Machine Learning: Algorithms and Applications

Philip O. Ogunbona

Advanced Multimedia Research Lab
University of Wollongong

General Introductions
Autumn 2024

"The formulation of the problem is often more essential than its solution, which may be merely a matter of mathematical or experimental skill." -Albert Einstein

Outline

Machine Learning - General ideas

- Machine learning is the automated process of extracting patterns from data
- Machine learning is programming computers to optimize a performance criterion using example data or past experience
- In a general sense machine learning algorithms set out to learn some function that maps "units" from one space to "units" in another space
- A learning algorithm is one that can learn from data
- Learning algorithm may involve: optimization, a cost function, a model and a data set, to build the algorithm
- There is a model defined up to some parameters and learning is the process of optimizing the parameters using training data
- Model may be predictive (make prediction about new or future data) or descriptive (gain knowledge or insight about data) or both

Machine Learning - General ideas

What is machine learning?

Machine learning algorithms work by searching through a set of possible models to select the model that best captures the relationship between the descriptive features and the target feature in the dataset (this is only part of the story; there is a rich theory that explains it in full)

Mitchell 1997 defined learning broadly:

What is learning?

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

- Tasks of interest are “too difficult to solve with fixed programs written and designed by humans”
- The process of learning is not the task; learning is a means of acquiring ability to perform the task

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri, Rostamizadeh, & Talwalkar, 2012):

- Supervised learning: The learner receives a set of lebelled examples as training data and makes predictions for all unseen points. For example we encounter this in classification, regression, and ranking problems.
- Unsupervised learning: The learner exclusively receives unlabelled data and makes predictions for all unseen points. Clustering and dimensionality reduction are examples.
- Semi-supervised learning: The learner receives a training sample consisting of both lebelled and unlabelled data and makes predictions for all unseen points. This is usually employed in cases where unlabelled data is readily available but labelled data is expensive to obtain.
- Transductive inference: Similar to semi-supervised learning, the learner receives a labelled training sample along with a set of unlabelled test points. The objective of the transductive inference is to predict labels only for these particular test points.

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri et al., 2012):

- Supervised learning: The learner receives a set of lebelled examples as training data and makes predictions for all unseen points. For example we encounter this in classification, regression, and ranking problems.

- Unsupervised learning: The learner exclusively receives unlabelled data and makes predictions for all unseen points. Clustering and dimensionality reduction are examples.

- Semi-supervised learning: The learner receives a training sample consisting of both lebelled and unlabelled data and makes predictions for all unseen points. This is usually employed in cases where unlabelled data is readily available but labelled data is expensive to obtain.

- Transductive inference: Similar to semi-supervised learning, the learner receives a labelled training sample along with a set of unlabelled test points. The objective of the transductive inference is to predict labels only for these particular test points.

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri et al., 2012):

- Supervised learning: The learner receives a set of lebelled examples as training data and makes predictions for all unseen points. For example we encounter this in classification, regression, and ranking problems.
- Unsupervised learning: The learner exclusively receives unlabelled data and makes predictions for all unseen points. Clustering and dimensionality reduction are examples.
- Semi-supervised learning: The learner receives a training sample consisting of both lebelled and unlabelled data and makes predictions for all unseen points. This is usually employed in cases where unlabelled data is readily available but labelled data is expensive to obtain.
- Transductive inference: Similar to semi-supervised learning, the learner receives a labelled training sample along with a set of unlabelled test points. The objective of the transductive inference is to predict labels only for these particular test points.

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri et al., 2012):

- Supervised learning: The learner receives a set of lebelled examples as training data and makes predictions for all unseen points. For example we encounter this in classification, regression, and ranking problems.
- Unsupervised learning: The learner exclusively receives unlabelled data and makes predictions for all unseen points. Clustering and dimensionality reduction are examples.
- Semi-supervised learning: The learner receives a training sample consisting of both lebelled and unlabelled data and makes predictions for all unseen points. This is usually employed in cases where unlabelled data is readily available but labelled data is expensive to obtain.
- Transductive inference: Similar to semi-supervised learning, the learner receives a labelled training sample along with a set of unlabelled test points. The objective of the transductive inference is to predict labels only for these particular test points.

