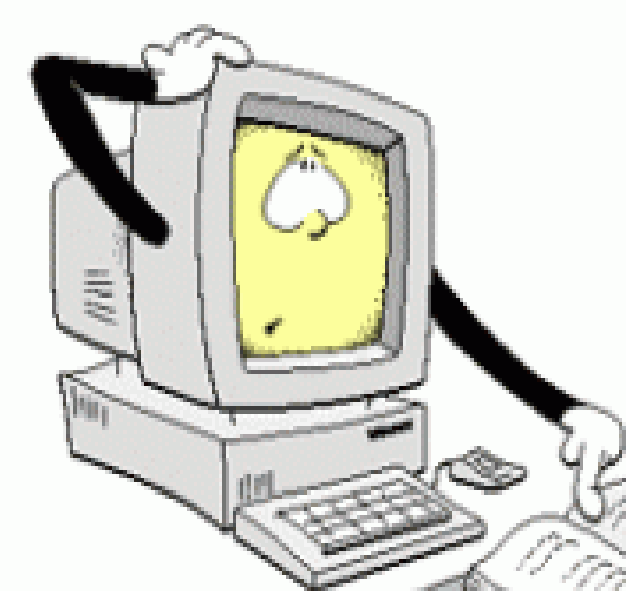
COMP6915 Machine Learning

Introduction to Machine Learning

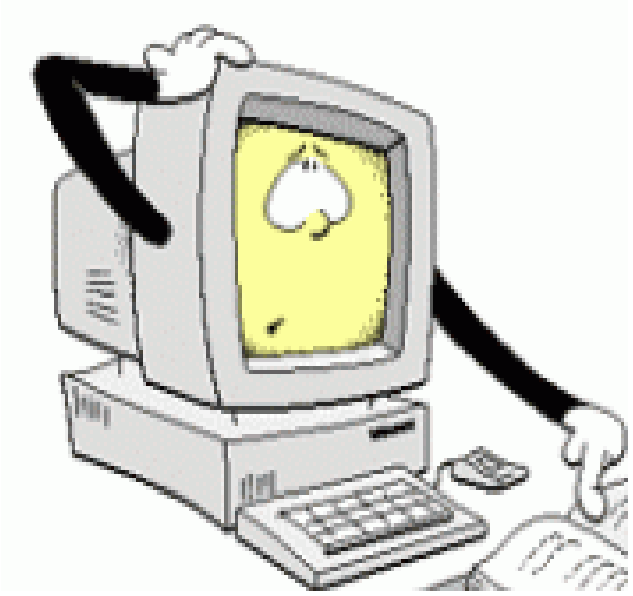
Dr. Lourdes Peña-Castillo
Departments of Computer Science and Biology
Memorial University of Newfoundland

What is Machine Learning (ML)?



- The development and study of computational systems (algorithms) that autonomously learn from data.
- At the intersection between computer science and statistics.
- Used in many fields from biology and medicine to robotics and marketing.

What is Machine Learning (ML)?



- Mitchell (1997) defines ML as “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 ML Applications

- Predict based on historical medical records which patients will respond best to which treatment (classification)
- Identify objects from digital images (pattern recognition, computer vision)
- Estimate the amount of glucose in the blood of a person after eating based on the food ingested and the gut microbes (regression)
- Discover astronomical objects from data collected by the Sloan Digital Sky Survey (knowledge extraction)
- Recommend items to customers based on a large database of online transactions (data mining)
- Control a robot driving autonomously (robotics)

Types of Machine Learning

- Supervised learning
  - Goal: Build a *model* for predicting, or estimating, an *output* based on one or more *inputs.* That is, use inputs to predict value of the output.
  - This course deals with supervised learning

Types of Machine Learning

- Unsupervised learning
  - Goal: Learn relationships and structure from inputs (data). That is, find regularities (patterns) in the data and describe how the data are organized or clustered (grouped). Inputs are not associated with an output or label.
  - Examples:
    - Find similarities in gene expression among samples of cancer tumours from different patients (*bioinformatics*)
    - Group customers based on demographic information and past transactions with the company to identify more frequent type of customer (*customer segmentation)*

Types of Machine Learning

- Reinforcement learning
  - Goal: Learn a sequence of actions (policy) to reach the goal based on a reward function.
  - The reward function indicates to the learner whether it is doing well or poorly
  - In reinforcement learning, the algorithm interacts with an environment; thus, there is a feedback loop between the learning system and its experiences.
  - Examples:
    - Legged robots learning to walk
    - Autonomous helicopter flight
    - Game playing

Supervised, unsupervised or reinforcement learning?

