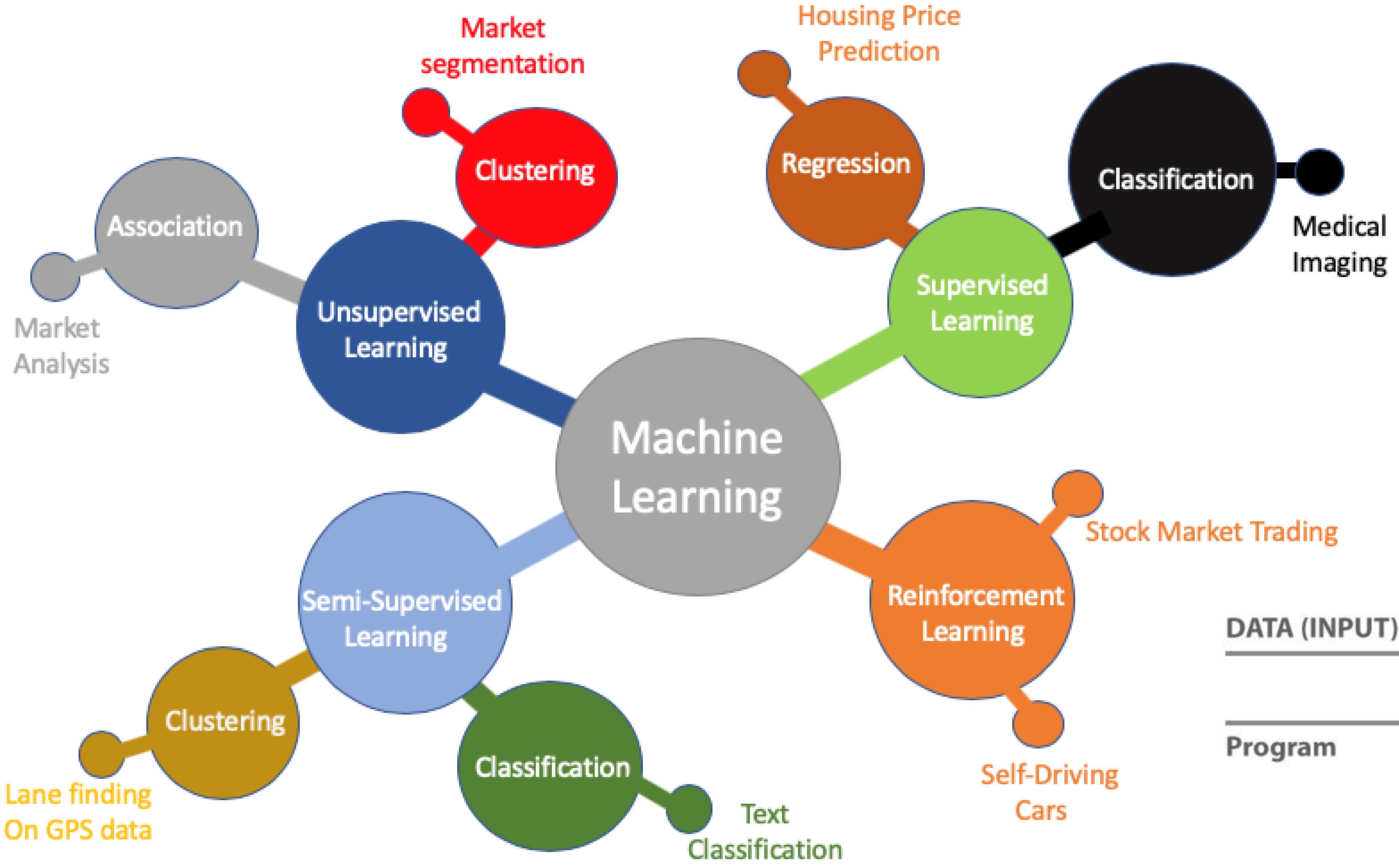
Machine Learning Fundamentals

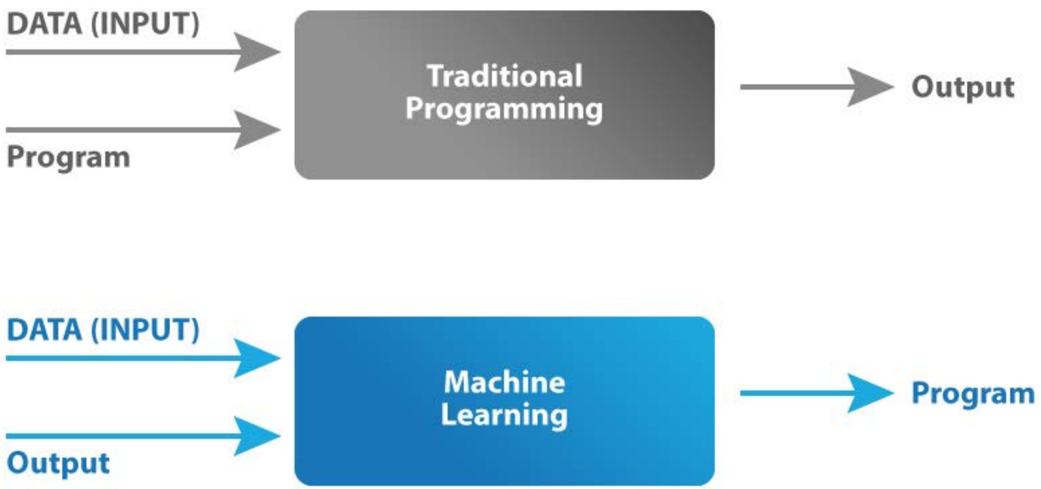
The ability to learn is a core artefact of intelligence

COURSE: CS60045

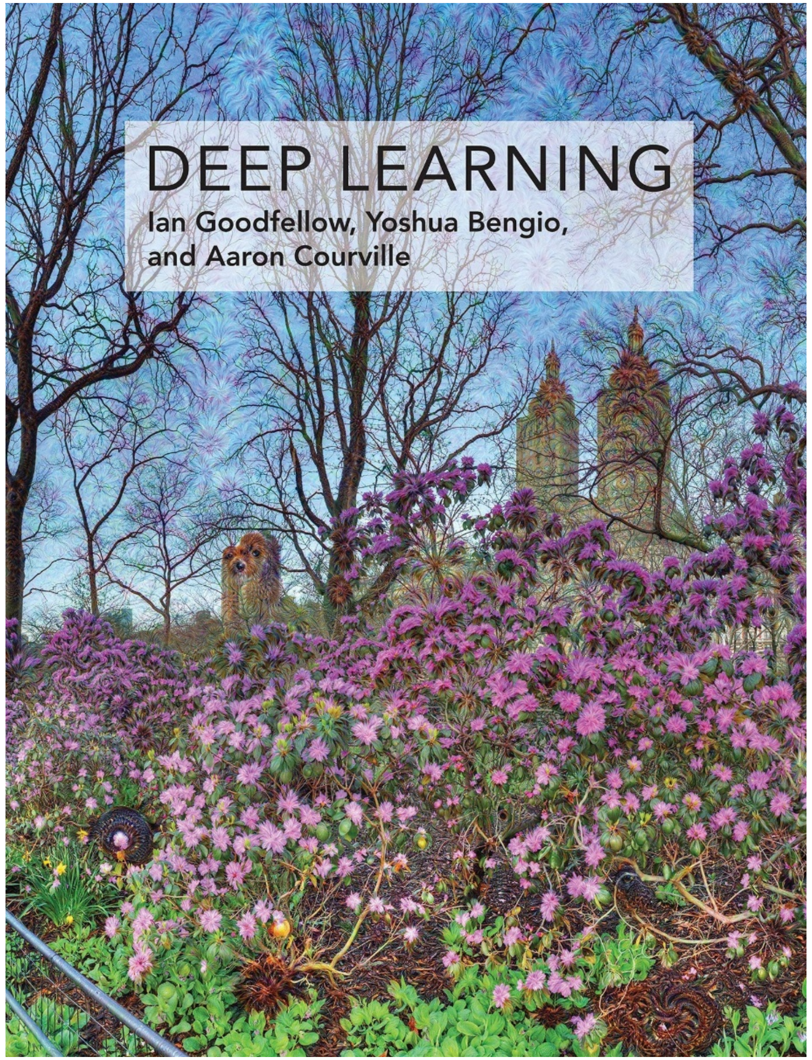
Pallab Dasgupta
Professor,
Dept. of Computer Sc & Engg

Machine Learning





Reference



Deep Learning
Ian Goodfellow, Yoshua Bengio,
and Aaron Courville
MIT Press, 2018

What is machine learning?

- [Mitchell 1997] *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, as measured by P, improves with experience E.*
- *Examples of the task, T*
  - Classification – to learn a function $\boldsymbol{f}\colon\ \mathbb{R}^{\boldsymbol{n}} \rightarrow \{\mathbf{1}, \ldots, \boldsymbol{k}\}$
  - Classification with missing inputs
  - Regression – to learn a function $\boldsymbol{f}\colon\ \mathbb{R}^{\boldsymbol{n}} \rightarrow \mathbb{R}$ (prediction of weather, temperature, stock market)
  - Transcription – for example, learning speech to text or image to text conversion
  - Machine translation – for example, learning to translate English sentences to German
  - Structured output – for example, learning to mark the roads in an aerially captured map
  - Anomaly detection
  - Denoising
  - Density estimation – to learn a function $\boldsymbol{p_{model}}\colon\ \mathbb{R}^{\boldsymbol{n}} \rightarrow \mathbb{R}$ where $\boldsymbol{p_{model}}(\boldsymbol{x})$ can be interpreted as a probability density function

What is machine learning?