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri et al., 2012):

- On-line learning: This scenario involves multiple rounds and, training and testing phases are intermixed. At each round, the learner receives an unlabelled training data point, makes a prediction and incurs a loss. The objective is to minimize the cumulative loss over all rounds.

- Reinforcement learning: Training and testing phases are intermixed. Learner collects information by actively interacting with the environment and sometime also affecting the environment, to receive immediate reward for each action. The goal is to maximize the reward over time. This learning scenario is related to dynamic programming.

- Active learning: The learner adaptively or interactively collects training examples by querying an oracle to request labels from new points. The goal is to achieve performance comparable to the standard supervised learning scenario.

- Few shot learning: The key idea is to emulate the human ability to learn from a handful of examples. "Few-shot learning methods range widely, from adapting pre-trained models for use in similar tasks to using generative models to create new samples to meta learning methods that train models to generalize well to new classification problems and different classes of data, rather than perform any one specific task"[1]. For a survey of this exciting learning paradigm see Parnami and Lee (2022).

[1] https://www.ibm.com/topics/few-shot-learning

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri et al., 2012):

- On-line learning: This scenario involves multiple rounds and, training and testing phases are intermixed. At each round, the learner receives an unlabelled training data point, makes a prediction and incurs a loss. The objective is to minimize the cumulative loss over all rounds.

- Reinforcement learning: Training and testing phases are intermixed. Learner collects information by actively interacting with the environment and sometime also affecting the environment, to receive immediate reward for each action. The goal is to maximize the reward over time. This learning scenario is related to dynamic programming.

- Active learning: The learner adaptively or interactively collects training examples by querying an oracle to request labels from new points. The goal is to achieve performance comparable to the standard supervised learning scenario.

- Few shot learning: The key idea is to emulate the human ability to learn from a handful of examples. "Few-shot learning methods range widely, from adapting pre-trained models for use in similar tasks to using generative models to create new samples to meta learning methods that train models to generalize well to new classification problems and different classes of data, rather than perform any one specific task"[1]. For a survey of this exciting learning paradigm see Parnami and Lee (2022).

[1] https://www.ibm.com/topics/few-shot-learning

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri et al., 2012):

- On-line learning: This scenario involves multiple rounds and, training and testing phases are intermixed. At each round, the learner receives an unlabelled training data point, makes a prediction and incurs a loss. The objective is to minimize the cumulative loss over all rounds.

- Reinforcement learning: Training and testing phases are intermixed. Learner collects information by actively interacting with the environment and sometime also affecting the environment, to receive immediate reward for each action. The goal is to maximize the reward over time. This learning scenario is related to dynamic programming.

- Active learning: The learner adaptively or interactively collects training examples by querying an oracle to request labels from new points. The goal is to achieve performance comparable to the standard supervised learning scenario.

- Few shot learning: The key idea is to emulate the human ability to learn from a handful of examples. "Few-shot learning methods range widely, from adapting pre-trained models for use in similar tasks to using generative models to create new samples to meta learning methods that train models to generalize well to new classification problems and different classes of data, rather than perform any one specific task"[1]. For a survey of this exciting learning paradigm see Parnami and Lee (2022).

[1] https://www.ibm.com/topics/few-shot-learning

Common Machine Learning Scenarios

Some machine learning scenarios (Mohri et al., 2012):

- On-line learning: This scenario involves multiple rounds and, training and testing phases are intermixed. At each round, the learner receives an unlabelled training data point, makes a prediction and incurs a loss. The objective is to minimize the cumulative loss over all rounds.

- Reinforcement learning: Training and testing phases are intermixed. Learner collects information by actively interacting with the environment and sometime also affecting the environment, to receive immediate reward for each action. The goal is to maximize the reward over time. This learning scenario is related to dynamic programming.

- Active learning: The learner adaptively or interactively collects training examples by querying an oracle to request labels from new points. The goal is to achieve performance comparable to the standard supervised learning scenario.

- Few shot learning: The key idea is to emulate the human ability to learn from a handful of examples. "Few-shot learning methods range widely, from adapting pre-trained models for use in similar tasks to using generative models to create new samples to meta learning methods that train models to generalize well to new classification problems and different classes of data, rather than perform any one specific task"[1]. For a survey of this exciting learning paradigm see Parnami and Lee (2022).

