Lecture 1: Machine Learning Problem

Qinfeng (Javen) Shi

28 July 2014

Intro. to Stats. Machine Learning
COMP SCI 4401/7401

Table of Contents I

Enrolment

Enrol yourself in Forum for messages, assignments and slides
Link: https://forums.cs.adelaide.edu.au/login/index.php
Go to Course “ISML-S2-2014”

Assessment

The course includes the following assessment components:

- Final written exam at 55% (open book).
- Three assignments at 15% each (report and code).

Required Skills

- Ability to program in Matlab, C/C++ is required.
- Knowing some basic statistics, probability, linear algebra and optimisation would be helpful, but not essential. They will be covered when needed.

Recommended books

1. Pattern Recognition and Machine Learning by Bishop, Christopher M.
2. Kernel Methods for Pattern Analysis by John Shawe-Taylor, Nello Cristianini
3. Convex Optimization by Stephen Boyd and Lieven Vandenberghe

Book 1 is for machine learning in general. Book 2 focuses on kernel methods with pseudo code and some theoretical analysis. Book 3 gives introduction to (Convex) Optimization.

External courses

- Learning from the Data by Yaser Abu-Mostafa in Caltech.
- Machine Learning by Andrew Ng in Stanford.
- Machine Learning (or related courses) by Nando de Freitas in UBC (now Oxford).

Machine Learning

Using data to uncover an underlying process.

Formulation

- Input: $\mathbf{x} \in \mathcal{X}$ (feature)
- Output: $y \in \mathcal{Y}$ (label)

Formulation

- Input: $\mathbf{x} \in \mathcal{X}$ (feature)
- Output: $y \in \mathcal{Y}$ (label)
- Underlying process (unknown) $f : \mathcal{X} \to \mathcal{Y}$

Formulation

- Input: $\mathbf{x} \in \mathcal{X}$ (feature)
- Output: $y \in \mathcal{Y}$ (label)
- Underlying process (unknown) $f : \mathcal{X} \to \mathcal{Y}$
- Data: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$

Formulation

- Input: $\mathbf{x} \in \mathcal{X}$ (feature)
- Output: $y \in \mathcal{Y}$ (label)
- Underlying process (unknown) $f : \mathcal{X} \to \mathcal{Y}$
- Data: $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$
  $\Downarrow$ Learn
- Decision function $g : \mathcal{X} \to \mathcal{Y}$, such that $g \approx f$.

Formulation

- Input: $\mathbf{x} \in \mathcal{X}$ (feature)
- Output: $y \in \mathcal{Y}$ (label)
- Underlying process (unknown) $f : \mathcal{X} \to \mathcal{Y}$
- Data: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$
  $\Downarrow$ Learn
- Decision function $g : \mathcal{X} \to \mathcal{Y}$, such that $g \approx f$.

For a new $\mathbf{x}'$, predict $y' = g(\mathbf{x}')$.

Examples

$\mathcal{X}$ $\mathcal{Y}$

Examples

$$\begin{array}{ccc} \mathcal{X} & & \mathcal{Y} \\ \text{(age, education, occupation, ...)} & \rightarrow & \text{income} > \$50k \text{ p.a.?} \end{array}$$

Examples

$\mathcal{X}$		$\mathcal{Y}$
(age, education, occupation, ...)	$\rightarrow$	income $>$ \$50k p.a.?
	$\rightarrow$	$\{0, 1, ..., 9\}$

Examples

$\mathcal{X}$		$\mathcal{Y}$
(age, education, occupation, ...)	$\rightarrow$	income $> \$50k$ p.a.?
	$\rightarrow$	$\{0, 1, ..., 9\}$
	$\rightarrow$	$\{John, Jenny, ...\}$

Examples

$\mathcal{X}$		$\mathcal{Y}$
(age, education, occupation, ...)	$\rightarrow$	income $>$ \$50k p.a.?
	$\rightarrow$	$\{0, 1, ..., 9\}$
	$\rightarrow$	$\{John, Jenny, ...\}$

To learn decision function $g : \mathcal{X} \rightarrow \mathcal{Y}$. What's g like?