- Predict based on historical medical records which patients will respond best to which treatment
- Identify objects from digital images
- Estimate the amount of glucose in the blood of a person after eating based on the food ingested and the gut microbes
- Discover astronomical objects from data collected by the Sloan Digital Sky Survey
- Recommend items to customers based on a large database of online transactions
- Control a robot driving autonomously
- Find patients subgroups based on genetic variations in cancer cells
- Recognize handwritten digits

Terminology

- Inputs are also called *attributes*, *predictors, independent variables or features.*
- Outputs are also called *responses*, *targets* or *dependent variables.*
- Variables types:
  - Quantitative (continuous)
  - Qualitative (also referred to as *categorical* or *discrete variables*, or *factors*)
    - commonly represented numerically by codes

Terminology

- Based on the output type (quantitative or categorical), prediction tasks are called:
  - *regression* when quantitative (continuous) outputs are predicted
  - *classification* when categorical outputs are predicted
- Both prediction tasks can be seen as a task in *function approximation*

Notation

- An input variable is denoted by *X.* If *X* is a vector, its components can be accessed by subscripts X_j
- Observed values are written in lowercase: x_i is the *i*th observed value of *X*
- *O*utputs are denoted by *Y*
- Suppose we observe a response *Y* and *p* different predictors, $X = (X_1, X_2, \ldots, X_p)$. We assume that there is a relationship between *Y* and *X,* which can be written as

  $Y = f(X)$ where *f* is some fixed but unknown function of *X*

Learning Task

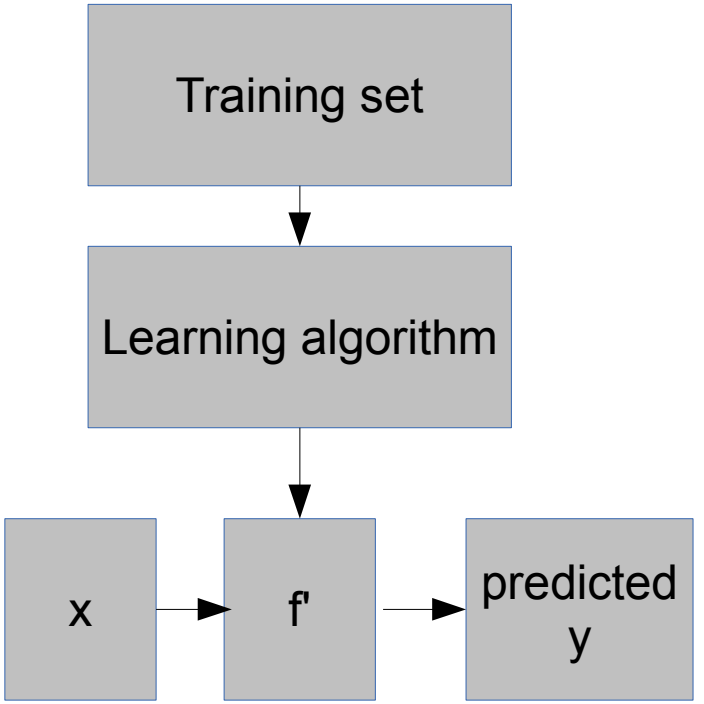
- The supervised learning task is:
  - Given the value of an input vector X, make a good prediction of the output Y, denoted by $\hat{Y}$. That is, estimate f.
- To construct our estimate of f, denoted by $\hat{f}$, we need data. We thus suppose we have available a set T of (x_i,y_i), $i = 1,...,N$ known as the training data.
- Each (x_i,y_i) is called an *example*, *instance*, *observation*, or *data point*.

Learning Process

1. Given a training set $T = (x_i, y_i)$, $i = 1,\ldots,N$. The observed input values x_i are fed into a learning algorithm which produces outputs $f'(x_i)$ in response to the inputs.
2. The learning algorithm has the property that it can modify its f' in response to differences $y_i - f'(x_i)$ between the original and generated outputs.