[1] https://www.ibm.com/topics/few-shot-learning

Classes of Learning Problems/Tasks

Examples of learning problems/tasks.

1. Classification - Assign a category to each item.
2. Regression - Predict a value for each item.
3. Ranking - Order items according to some criterion. E.g. Web search.
4. Clustering - Partition items into homogeneous regions.
5. Dimensionality reduction or manifold learning - Transform an initial representation of items into a lower dimension of these items while retaining some properties.
6. Natural language understanding - Given a piece of textual data (or speech), produce an understanding of meaning of the text or speech
7. Question answering - Given a question posed by human provide answer in a natural language
8. Dialogue - Design a system that converses with human using speech, text, and other modes of communication

Tasks:

- Tasks are described in terms of how the machine learning system should process the example
- Examples are collection of *features* quantitatively measured from some object or event that machine learning system will process - $\boldsymbol{x} \in \mathbb{R}^n$
- Classification
  - Required to specify, to which of k categories some input belongs
  - A function of the following form is to be learned from data:

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

  - $y = f(\boldsymbol{x})$ assigns feature $\boldsymbol{x}$ to category identified by y
  - Function could also output a probability distribution over classes (categories)
- Regression
  - Required to predict a numerical value given some input
  - A function of the following form is to be learned from data:

$$f : \mathbb{R}^n \rightarrow \mathbb{R} \tag{2}$$

  - Contrast with classification where output is categorical data type

Tasks:

- Transcription
  - Required to observe a relatively unstructured representation of some kind of data and transcribe it into discrete textual form
  - A function of the following form is to be learned from data:

$$f : \mathbb{R}^n \rightarrow \mathbb{A}^{k(m)} \tag{3}$$

where $\mathbb{A}$ is the set of some language alphabets (English, Yoruba, etc) and $k(m)$ is some variable number that indicates variable lengths of alphabets and depends on the application. Speech recognition is an example.

- Machine translation
  - Required to convert sequence of symbols (or alphabets) in some language to sequence of symbols (alphabets) in another language
  - A function of the following form is to be learned from data:

$$f : \mathbb{A}_x^{k(n)} \rightarrow \mathbb{A}_y^{k(m)} \tag{4}$$

where $\mathbb{A}_x$ is the set of some source language alphabets (English, Punjabi, Urdu, Mandarin, Yoruba, etc.) and $k(m)$ is some variable number that indicates variable lengths of alphabets, $\mathbb{A}_y$ is the set of some target language alphabets (German, French, Russian, etc.).

Tasks:

- Structured output
  - This category of tasks subsumes transcription and translation
  - Required to convert input into an output modelled as data structure containing multiple values with important relationship between elements
  - Example includes providing a textual (sentence) description of given picture or textual description of the activity being performed in a given video
- Anomaly detection
  - Required to sift through a set of events or objects and flag some of them as being unusual or atypical
  - Anomalous object belongs to a probability distribution very different from the rest of events or object.
  - How to estimate distribution and how to measure distance between distribution?

Tasks:

- synthesis and sampling
  - Required to generate new examples that are similar to those in the training data
  - Possibly, required output has specified structure with bounds
  - Example: synthesize speech from written text in various accents

Challenge

- Given a description of a suspected criminal, can we generate possible pictures of the person?
- What type of training samples will you consider?

Tasks:

Figure 1: Generative adversarial network (GAN) was given examples of images of bedroom and these outputs were automatically generated

Tasks:

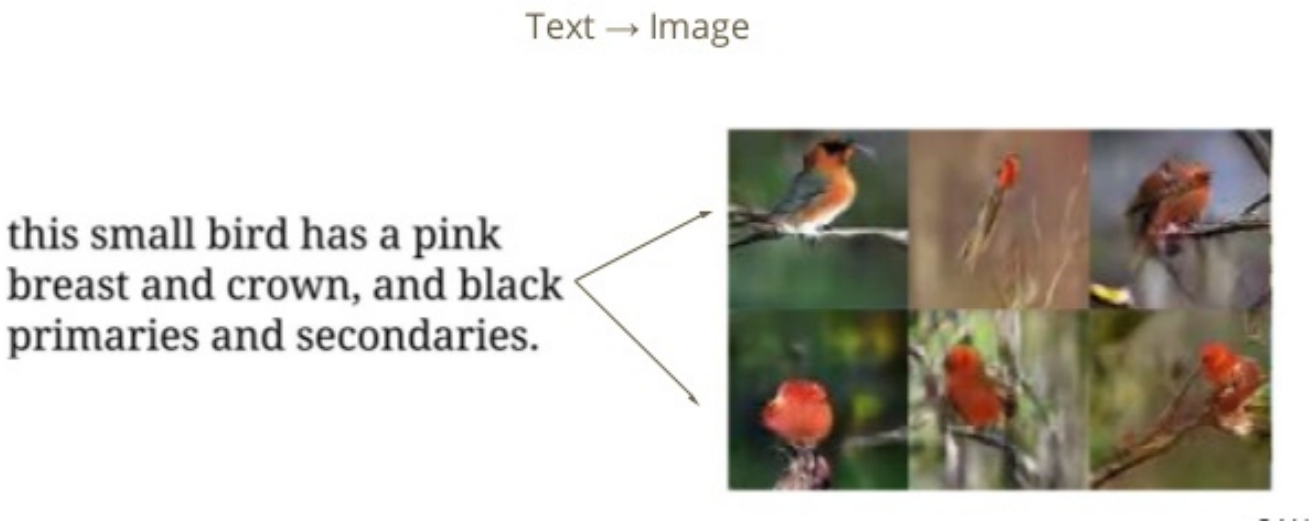

Figure 2: Generative adversarial network (GAN) was given text and examples of images of birds that match the description were automatically generated

Tasks:

- Imputation of missing values
  - Required to provide a prediction of values of missing entries, x_i in a given example, $\boldsymbol{x} \in \mathbb{R}^n$
- Denoising
  - Required to predict the clean example $\boldsymbol{x}$, from its corrupted version $\tilde{\boldsymbol{x}}$
  - This is the same as predicting the conditional probability distribution $p(\boldsymbol{x}|\tilde{\boldsymbol{x}})$

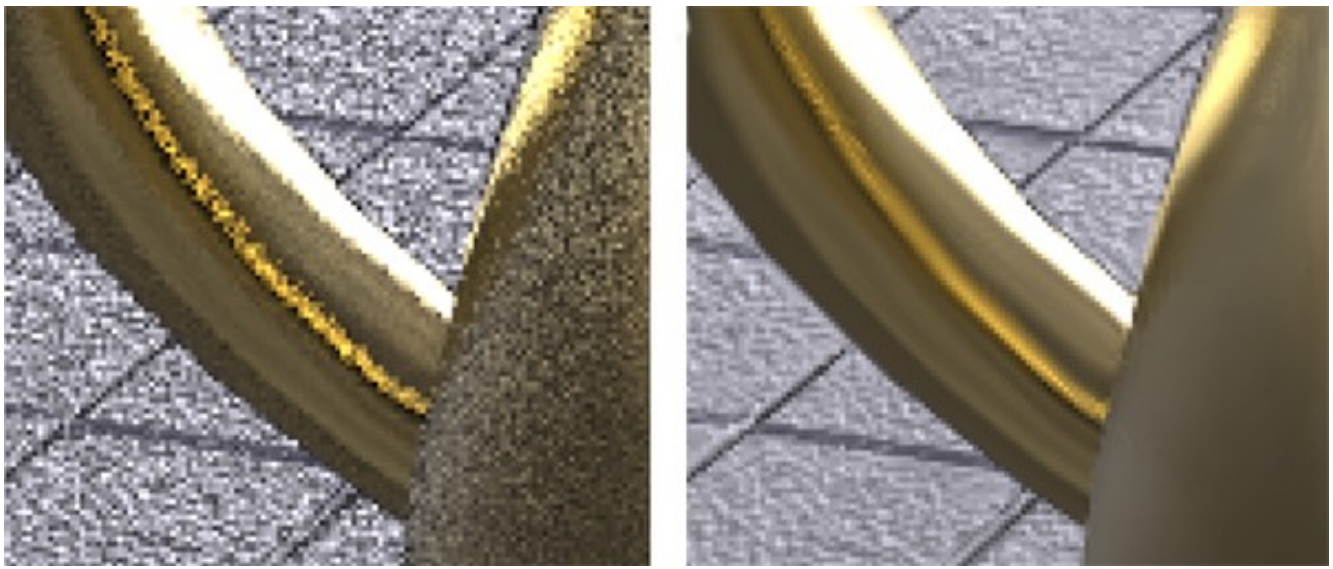

Figure 3: Image on the right is a predicted clean version of the left image

Tasks:

- Density estimation or probability function estimation
  - Required to learn the function

$$p_{\text{model}} : \mathbb{R}^n \rightarrow \mathbb{R} \tag{5}$$

    where p_{model} is the probability density function or probability mass function on the space from which the examples were drawn.
  - Many of the tasks in machine learning requires the estimation of the probability density.