Decision functions

Inner product

For vectors $\mathbf{x} = [x^1, x^2, \cdots, x^d]^\top, \mathbf{w} = [w^1, w^2, \cdots, w^d]^\top$, the inner product

$$\langle \mathbf{x}, \mathbf{w} \rangle = \sum_{i=1}^{d} x^i w^i.$$

We write $\mathbf{x}, \mathbf{w} \in \mathbb{R}^d$ to say they are d-dimensional real number vectors. We consider all vectors as column vectors by default.

Decision functions

Inner product

For vectors $\mathbf{x} = [x^1, x^2, \cdots, x^d]^\top, \mathbf{w} = [w^1, w^2, \cdots, w^d]^\top$, the inner product

$$\langle \mathbf{x}, \mathbf{w} \rangle = \sum_{i=1}^{d} x^i w^i.$$

We write $\mathbf{x}, \mathbf{w} \in \mathbb{R}^d$ to say they are d-dimensional real number vectors. We consider all vectors as column vectors by default.

Sign function

For any scalar $a \in \mathbb{R}$,

$$\mathrm{sign}(a) = \begin{cases} 1 & \text{if } a > 0 \\ -1 & \text{otherwise} \end{cases}$$

Decision functions

Typical decision functions for classification [1] :

$$\text{Binary-class} \quad g(\mathbf{x}; \mathbf{w}) = \operatorname{sign}(\langle \mathbf{x}, \mathbf{w} \rangle).$$

$$\text{Multi-class} \quad g(\mathbf{x}; \mathbf{w}) = \operatorname*{argmax}_{y \in \mathcal{Y}}(\langle \mathbf{x}, \mathbf{w}_y \rangle).$$

where $\mathbf{w}, \mathbf{w}_y$ are the parameters, and $\mathbf{x}, \mathbf{w}, \mathbf{w}_y \in \mathbb{R}^d$.

[1] for $b \in \mathbb{R}$, more general form $\langle \mathbf{x}, \mathbf{w} \rangle + b$ can be rewritten as $\langle [\mathbf{x}; 1], [\mathbf{w}; b] \rangle$

Decision functions

Typical decision functions for classification [1] :

$$\text{Binary-class} \quad g(\mathbf{x};\mathbf{w}) = \text{sign}(\langle \mathbf{x}, \mathbf{w} \rangle).$$

$$\text{Multi-class} \quad g(\mathbf{x};\mathbf{w}) = \operatorname*{argmax}_{y \in \mathcal{Y}}(\langle \mathbf{x}, \mathbf{w}_y \rangle).$$

where $\mathbf{w}, \mathbf{w}_y$ are the parameters, and $\mathbf{x}, \mathbf{w}, \mathbf{w}_y \in \mathbb{R}^d$.

Parameterisation

To learn g is to learn $\mathbf{w}$ or $\mathbf{w}_y$.

[1] for $b \in \mathbb{R}$, more general form $\langle \mathbf{x}, \mathbf{w} \rangle + b$ can be rewritten as $\langle [\mathbf{x}; 1], [\mathbf{w}; b] \rangle$

Types of Learning

1. Supervised Learning
2. Unsupervised Learning
3. Semi-supervised Learning

Supervised Learning

Definition

Given input-output data pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ sampled from an unknown but fixed distribution $p(\mathbf{x}, y)$, the goal is to learn $g : \mathcal{X} \to \mathcal{Y}$, $g \in \mathcal{G}$ s.t. $p(g(\mathbf{x}) \neq y)$ is small.