   This process is known as learning by example.
3. Upon completion of the learning process, the hope is that f' is a good approximation of f and thus a good predictor of $f(x) = y$

   f' is called a *hypothesis* or *model*



In sum, machine learning refers to a set of approaches for estimating *f*

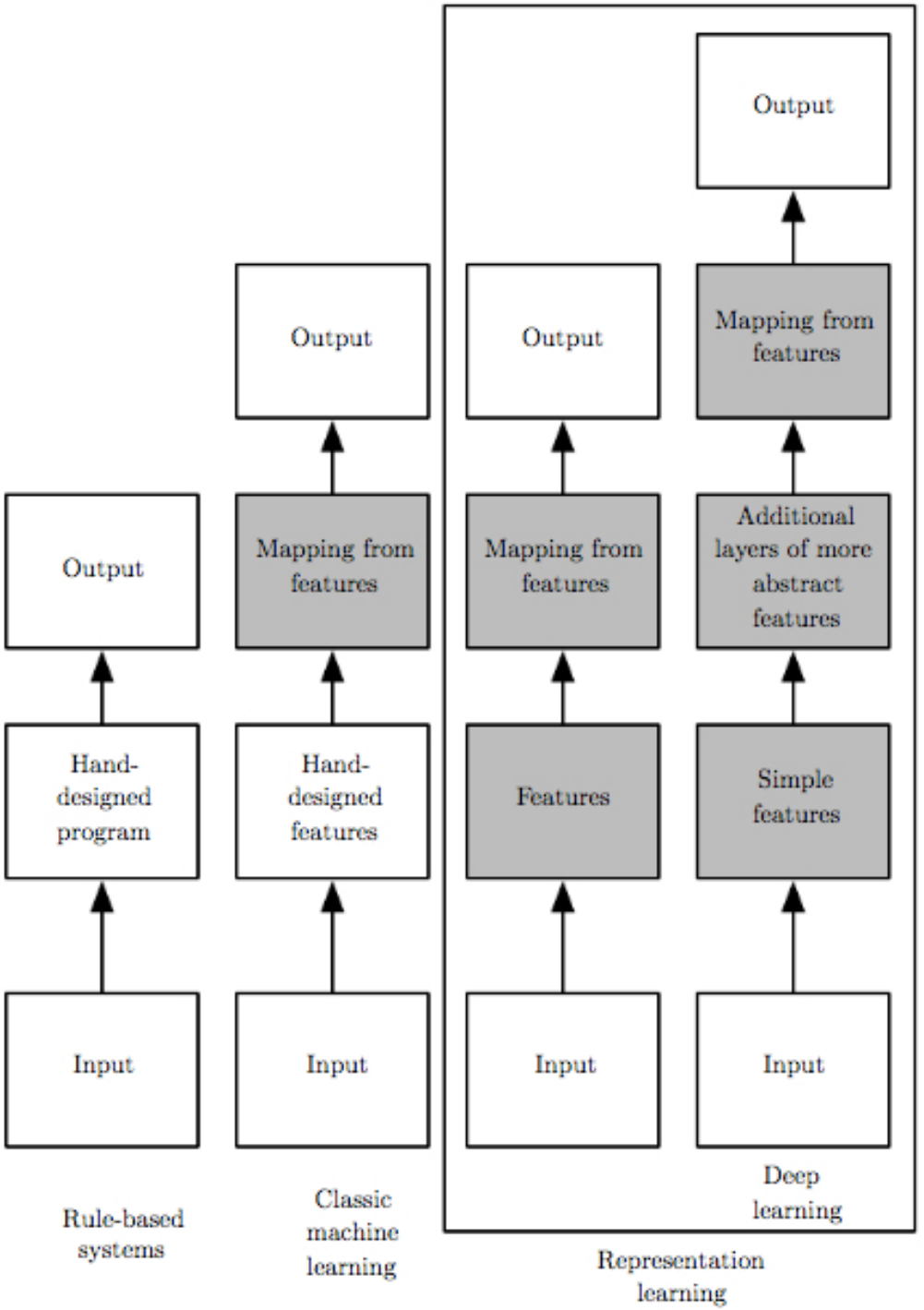


Figure 1.5: Flowcharts showing how the different parts of an AI system relate to each other within different AI disciplines. Shaded boxes indicate components that are able to learn from data.