- [Mitchell 1997] *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, as measured by P, improves with experience E.*
- *Performance measure, P*
  - For tasks like transcription and classification we often measure the *accuracy* of the model
    - Accuracy = the proportion of examples for which the model produces the correct output
    - Error rate = the proportion of examples for which the model produces incorrect output
  - Data = { Training set, Test set }
    - The model is trained using the training set
    - Performance is assessed using the test set
  - Deciding the performance measure is not always very straight forward
    - For example, when performing a regression task, should we penalize the system more if it frequently makes medium sized mistakes or if it rarely makes very large mistakes

What is machine learning?

- [Mitchell 1997] *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, as measured by P, improves with experience E.*
- *The Experience, E*
  - Unsupervised Learning Algorithms experience a dataset containing many features, then learn useful properties of the structure of the dataset (such as anomaly detection, denoising, etc)
    - Learning the probability distribution *p*(x) by observing several examples of a random vector x.
  - Supervised Learning Algorithms experience a dataset containing features, but each example is also associated with a label or target
    - Observing several examples of a random vector x and an associated value or vector y, and then estimating *p*(y | x)

Unsupervised and Supervised Learning are not formally different

- The chain rule of probability states that for a vector $\boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}}$, the joint probability distribution can be decomposed as:

$$\boldsymbol{p}(\boldsymbol{x}) = \prod_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{p}(\boldsymbol{x}_{\boldsymbol{i}} \mid \boldsymbol{x}_{1}, \ldots, \boldsymbol{x}_{\boldsymbol{i}-1})$$

The decomposition means that we can solve the problem of learning *p*(x) by splitting it into *n* supervised learning problems

- Alternatively we can solve the supervised learning problem *p*(y | x) by using the traditional unsupervised learning technologies to learn the joint distribution *p*(x, y), then inferring:

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

Why would one want machine learning?

- Static, pre-programmed rules might give poor performance
- Difficulties in analytically specifying the system
- Dimensionality of the problem is prohibitive
- ….

It is one tool that might or might not be able to solve the problem at hand!

Linear Regression

Given a training data set $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ where: $\boldsymbol{x}^{(j)} = [\boldsymbol{x}_1^{(j)}\ \boldsymbol{x}_2^{(j)} \dots \boldsymbol{x}_n^{(j)}]^T$ for $\boldsymbol{j} = \boldsymbol{1} \dots \boldsymbol{m}$.

The output is a linear function of the input which predicts the value of a scalar $\boldsymbol{w} \in \mathbb{R}$:

$$\hat{\boldsymbol{y}} = \boldsymbol{w}^T \boldsymbol{x}$$

where $\boldsymbol{w} \in \mathbb{R}^{\boldsymbol{n}}$ is a vector of parameters. We can think of *w* as a set of weights.

Task T: Predict y from x by outputting $\hat{\boldsymbol{y}} = \boldsymbol{w}^T \boldsymbol{x}$

Performance: The mean squared error on the test data (which is different from the training data)

$$\boldsymbol{MSE_{test}} = \frac{\boldsymbol{1}}{\boldsymbol{m}} \sum_{\boldsymbol{i}} \left(\hat{\boldsymbol{y}}^{(\boldsymbol{test})} - \boldsymbol{y}^{(\boldsymbol{test})}\right)_{\boldsymbol{i}}^{\boldsymbol{2}}$$

Linear Regression – *a first cut approach*

Performance: The mean squared error on the test data (which is different from the training data)

$$\boldsymbol{MSE_{test} = \frac{1}{m}\sum_i \left(\hat{y}^{(test)} - y^{(test)}\right)_i^2}$$

… but we are not allowed to use the test data at the time of training.

