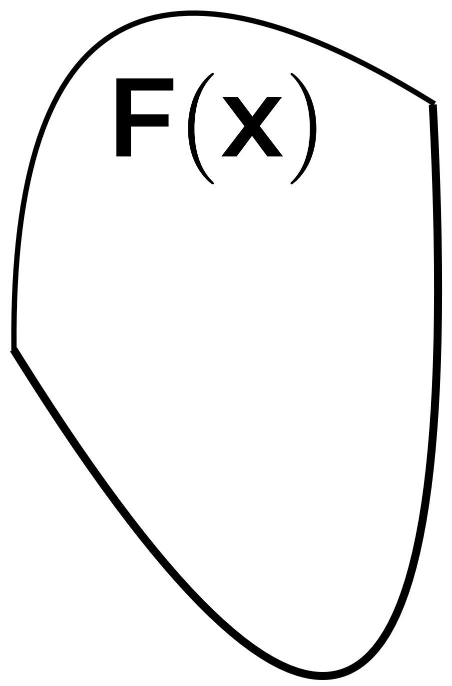


Pick the model/hyperparameters that does best (e.g., smallest loss) according to the average of the validation sets

Bias/Variance:

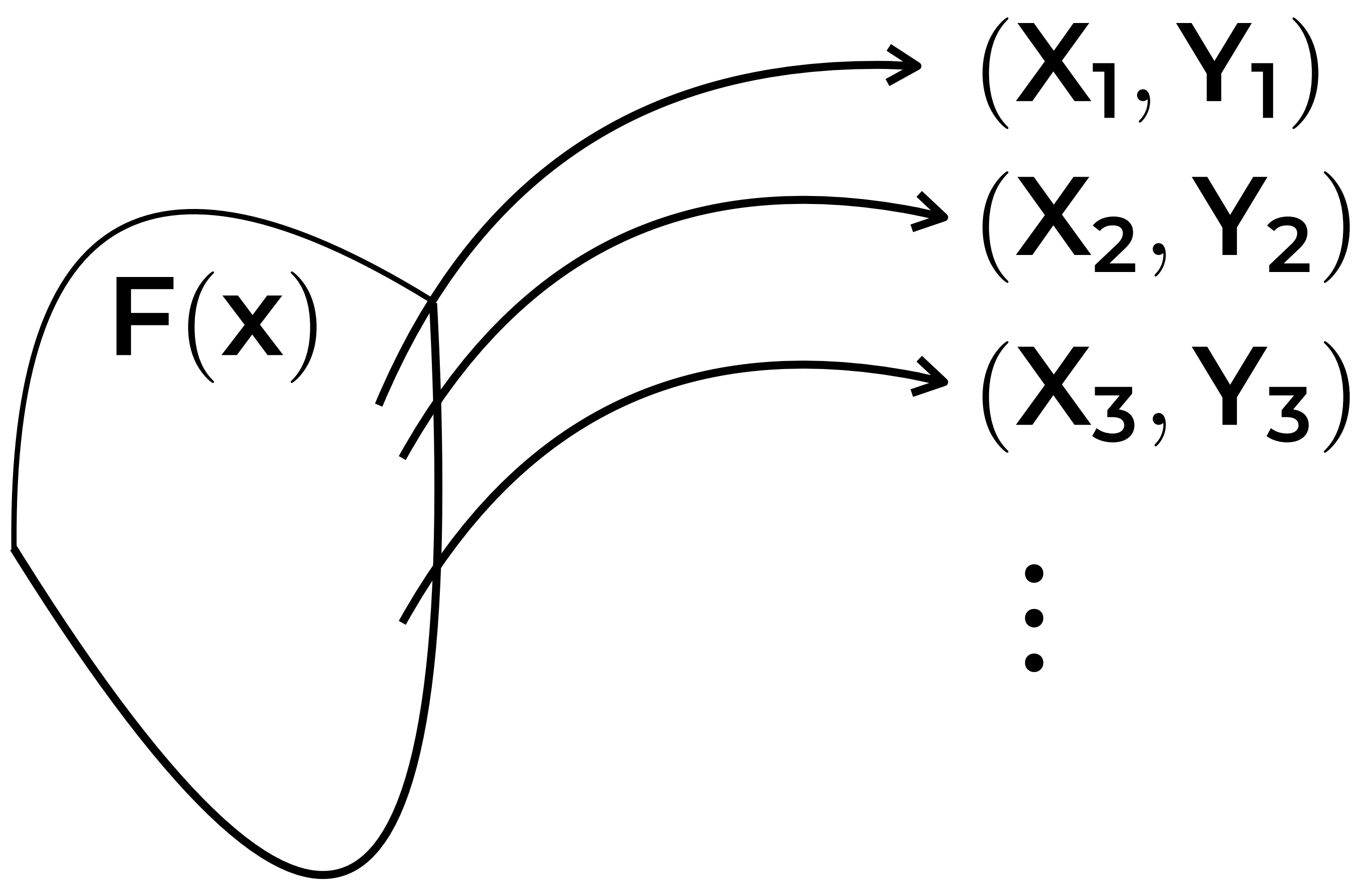
A second perspective on generalization


F(x)



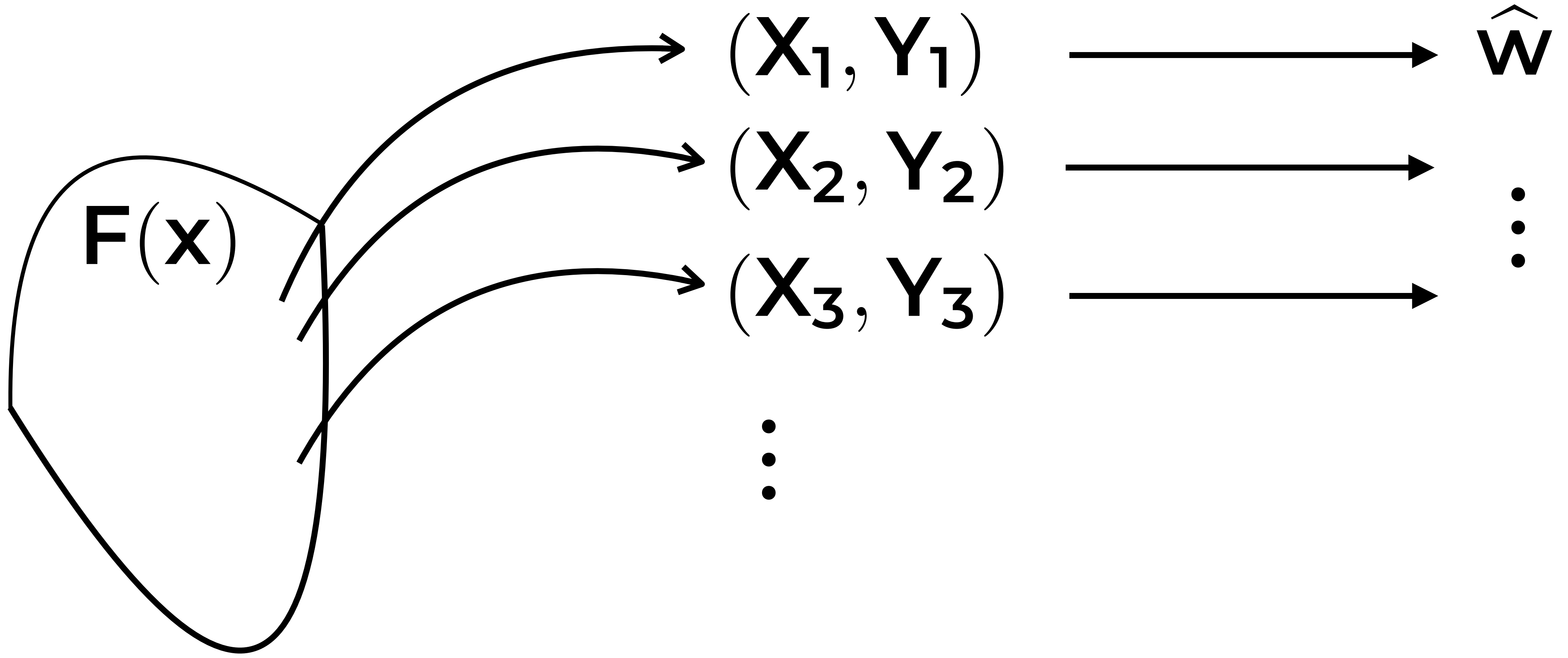
$F(x) =$ linear regression with parameters w^*