Performance

- Quantitative measures of performance are required to evaluate the abilities of machine learning algorithms
- Performance is task-specific
- Examples:
  - Accuracy,
  - Error rate,
  - Precision,
  - Recall, etc.
- Performance measure must be chosen to match the desired behaviour of the system

Experience

- The experience a machine learning algorithm is exposed to could be supervised or unsupervised
- Unsupervised learning algorithms experience dataset containing many features and are required to learn useful properties from the dataset
  - In essence, given several examples of a random vector $\boldsymbol{x}$, implicitly or explicitly learn the probability distribution, $p(\boldsymbol{x})$
- Supervised learning algorithms experience dataset containing many features as well as label (or target) associated with each example
  - In essence, given several examples of a random vector $\boldsymbol{x}$, and associated labels ($\boldsymbol{y}$), learn to estimate the conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$

Discussion point

Discussion

1. What amount of data do we need to build a good machine learning system?
2. How will a machine learning system perform if there is a mismatch between the distribution of the training data and test data?
3. How can we quantify such mismatch?

Data representation

- Design matrix, $\boldsymbol{X} \in \mathbb{R}^{N \times P}$

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_i^t \\ \vdots \\ \boldsymbol{x}_N^t \end{bmatrix} \tag{6}$$

represents N examples (or observations or samples) of a collection of P features, where the i-th feature vector is represented as

$$\boldsymbol{x}_i = \begin{bmatrix} x_i \\ \vdots \\ x_P \end{bmatrix} \tag{7}$$

and x_i is a measured feature value.

- A set can also be used in situations where number of features in examples are not equal:

$$\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}\} \tag{8}$$

represents a collection of m elements, not all of equal size.

Example: I

Linear Regression

- Build a system that can take a vector $\boldsymbol{x} \in \mathbb{R}^P$ and predict the value of a scalar $y \in \mathbb{R}$ that is postulated to depend on $\boldsymbol{x}$
- Linear regression model $\rightarrow$ output is a linear function of input

$$\hat{y} = \omega^t \boldsymbol{x} \tag{9}$$

where $\omega \in \mathbb{R}^P$ is a vector of parameters. They determine how each feature affects the prediction.

- The task T, is:

> Predict y from $\boldsymbol{x}$ by computing $\hat{y} = \omega^t \boldsymbol{x}$

- How do we measure performance, P?

Example: II

- In a regression problem we will have a dataset represented by design matrix X.
- Let us partition the data set into two: *test* and *training* sets. So we have $\boldsymbol{X}^{(\text{test})}$ and $\boldsymbol{X}^{(\text{training})}$
- We learn the model with $\boldsymbol{X}^{(\text{training})}$ and test performance with $\boldsymbol{X}^{(\text{test})}$.
- Assume we use mean squared error of model as performance measure

$$\text{MSE}_{test} = \frac{1}{N_{(\text{test})}} \sum_i \left(\hat{y}^{(\text{test})} - y^{(\text{test})} \right)_i^2 \tag{10}$$

- Training involves finding the weight vector ω that will minimize the $\text{MSE}_{training}$:

$$\min_{\omega} \frac{1}{P} ||\hat{\boldsymbol{y}}^{(\text{training})} - \boldsymbol{y}^{(\text{training})}||_2^2$$
$$\text{s.t} \quad y = \omega^t \boldsymbol{x} \tag{11}$$

- Differentiating and equating to zero gives:

$$\omega = \left(\boldsymbol{X}^{(\text{training})t} \boldsymbol{X}^{(\text{training})} \right)^{-1} \boldsymbol{X}^{(\text{training})t} \boldsymbol{y}^{(\text{training})} \tag{12}$$