We can choose *w* such that $\boldsymbol{MSE_{train}}$ is minimized instead of $\boldsymbol{MSE_{test}}$. This can be done by solving:

$$\boldsymbol{\nabla_w MSE_{train} = 0}$$

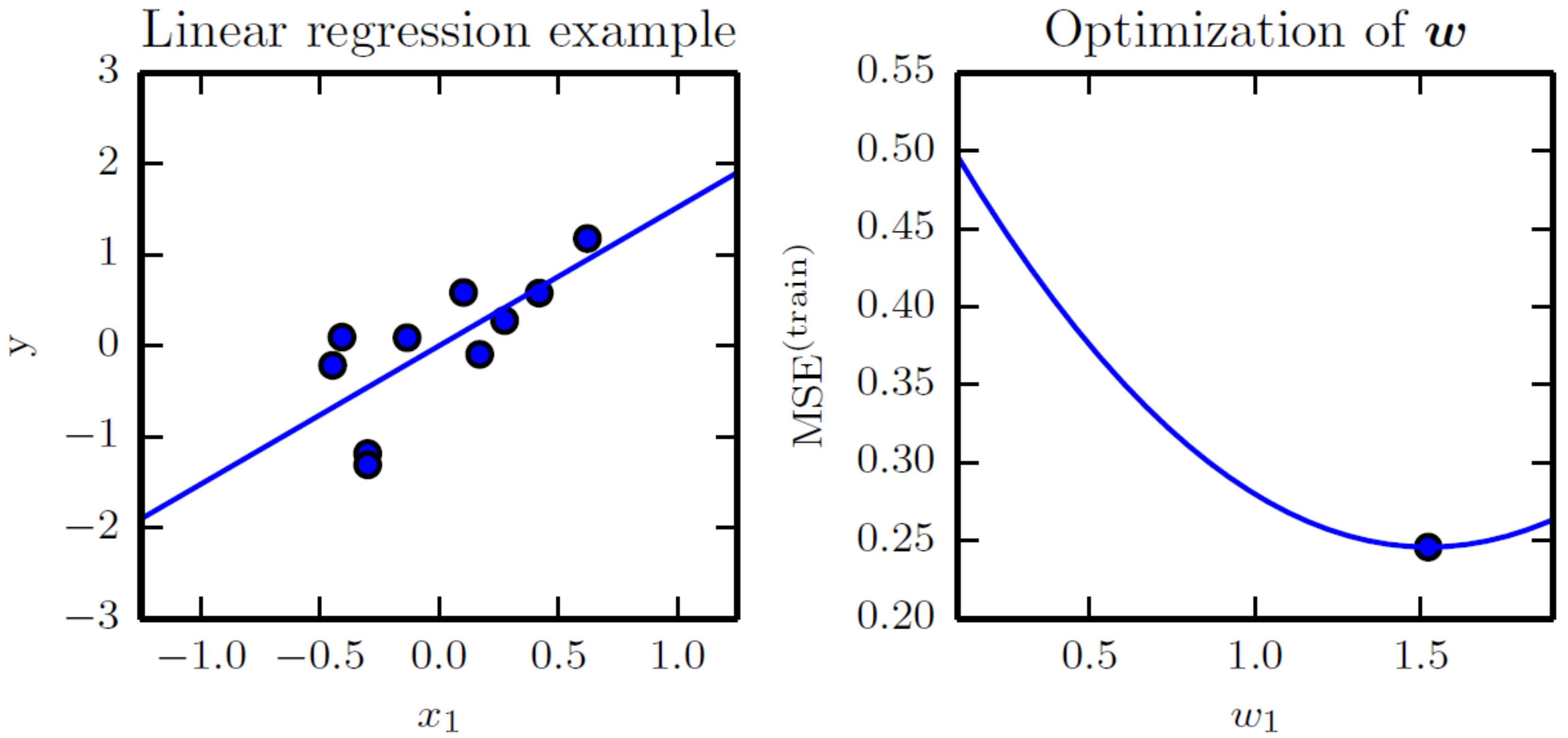
This yields:

$$\boldsymbol{w = \left(X^{(train)\,T} X^{(train)}\right)^{-1} X^{(train)\,T} y^{(train)}}$$

The term *linear regression* is often used to refer to a slightly more sophisticated *affine* model:

$$\boldsymbol{\hat{y} = w^T x + b}$$

Linear regression example



$\text{MSE}^{(\text{train})}$ is minimized for $w_1 = 1.5$

Therefore we learn $y = w_1 x = 1.5x$

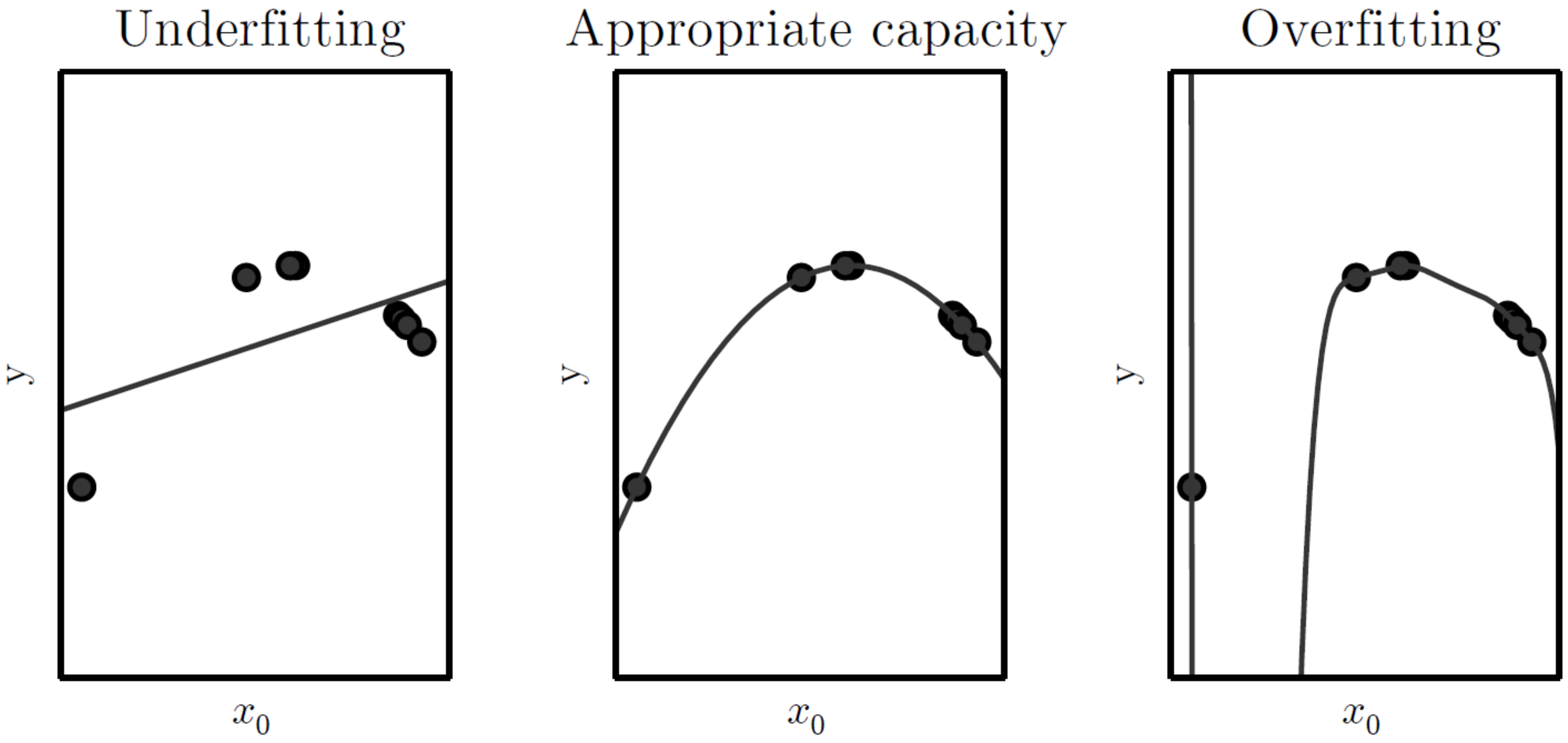
Capacity, Overfitting and Underfitting

- The ability of a machine learning algorithm to perform well on previously unobserved inputs is called *generalization.*
  - Machine learning aims for low generalization error (also called test error)
  - This depends on
    - Accurate choice of the model
    - Quality of the training data

A model’s capacity is its ability to fit a wide variety of functions

- For example, linear regression models cannot fit a sine function
- Low capacity leads to under-fitting – the model is unable to obtain a sufficiently low error on the training data
- High capacity leads to over-fitting – the model learns fictitious functions which results in high test error