$F(x) = w_0^* + w_1^* x + w_2^* x^2 + w_3^* x^3$



$$F(x) = \text{linear regression with parameters } w^*$$

$$F(x) = w_0^* + w_1^* x + w_2^* x^2 + w_3^* x^3$$



$$F(x) = \text{linear regression with parameters } w^*$$

$$F(x) = w_0^* + w_1^* x + w_2^* x^2 + w_3^* x^3$$

Bias/Variance tradeoff (in 4 slides)

- An alternate framework

Bias/Variance tradeoff (in 4 slides)

- An alternate framework

$$\mathbf{w}^* := \text{the value of parameters that generated the data}$$
$$\widehat{\mathbf{w}} := \text{estimator of the true } \mathbf{w}^*$$

Bias/Variance tradeoff (in 4 slides)

- An alternate framework

$$\mathbf{w}^* := \text{the value of parameters that generated the data}$$
$$\widehat{\mathbf{w}} := \text{estimator of the true } \mathbf{w}^*$$

$$\begin{aligned}\text{MSE} &:= \mathbb{E}[(\widehat{\mathbf{w}} - \mathbf{w}^*)^2] \\ &= \text{Bias}(\widehat{\mathbf{w}})^2 + \text{Var}(\widehat{\mathbf{w}})\end{aligned}$$

Bias/Variance tradeoff (in 4 slides)