Fig from *Deep Learning* by Goodfellow et al.

Methods to estimate *f*

- Parametric Methods simplify the problem of estimating *f* to one of estimating a set of parameters.

1. Make an assumption about the shape of *f*. For example, assume *f* is linear in *X* (i.e., a linear model):

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

2. Use training data to *fit* or *train* the model. In the case of the linear model, we need to estimate the *beta* parameters.

Methods to estimate *f*

- A potential disadvantage of a parametric approach is that the model chosen does not match the true unknown form of *f*
- The set of functions a learning algorithm can select as *f'* is called its *hypothesis space*
  - The linear model has the set of all linear functions of its input as its hypothesis space
  - Increasing the hypothesis space increases the model's capacity

Methods to estimate *f*

- Non-Parametric Methods
  - do not make explicit assumptions about the shape of *f* instead they seek an estimate of *f* that fits the data points.
  - have the potential to fit a wider range of possible shapes for *f*
  - require a larger number of observations than the parametric approaches to obtain an accurate estimate for *f*

Underfitting and Overfitting

- A learned model must perform well on ***new, previously unseen*** inputs (i.e., not in the training data). This is called *generalization*.
- Empirical error or Training error is the proportion of instances in T where $f'(x) \neq f(x)$
- *Generalization error or* Test error is the proportion of unseen instances (i.e., examples not in T) where $f'(x) \neq f(x)$

Underfitting and Overfitting

- *Underfitting* occurs when the model is not able to obtain a low training error
- *Overfitting* occurs when the gap between the training error and test error is too large (i.e, the model is not able to generalize)
- Models with insufficient capacity to solve a specific task will underfit
- Models with higher capacity than needed to solve the task may overfit