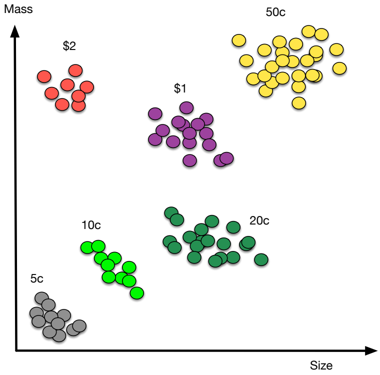
$p(g(\mathbf{x}) \neq y)$ (*i.e.* expected testing error) is generalisation error.

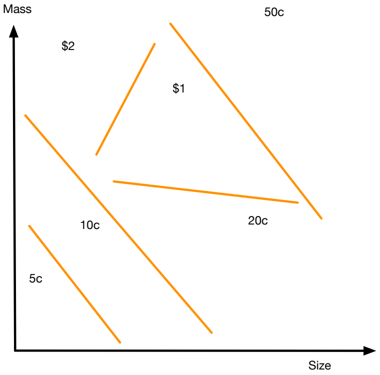
Supervised Learning

Coin recognition (vending machines and parking meters).

Supervised Learning

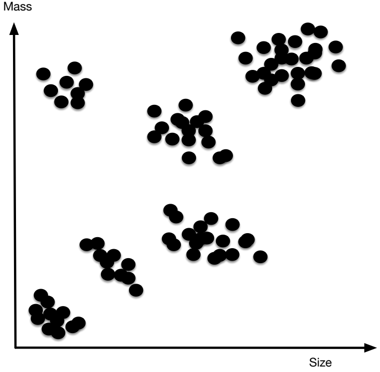
We have (input, correct output) in the training data.





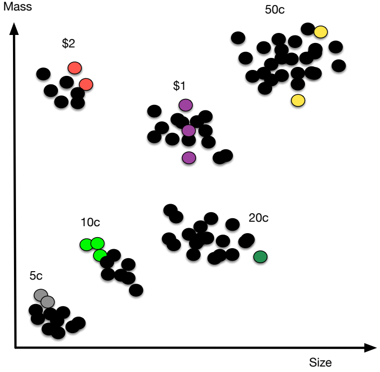
Unsupervised Learning

Instead of (input, correct output), we have (input, ?).



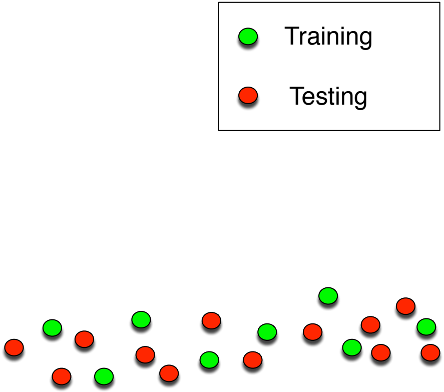
Semi-supervised Learning

We have some (input, correct output), and some (input, ?).



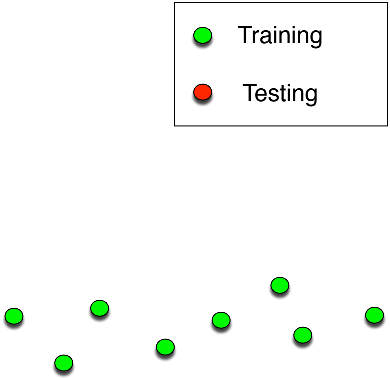
Overfitting

Fitting the training data too well cause a problem.



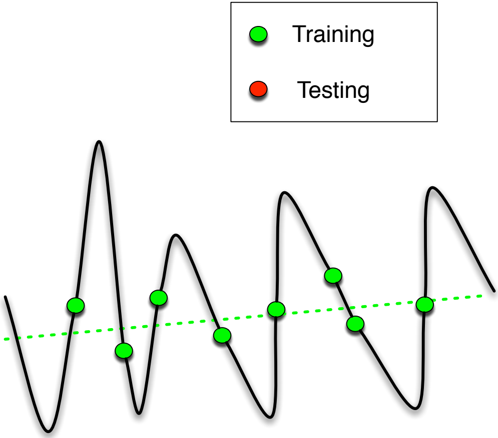
Overfitting

Train on training data (testing data are hidden from us).