- An alternate framework

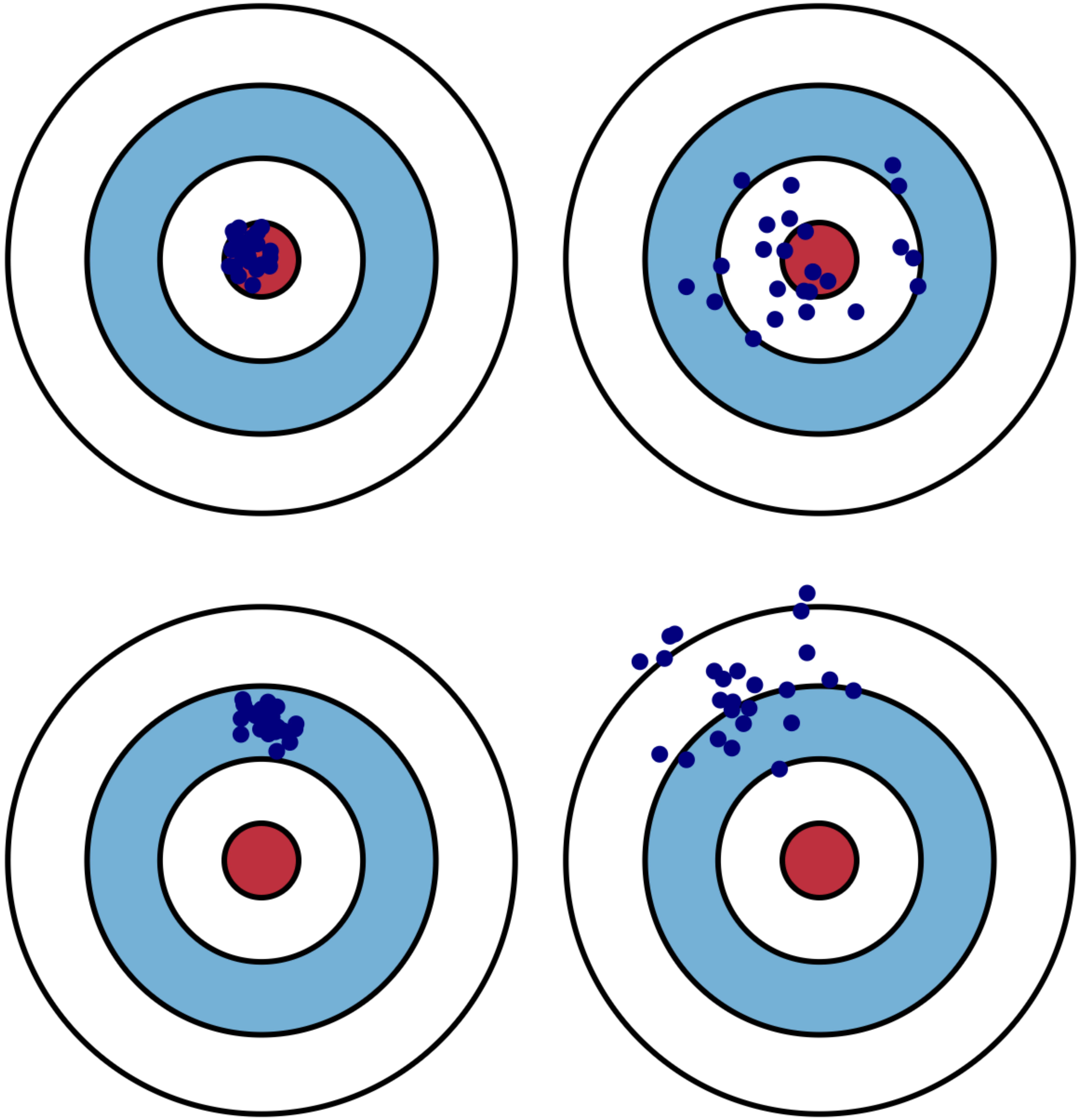
$\mathbf{w}^* :=$ the value of parameters that generated the data
$\widehat{\mathbf{w}} :=$ estimator of the true $\mathbf{w}^*$

$$\begin{aligned}\text{MSE} &:= \mathbb{E}[(\widehat{\mathbf{w}} - \mathbf{w}^*)^2] \\ &= \text{Bias}(\widehat{\mathbf{w}})^2 + \text{Var}(\widehat{\mathbf{w}})\end{aligned}$$

$$\text{Bias}(\widehat{\mathbf{w}}) = \mathbb{E}[\widehat{\mathbf{w}}] - \mathbf{w}^*$$