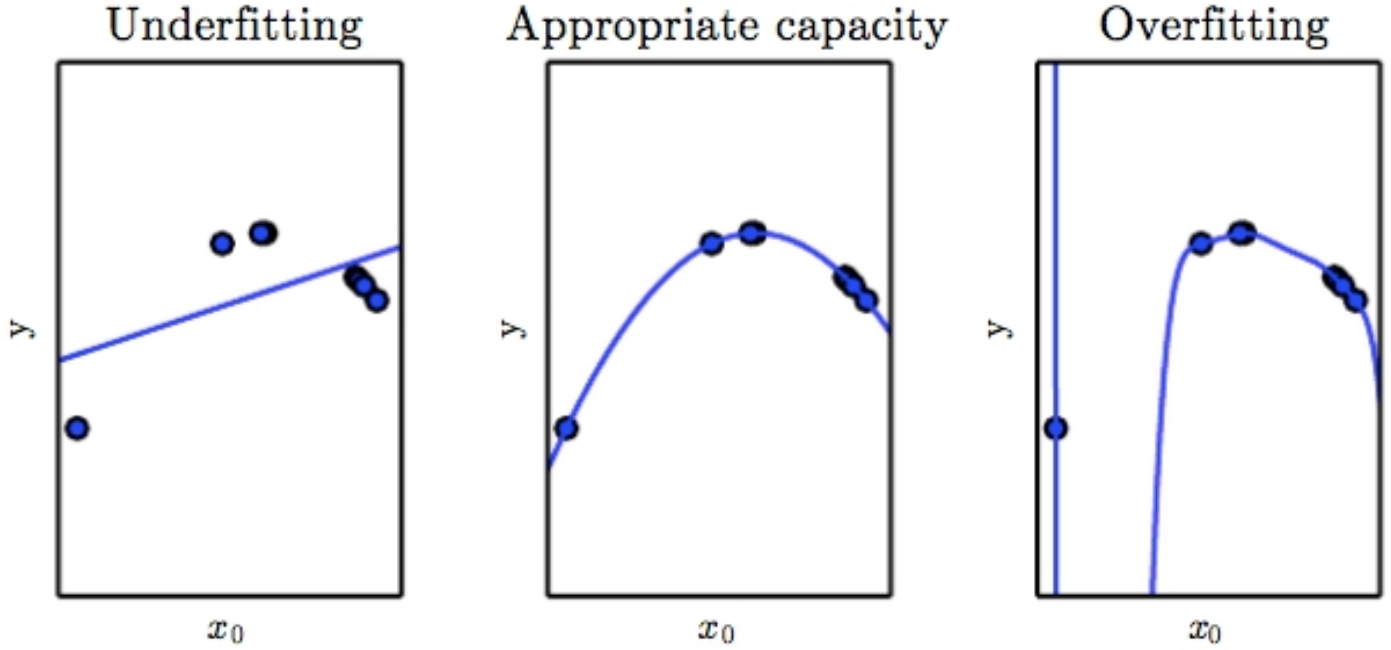


Figure 5.2: We fit three models to this example training set. The training data was generated synthetically, by randomly sampling x values and choosing y deterministically by evaluating a quadratic function. *(Left)*A linear function fit to the data suffers from underfitting—it cannot capture the curvature that is present in the data. *(Center)*A quadratic function fit to the data generalizes well to unseen points. It does not suffer from a significant amount of overfitting or underfitting. *(Right)*A polynomial of degree 9 fit to the data suffers from overfitting. Here we used the Moore-Penrose pseudoinverse to solve the underdetermined normal equations. The solution passes through all the training points exactly, but we have not been lucky enough for it to extract the correct structure. It now has a deep valley between two training points that does not appear in the true underlying function. It also increases sharply on the left side of the data, while the true function decreases in this area.

Fig from *Deep Learning* by Goodfellow et al.

Undefitting and Overfitting

- To improve the generalization of machine learning models (and avoid overfitting), we follow Occam's razor (c. 1287-1347):
  - Among competing hypotheses (models) that explain known observations equally well, we should choose the “simplest” one
- Simpler functions are more likely to generalize, but we need to choose a sufficiently complex hypothesis to achieve a low training error