Example: III

- Equation (12) constitutes a simple learning algorithm.
- More generally the linear regression model will be written as,

$$\hat{y} = \omega^t \boldsymbol{x} + \boldsymbol{b}. \tag{13}$$

Capacity, Overfitting and Underfitting:

- Generalization is key in machine learning
  - Trained algorithm must perform well on new and previously unseen inputs
- We optimize to reduce the training error
  - But we want the generalization error (test error) to be low as well
  - This is the expected error on new inputs
  - IMPORTANT: Expectation is taken across different possible inputs drawn from the distribution of inputs we expect the system to encounter in practice

$$\begin{cases} \text{Train to minimize:} & \frac{1}{N^{(\text{training})}} ||\boldsymbol{X}^{(\text{training})}\boldsymbol{\omega} - \boldsymbol{y}^{(\text{training})}||_2^2 \\ \text{Judge generalization on test error:} & \frac{1}{N^{(\text{test})}} ||\boldsymbol{X}^{(\text{test})}\boldsymbol{\omega} - \boldsymbol{y}^{(\text{test})}||_2^2 \end{cases} \tag{14}$$

- This is possible because of assumption of (i.i.d) - independent and identically distributed random variates generated by the data generating process

Capacity, Overfitting and Underfitting:

- Factors that determine how well a machine learning algorithm will perform are its ability to:
  - make training error small
  - make gap between training and test error small
- Underfitting: model is unable to obtain sufficently low error value on training set
- Overfitting: the gap between training and test error is too large
- Capacity: ability of model to fit a wide variety of functions
- Control Underfitting and Overfitting by altering capacity and/or by increasing the number of training samples
  - Alter the capacity by choosing the hypothesis space (the set of functions the learning algorithm is allowed to select as possible solution) - equivalent to increasing/decreasing the number of features
  - Overfitting can also be avoided by increasing the number of example data
- The triple trade-off of learning algorithms trained from example data (Alpaydin, 2010, pp. 39):
  - complexity of the hypothesis fitted to data; in other words capacity of the hypothesis class,
  - amount of training data available,
  - generalization error on new examples.

Capacity, Overfitting and Underfitting:

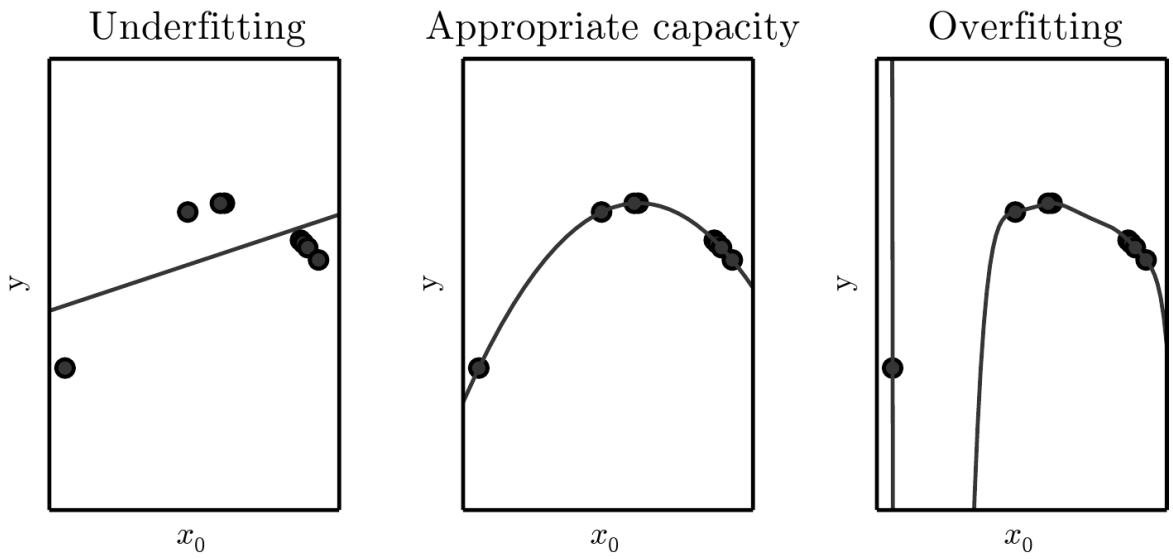

Figure 4: Three models fitted to example of synthetic data generated by randomly sampling x, and computing corresponding value of y from a quadratic equation. (*Left*) fits linear function; unable to capture the curvature and hence underfits, (*Centre*) fits a quadratic function; generalises well to unseen data; no significant underfitting or overfitting, (*Right*) fits polynomial of degree 9; suffers overfitting; note the strange structure of the curve as it tries to pass through all training data (Goodfellow et al., 2016).