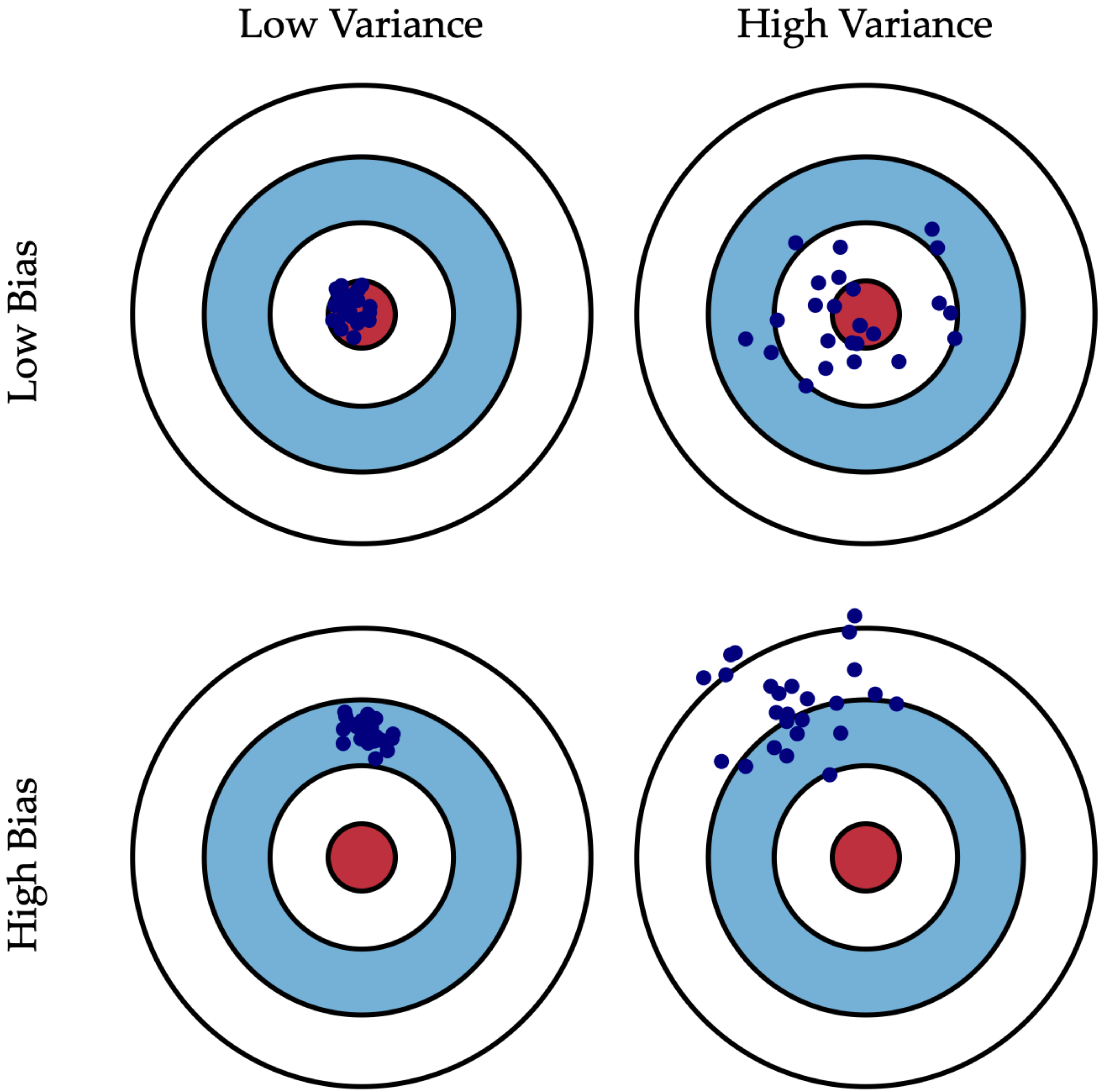
$$\text{Var}(\widehat{\mathbf{w}}) = \mathbb{E}[(\widehat{\mathbf{w}} - \mathbb{E}[\widehat{\mathbf{w}}])^2]$$

- The goal is to hit the bull's eye
- Each blue dot represents the "performance" of a fixed model on different data from the same distribution