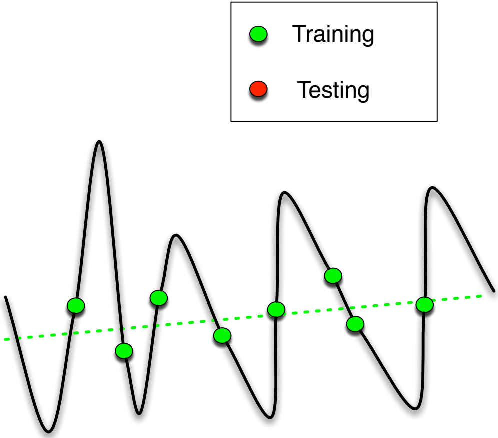
Overfitting

Two possible models. Which model fits the training data better?



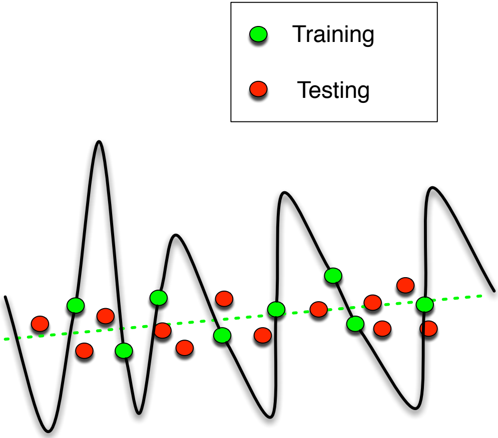
Overfitting

Two possible models. Which model fits the testing data better?



Overfitting

Reveal the testing data.



Occam's Razor

"The simplest model that fits the data is also the most plausible."

Occam's Razor

"The simplest model that fits the data is also the most plausible."

Two questions:

1. What does it mean for a model to be simple?

Occam's Razor

"The simplest model that fits the data is also the most plausible."

Two questions:

1. What does it mean for a model to be simple?
2. Why simpler is better?

Simpler means less complex

Model complexity – two types:

1. complexity of the function g: order of a polynomial, MDL
    - a straight line (order 0 or 1) is simpler than a quadratic function (order 2).
    - computer program: 100 bits simpler than 1000 bits
2. complexity of the space $\mathcal{G}$: $|\,\mathcal{G}\,|$, VC dimension, noise-fitting, ...
    - Often used in proofs.

Simpler is better

1. What do you mean by "better"?
   - smaller generalisation error (*e.g.* smaller expected testing error).
2. Why simpler is better?
   - Practically implemented by regularisation techniques, which will be covered in Lecture 2.
   - Theoretically answered by generalisation bounds, which will be covered in Learning Theory in Lecture 12.

Typical assumptions

1. Small-scale data
   - Model fits in the memory
   - Data fit in the memory or at least the disk
   - Computer is fast enough
2. $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ are independent and identically distributed (i.i.d.) samples from $p(\mathbf{x}, y)$
3. Underlying process ($f(\mathbf{x})$ or $p(\mathbf{x}, y)$) unknown but fixed

In real life things are more complex

1. ~~Small-scale~~ data
   - Large-scale $\rightarrow$ Random Projection

In real life things are more complex

1. ~~Small-scale~~ data
   - Large-scale → Random Projection
2. $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ are ~~independent~~ and ~~identically~~ distributed (i.i.d.) samples from $p(\mathbf{x}, y)$
   - Correlated → Structured Learning and Graphical Models

In real life things are more complex

1. ~~Small-scale~~ data
   - Large-scale → Random Projection
2. $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ are ~~independent~~ and ~~identically~~ distributed (i.i.d.) samples from $p(\mathbf{x}, y)$
   - Correlated → Structured Learning and Graphical Models
3. Underlying process unknown but ~~fixed~~
   - Changing environment → Online Learning (with Structured Data)

Large-scale data

Assumption 1: ~~Small-scale~~ data.