PAC Learning Framework[2] I

PAC - Probably Approximately Correct

We start by stating some definitions:

- Let $\mathcal{X}$ denote the set of all possible examples or instances. This is the input space.
- The set of all possible labels or target values is denoted $\mathcal{Y}$. Without loss of generality, let $\mathcal{Y} = \{0, 1\}$; implying a binary classification.
- A concept $c : \mathcal{X} \to \mathcal{Y}$ is a mapping from $\mathcal{X}$ to $\mathcal{Y}$. Thus we identify c with the subset of $\mathcal{X}$ over which it takes value 1.
- Assume examples are independent and identically distributed (i.i.d) according to some fixed but unknown distribution $\mathcal{D}$.

[2] This concept can be visited later when students gain deeper intuitive understanding

PAC Learning Framework

We state the learning problem:

Problem (Learning)

The learner considers a fixed set of possible concepts $\mathcal{H}$, called a hypothesis set, which may not conincide with $\mathcal{C}$. Learner receives a sample $\mathcal{S} = \{x_1, \ldots, x_m\}$ drawn i.i.d according to $\mathcal{D}$ as well as labels $(c(x_1), \ldots, c(x_m))$, which are based on specific target concept $c \in \mathcal{C}$ to learn. The task is to use the labelled sample $\mathcal{S}$ to select a hypothesis $h_s \in \mathcal{H}$ that has a small generalization error with respect to concept c.

PAC Learning Framework

We state the learning problem:

Problem (Learning)

The learner considers a fixed set of possible concepts $\mathcal{H}$, called a hypothesis set, which may not conincide with $\mathcal{C}$. Learner receives a sample $\mathcal{S} = \{x_1, \ldots, x_m\}$ drawn i.i.d according to $\mathcal{D}$ as well as labels $(c(x_1), \ldots, c(x_m))$, which are based on specific target concept $c \in \mathcal{C}$ to learn. The task is to use the labelled sample $\mathcal{S}$ to select a hypothesis $h_s \in \mathcal{H}$ that has a small generalization error with respect to concept c.

Definition (Generalization error (Mohri et al., 2012))

Given a hypothesis $h \in \mathcal{H}$, a target concept $c \in \mathcal{C}$, and an underlying distribution $\mathcal{D}$, the generalization error or risk of h is defined by

$$R(h) = \Pr_{x \sim \mathcal{D}}[h(x) \neq c(x)] = \mathop{E}_{x \sim \mathcal{D}}[1_{h(x) \neq c(x)}] \tag{15}$$

where 1_ω is the indicator function of event ω and $\mathop{E}_{x \sim \mathcal{D}}$ is the expectation over x drawn from distribution $\mathcal{D}$.

PAC Learning Framework

Distribution $\mathcal{D}$ and concept c are unknown to the learner, so we measure the empirical error:

Definition (Empirical error (Mohri et al., 2012))

Given a hypothesis $h \in \mathcal{H}$, a target concept $c \in \mathcal{C}$ and a sample $\mathcal{S} = (x_1, \ldots, x_m)$, the empirical error or empirical risk of h is defined by,

$$\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} 1_{h(x) \neq c(x)}. \tag{16}$$

The empirical error is the average error over the sample $\mathcal{S}$.

Expectation of empirical error

For a fixed hypothesis $h \in \mathcal{H}$, the expectation of the empirical error based on an i.i.d sample $\mathcal{S}$ is equal to the generalization error :

$$E[\hat{R}(h)] = R(h)$$

PAC Learning Framework

- PAC-learning framework provides theoretical limits on the size of the training sample, m, the gap between training and true errors, complexity of the hypothesis space $\mathcal{H}$ and the confidence we have in this relation (at least, $1 - \delta$).