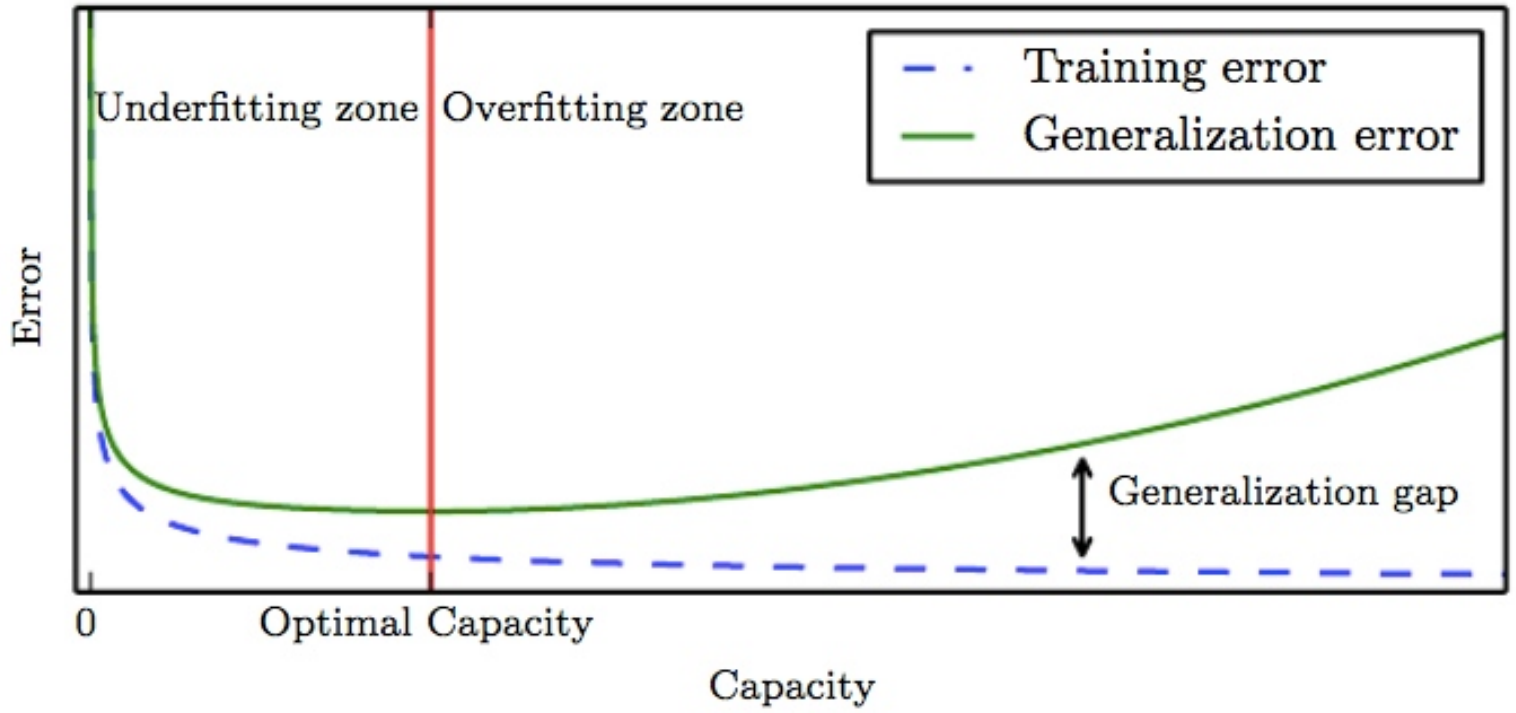


Figure 5.3: Typical relationship between capacity and error. Training and test error behave differently. At the left end of the graph, training error and generalization error are both high. This is the **underfitting regime**. As we increase capacity, training error decreases, but the gap between training and generalization error increases. Eventually, the size of this gap outweighs the decrease in training error, and we enter the **overfitting regime**, where capacity is too large, above the **optimal capacity**.

Fig from *Deep Learning* by Goodfellow et al.

The No Free Lunch Theorem

- We will see a wide range of learning methods in this course. Why? Why we don't just discuss the single best method?
- The **no free lunch theorem** for ML (Wolpert, 1996) states that, averaged over all possible data-generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points
  - no one method dominates all others over all possible data sets
- ML challenge: select the method that for a given set of data produces the best results. That is, ML algorithms are designed to perform well on a specific task.

Regularization

- Regularization is any modification made to a learning algorithm that is intended to reduce its generalization error but not its training error

Bias

- Inductive bias is the set of assumptions made to find a hypothesis (i.e., to make learning possible)
- For example,
  - assuming f is a linear function is an inductive bias
  - choosing the f' that minimizes empirical error is another inductive bias

Variance

- Variance is the amount by which *f'* changes if we estimated it using a different training data set
- Ideally the estimate for *f* should not vary too much between training sets.
- If a ML method has high variance then small changes in the training data can result in large changes in *f'*

Bias-Variance Trade-off

- As a general rule, more flexible methods (i.e., with larger capacity) will have higher variance and lower bias
- As the variance increases, the bias decreases