Example



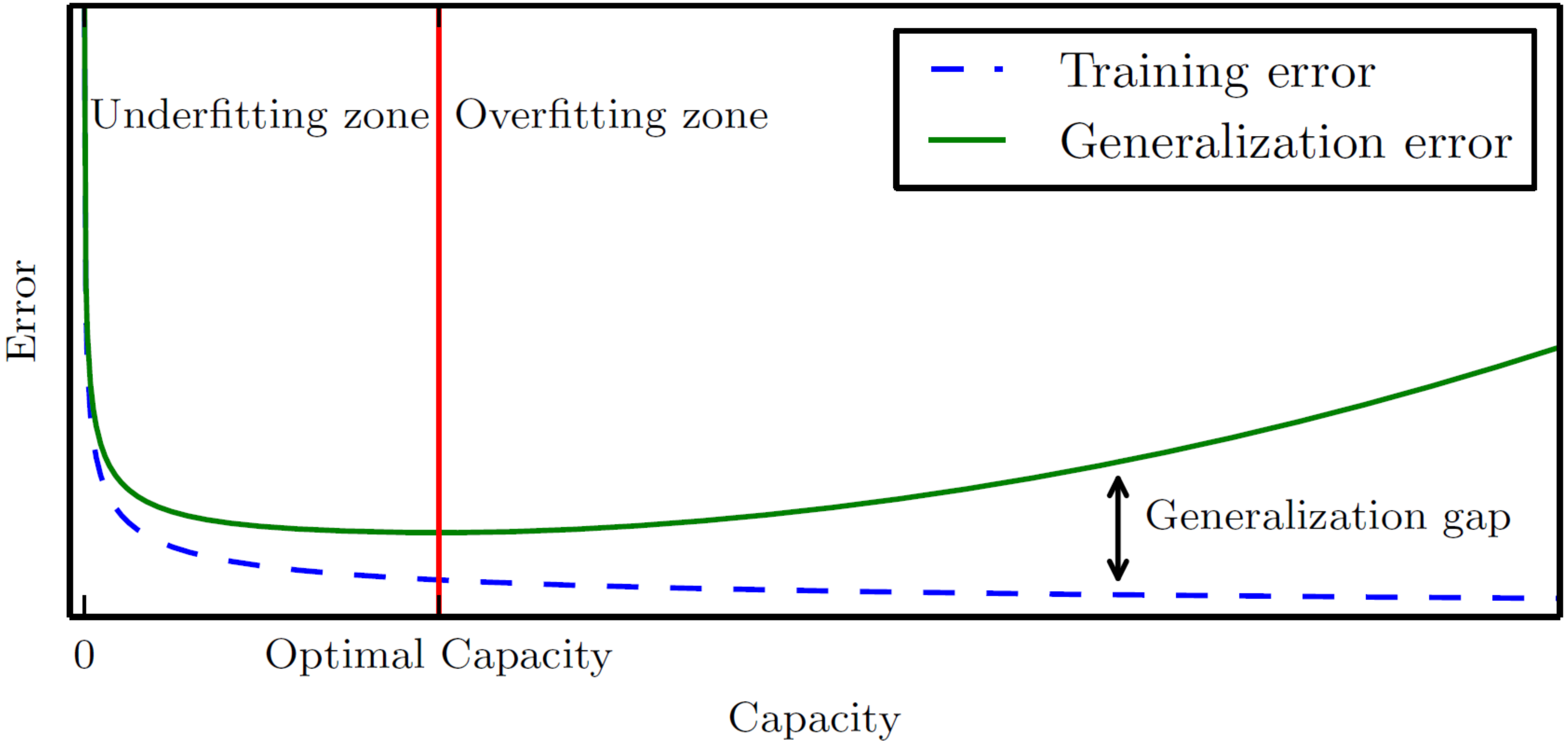
Linear function

Quadratic function

Polynomial of degree 9

- The representational capacity of a model family is the set of functions that we are allowed to select
- The effective capacity of a learning algorithm may be less than the representational capacity of the model family because the algorithm may restrict the hypothesis space
- Occam’s razor: Among competing hypotheses that explain known observations equally well, we should choose the “simplest” one. We will therefore choose the quadratic one in the above.

Capacity versus Generalization Error



The No Free Lunch Theorem

How well can a machine learning algorithm generalize from a finite training set of examples?

The no free lunch theorem for machine learning [Wolper 1996] states:

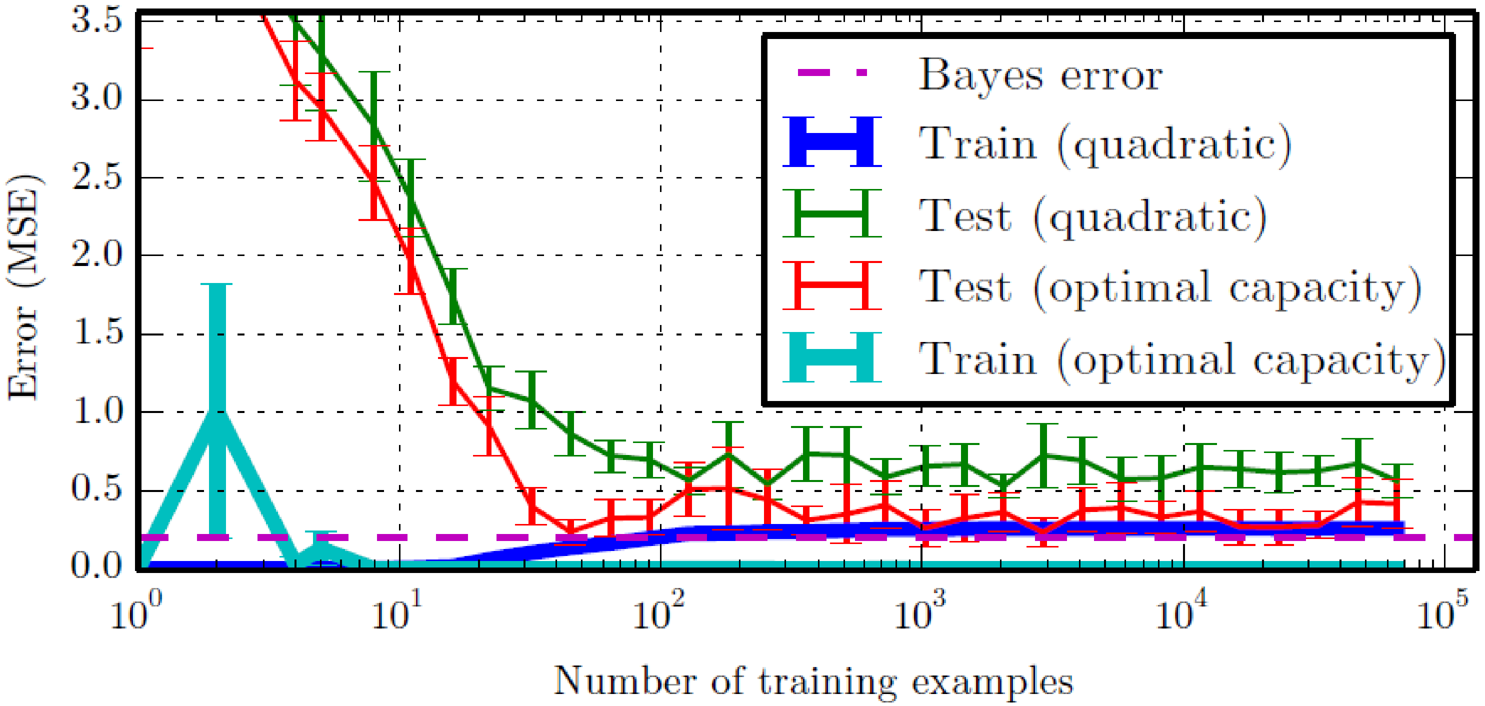
> Averaged over all possible data-generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points.

Therefore in the absence of domain knowledge, no machine learning algorithm is universally any better than any other.

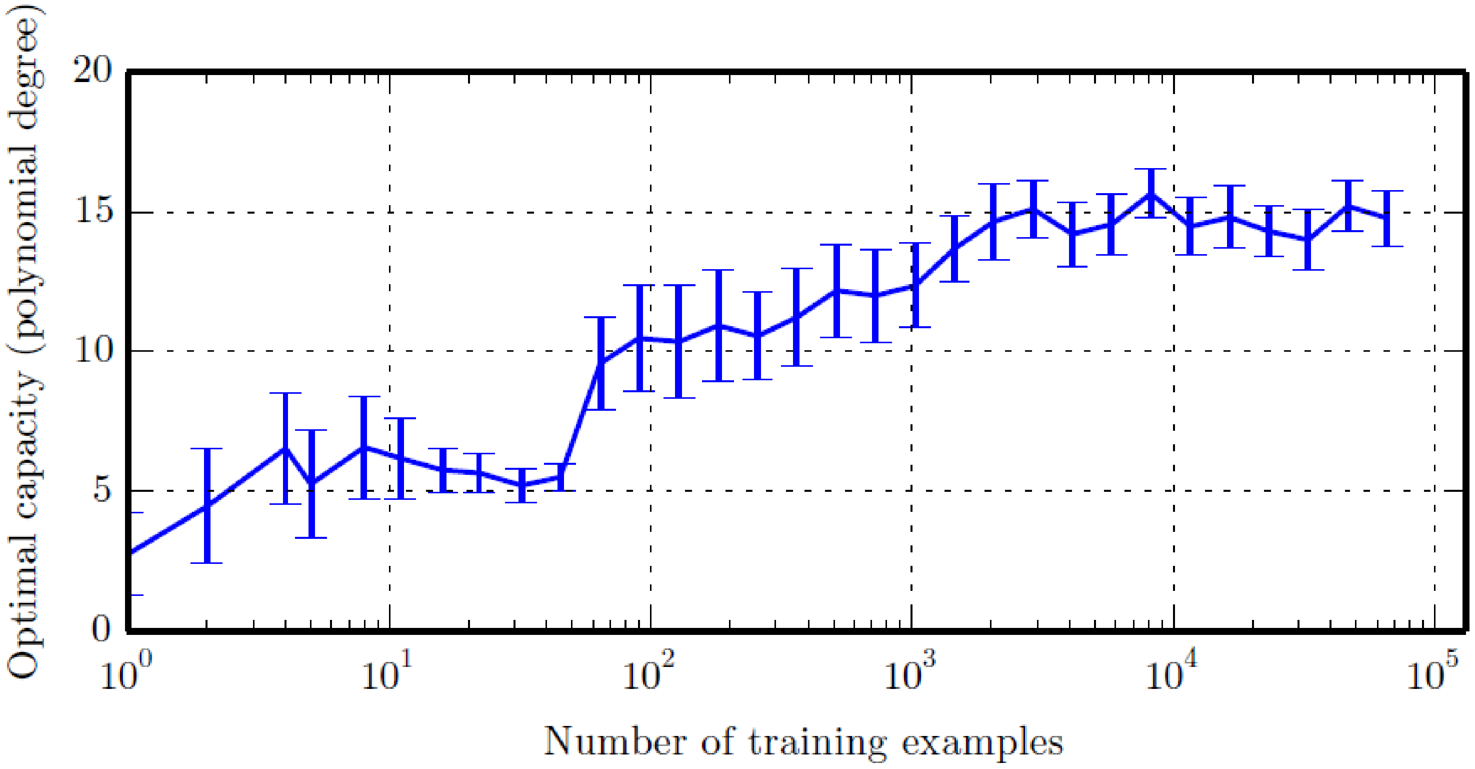
It is therefore important to understand the distributions relevant to the domain for choosing the model family and the learning algorithm.