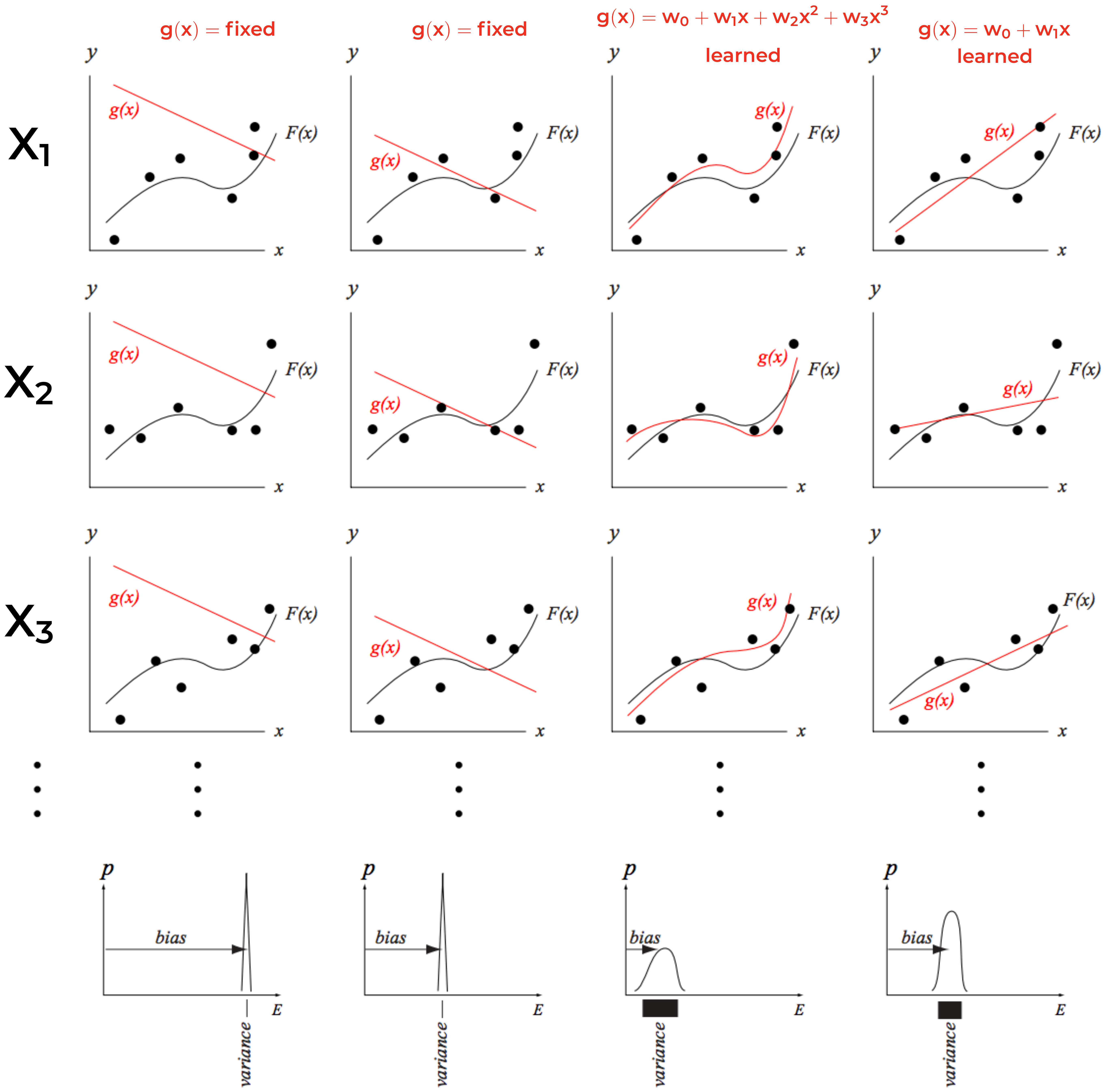


http://scott.fortmann-roe.com/docs/BiasVariance.html

- The goal is to hit the bull’s eye
- Each blue dot represents the “performance” of a fixed model on different data from the same distribution