Bias-Variance Trade-off

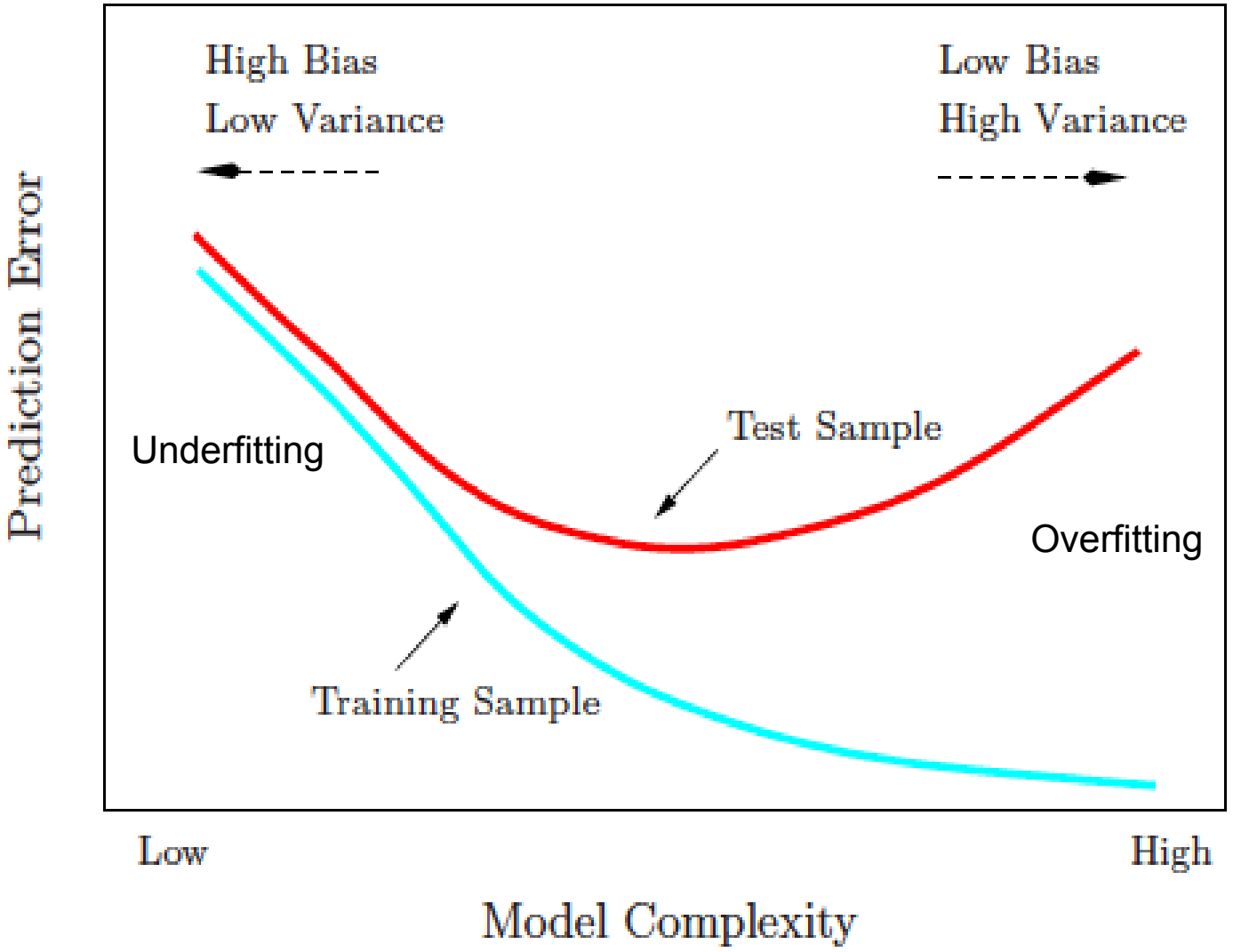
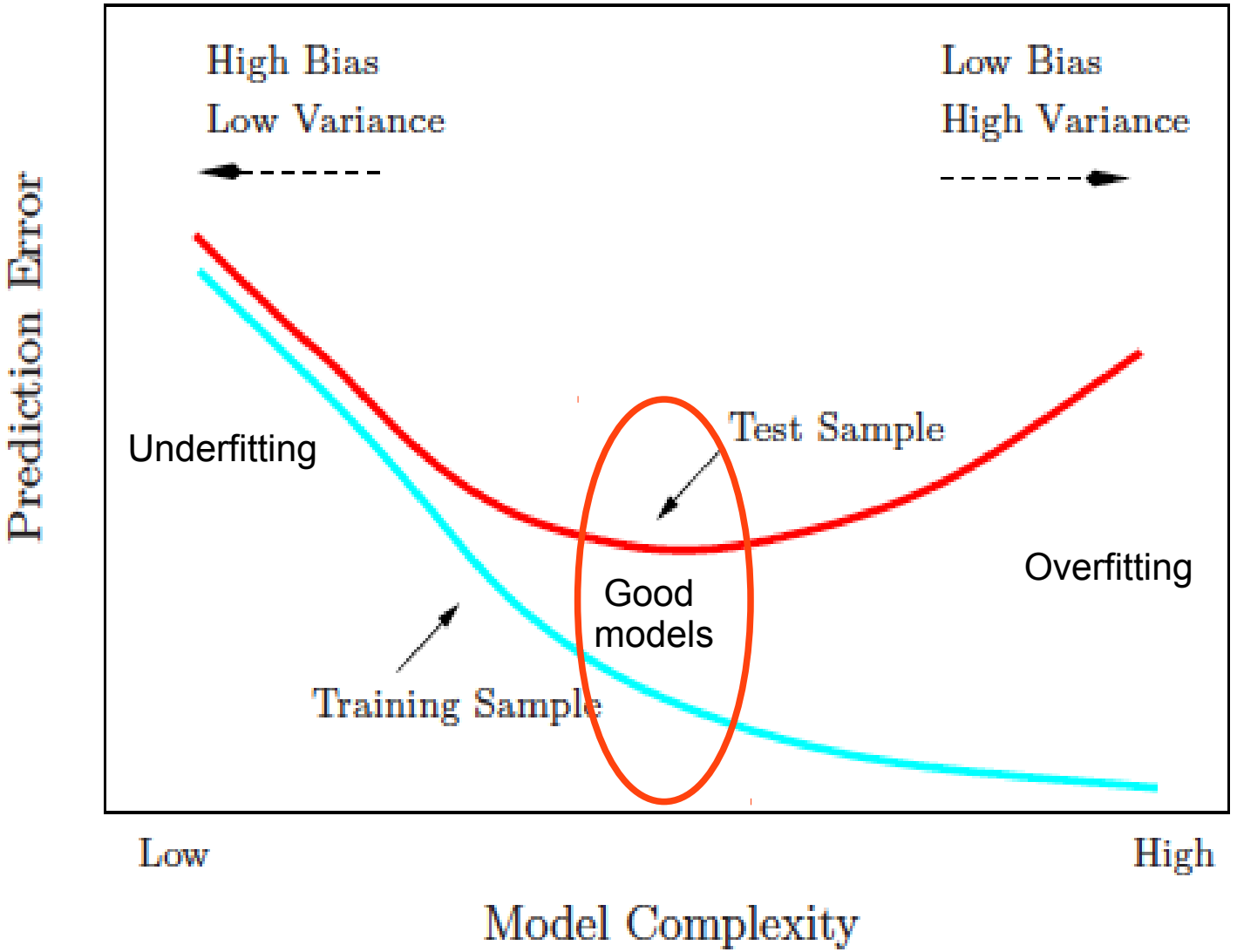