- Web topic classification: 4.4 million data, input vector 1.8 million dimensions, and output $7k$ classes?
- $\operatorname{argmax}_{y \in \mathcal{Y}}(\langle \mathbf{x}, \mathbf{w}_y \rangle)$? No! "store all $\mathbf{w}_y$" $\approx 100G$ memory.

Large-scale data

Assumption 1: ~~Small-scale~~ data.

- Web topic classification: 4.4 million data, input vector 1.8 million dimensions, and output $7k$ classes?
- $\operatorname{argmax}_{y \in \mathcal{Y}}(\langle \mathbf{x}, \mathbf{w}_y \rangle)$? No! "store all $\mathbf{w}_y$" $\approx 100G$ memory.
- Our methods:
  - Loading data, training and testing on 804, 414 news articles to predict the topics in 25.16s!
  - Training 4.4 million data in 0.5 hours (normally 2000 days).

Structured data

Assumption 2: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ are ~~independent~~.

Structured data

Assumption 2: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ are ~~independent~~.

Figure : Tennis action recognition

Most likely actions $= \operatorname{argmax}_{y_1, y_2, y_3, y_4} P(y_1, y_2, y_3, y_4 | Image)$.

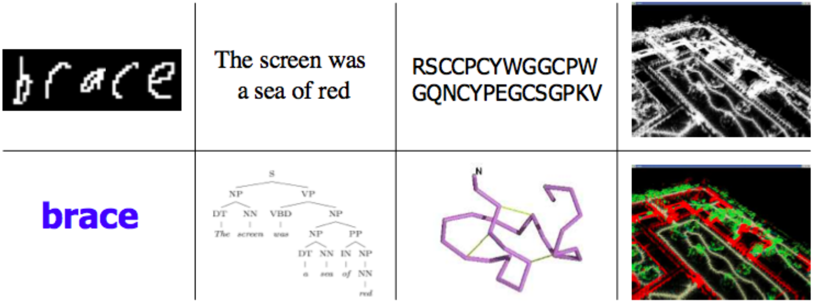
Structured data

Assumption 2: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ are ~~independent~~.

Figure : Tennis action recognition

Most likely actions $= \operatorname{argmax}_{y_1,y_2,y_3,y_4} P(y_1, y_2, y_3, y_4 | Image)$.
$\mathbf{y} = (y_1, y_2, y_3, y_4)$ is a structure of an array.

Structured data



[2]

Structured output: a sequence, a tree, or a network, ...

[2] courtesy of B. Taskar

Online Learning

Assumption 3 fails: Underlying process changes. +
Assumption 1 fails too. *i.e.* We have Large-scale data.

Online Learning

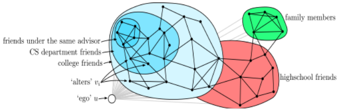
Assumption 3 fails: Underlying process changes. $+$
Assumption 1 fails too. *i.e.* We have Large-scale data.
$\Downarrow$
Online Learning (OL): predicting answers for a sequence of questions.

- processing one datum at a time (theoretical guarantee)
- no assumption on underlying process being fixed

Online Learning

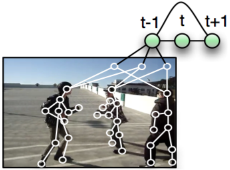
Assumption 3 fails: Underlying process changes. $+$
Assumption 1 fails too. *i.e.* We have Large-scale data.
$\Downarrow$
Online Learning (OL): predicting answers for a sequence of questions.

- processing one datum at a time (theoretical guarantee)
- no assumption on underlying process being fixed
  Problem 1: it does not scale for structured data.



(a) Social networks


(b) Fraud detection



(c) Activity recognition

That's all

Thanks!