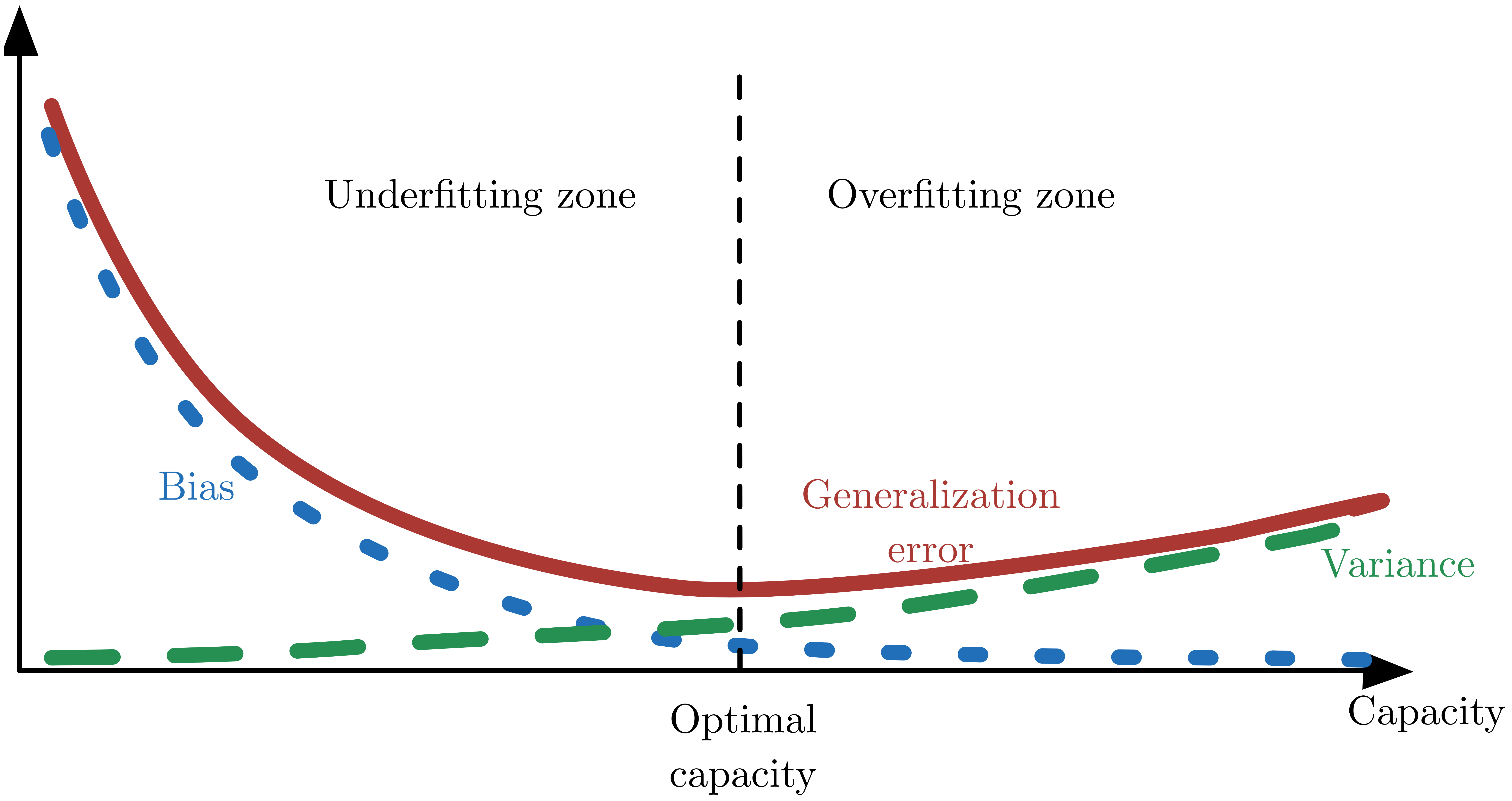


http://scott.fortmann-roe.com/docs/BiasVariance.html



Histogram of the error over training sets

[Figure 9.2. Pattern Classification. Duda, Hart, Stork. 2001]


Underfitting zone
Overfitting zone
Bias
Generalization
error
Variance
Optimal
capacity
Capacity

Other frameworks

- Bayesian
  - Uncertainty indicates you degree of belief
  - Unknown quantities are random variables

Other evaluations

- Is test set evaluation enough?
  - The test error may be a proxy for what you are really trying to evaluate
  - You model may be used inside a larger system
  - How can you convince that an X % improvement in test error is meaningful?

Other evaluations

- Model exploration
  - Are the parameter values it has learned sensible?
  - Plot the residuals
  - Dive into your model’s predictions
    - Where does it do better/worse than others?
- Model criticism
  - How do generated data from your fitted model look like?