Effect of training set

The quadratic model has inadequate capacity, hence training error increases with number of training examples. Also the test error asymptotes to a high value.



The optimal capacity plateaus after reaching sufficient complexity to solve the task



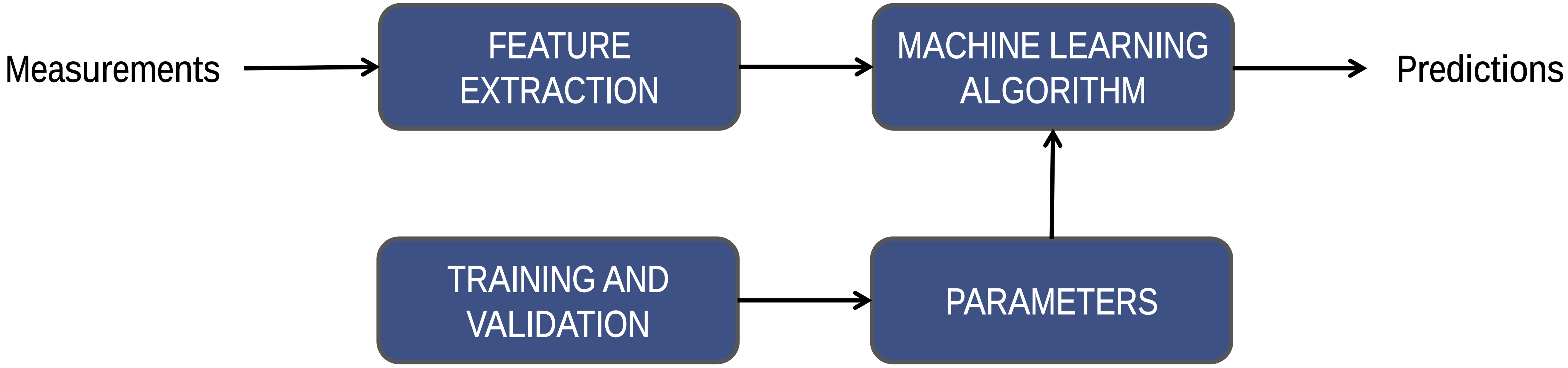
Hyper-parameters and Validation Sets

- Most machine learning algorithms have hyper-parameters or settings that we can tune to control the algorithm’s behavior
  - For example, in regression the degree of the polynomial acts as a *capacity hyper-parameter*

- Hyper-parameters should not be learned from the entire training set (why?)
  - If learned on the training set, such hyper-parameters would always choose the maximum possible model capacity
  - The test set must also not be used to learn the hyper-parameters

- Typically one uses about 80% of the training data for training and 20% for validation. This subset is called the validation set.

Machine Learning Process



Measurements

FEATURE EXTRACTION

MACHINE LEARNING ALGORITHM

Predictions

TRAINING AND VALIDATION

PARAMETERS

How to do Machine Learning

For (supervised) classification and regression (the most common tasks):

1. Algorithm selection: Choose an algorithm
2. Feature selection: Choose features that capture the important characteristics of the system
3. Training/model building: Use part of the labeled set to build the model
4. Parameter optimization (cross-validation): Optimize the parameters using a second part of the labeled set to minimize the error rate
5. Test: Use the remainder of the dataset to validate and assess the performance of the tuned model
6. Apply