PAC Learning Framework

The Probably Approximately Correct (PAC) learning framework:

Definition (PAC-learning (Mohri et al., 2012))

A concept class $\mathcal{C}$ is said to be PAC-learnable if there exists an algorithm $\mathcal{A}$ and a polynomial function $poly(\cdot,\cdot,\cdot,\cdot)$ such that for any $\epsilon > 0$ and $\delta > 0$, for all distributions $\mathcal{D}$ on $\mathcal{X}$ and for any target concept $c \in \mathcal{C}$, the following holds for any sample size $m \geq poly(1/\epsilon, 1/\delta, n, size(c))$:

$$\Pr_{\mathcal{S}\sim\mathcal{D}^m}[R(h_{\mathcal{S}}) \leq \epsilon] \geq 1 - \delta \tag{16}$$

If the algorithm $\mathcal{A}$ further runs in $poly(1/\epsilon, 1/\delta, n, size(c))$, then $\mathcal{C}$ is said to be efficiently PAC-learnable. The algorithm, $\mathcal{A}$ (when it exists) is called a PAC-learning algorithm for $\mathcal{C}$.

- In the definition above, n is associated with the upper bound ($\mathcal{O}(n)$) on the cost of computational representation of any element $x \in \mathcal{X}$. Similarly, size(c) is the maximal cost of the computational representation of $c \in \mathcal{C}$.

Loss functions

Loss function

"Within the machine learning literature, objective functions are usually defined in the form of loss functions, which are optimal when they are minimised. The exact form of the loss function depends on the nature of the problem to be solved, the data available and the type of machine learning algorithm being optimised. Finding appropriate loss functions is therefore one of the most important research endeavours in machine learning" (Ciampiconi, Elwood, Leonardi, Mohamed, & Rozza, 2023).

In a general machine learning problem, the aim is to learn a function $\boldsymbol{f}$ that transforms an input, defined by the input space Φ into a desirable output, defined by the output space $\mathcal{Y}$:

$$f : \Phi \mapsto \mathcal{Y}$$

where $\boldsymbol{f}$ is a function that can be approximated by a model $\boldsymbol{f}_\Theta$, parameterised by parameters, Θ.

Loss functions

Reacall:

- Given a set of inputs $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\} \in \Phi$
- Train the model with reference to target variables in output space, $\{y_0, \cdots, y_N\} \in \mathcal{Y}$

A loss function, $\mathcal{L}$, is defined as a mapping of $\boldsymbol{f}(x)_i$ along with the with the corresponding y_i to a real number $l \in \mathbb{R}$, which captures the similarity between $\boldsymbol{f}(x)_i$ and y_i.

Aggregating over all the points of the dataset we find the overall loss, $\mathcal{L}$:

$$\mathcal{L}(f|\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}, \{y_0, \cdots, y_N\}) = \frac{1}{N+1} \sum_{i=0}^{N} L(f(x_i), y_i)$$

The optimisation problem to be solved is written as:

$$\min_f \mathcal{L}(f|\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}, \{y_0, \cdots, y_N\})$$

Loss functions

The complexity of the model is often constrained by a regularisation term $R(f)$ and the optimisation becomes:

$$\min_f \frac{1}{N+1} \sum_{i=0}^{N} L(f(x_i), y_i) + R(f)$$

More generally, the model is parameterised by parameters Θ and

$$\min_\Theta \frac{1}{N+1} \sum_{i=0}^{N} L(f_\Theta(x_i), y_i) + R(\Theta)$$

We are searching the parameter space for values that minimise the loss function

Bibliography

Alpaydin, E. (2010). *Introduction to machine learning* (Second ed.). The MIT Press, Cambridge Massachusetts.

Ciampiconi, L., Elwood, A., Leonardi, M., Mohamed, A., & Rozza, A. (2023). *A survey and taxonomy of loss functions in machine learning.* online.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern classification* (Second ed.). John Wiley and Sons.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Kellleher, J. D., Namee, B. M., & D'Arcy, A. (2015). *Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples and Case Studies*. The MIT Press, Cambridge Massachusetts.

Mitchell, T. M. (1997). *Machine learning*. WCB McGraw-Hill.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). *Foundations of Machine Learning*. MIT Press.

Parnami, A., & Lee, M. (2022). *Learning from few examples: A summary of approaches to few-shot learning.* online.

Webb, A. (2002). *Statistical pattern recognition* (Second ed.). John Wiley and Sons.