Fig. from *The Elements of Statistical Learning* by Hastie, Tibshirani and Friedman, 2009.

Bias-Variance Trade-off



Note: Training error is not a good estimate of test error!

Adapted from *The Elements of Statistical Learning* by Hastie, Tibshirani and Friedman, 2009.

Noise

- Noise is an anomaly of the data that may cause that zero test error is infeasible
- Noise causes an irreducible error
- Noise may be due to:
  - Errors recording input attributes
  - Error labelling training instances
  - Missing additional attributes

Reducible and Irreducible Error

$Y = f(x) + \epsilon$ where ϵ is a random error term, which is independent of X and has mean zero.

Assume that both f and X are fixed. Then

$E(Y - \hat{Y})^2 = E[f(X) + \epsilon - f'(X)]^2 = [f(X) - f'(X)]^2 + Var(\epsilon)$

where $E(Y - \hat{Y})^2$ represents the average, or expected value, of the squared difference between the predicted and actual value of Y.

Reducible and Irreducible Error

$[f(X) - f'(X)]^2$ is a **reducible error** because we can potentially improve the accuracy of f'.

However, $Var(\epsilon)$ is an **irreducible error** because Y is also a function of ϵ, which cannot be predicted using X.

Machine learning focuses on techniques for estimating f with the aim of minimizing the reducible error.

By now, you should be able to

- define machine learning
- explain what supervised learning, unsupervised learning and reinforcement learning are
- classify learning tasks as supervised, unsupervised or reinforcement learning
- formally define supervised machine learning
- know machine learning terminology
- understand the machine learning notation used in this course
- explain the learning process of learning by example
- explain the differences between parametric and non-parametric methods
- define training error, test error, underfitting, overfitting, bias, variance and regularization
- say Occam's razor and the No Free Lunch theorem and explain their relationship to machine learning
- understand the bias-variance trade-off
- explain reducible and irreducible error