

**COPYRIGHT WARNING:** Copyright in these original lectures is owned by Monash University. You may transcribe, take notes, download or stream lectures for the purpose of your research and study only. If used for any other purpose, (excluding exceptions in the Copyright Act 1969 (Cth)) the University may take legal action for infringement of copyright.

Do not share, redistribute, or upload the lecture to a third party without a written permission!

# **FIT 3181 Deep Learning**

## **Week 01: Mathematics and Machine Learning Revisit**

**Lecturer: Dr Lim Chern Hong**

Email: [lim.chernhong@monash.edu](mailto:lim.chernhong@monash.edu)

# Outline

- Unit Logistics
- Mathematical Background Revisit
- Machine Learning Revisit
- Summary

Unit Logistics (Will go through this in the forum)

# Unit logistics

FIT3181, 2022

## □ FIT 3181 Team

- CE/Lecturer (Clayton): **Dr Trung Le**
  - Email: trunglm@monash.edu
- Lecturer (Malaysia): **Dr Lim Chern Hong**
  - Email: lim.chernhong@monash.edu
- Head Tutor: **Mr Thanh Nguyen**
  - Email: Thanh.Nguyen4@monash.edu
- Tutor: **Mr James Tong**
  - Email: james.tong1@monash.edu
- Tutor: **Mr Anh Bui (Tony)**
  - Email: tuananh.bui@monash.edu
- Tutor: **Dr Binh Nguyen**
  - Email: binh.nguyen1@monash.edu
- Tutor: **Mr Tuan Nguyen**
  - Email: Tuan.Ng@monash.edu
- Tutor: **Mr Md Mohaimenuzzaman**
  - Email: md.mohaimen@monash.edu



Trung



Lim



Thanh



Tuan



James



Tony



Md Mohaimenuzzaman



Binh

# Lecture and consultation times

- ❑ Lectures for FIT 3181 will be pre-recorded
  - Please refer to the watch section in moodle
- ❑ Forum for FIT 3181 will be conducted via Zoom
  - Time: Tuesday 8:00 am – 9:00 am
- ❑ Consultation will be conducted via Zoom (by appointment)
  - Please email to your tutor with heading [FIT3181]
- ❑ Important notes:
  - There will be 12 Forums in total, each is 1 hour long.
  - There will be 12 labs in total, each is 2 hours long.

# Tutorials (Lab)

- Make sure you have been allocated to one tutorial.
  - **Three tutorial slots** per week for FIT3181
- **They will be on-campus**
  - **On-campus tutorial classes**
    - Make sure you know the tutorial room
- If you miss a tutorial and would like to join the next (same) tutorial, you must email the tutor for permission and detail to attend.

# Assessments

## Tests, Assignments and Exam

- **Two** in-semester online tests: **10% each**
  - During week **7** and week **12**
  - Approximately 1 hour long, online test, questions will be randomised each time.
- **Two** assignments: **20% each**
  - (Tentatively) due in week **7** and week **12** (Friday)
- **One** final exam (closed book): **40%**
  - **10** minutes reading
  - Duration: **2 hours**
- To pass this unit:

- **45%** or more in the final exam
  - **45%** or more for assignments + in-semester tests.
  - An overall unit mark of **50%** or more

# Assessments

## Tests, Assignments and Exam

- Every student is given a total quota of 4 days for late submission
  - Use this quota wisely!
- Late submission
  - Subject to marking penalty
  - 5% deduction per day once exceeding the quota
- Apply for extension:
  - Assignment extensions are only given when applied before the due date.
  - Supporting documents are required.
  - Maximum extension is 10 days.



# Communication

- Communication will be done mainly via emails and discussion (using Ed discussion via the link from Moodle site)
  - For FIT3181: email subject must start with “FIT3181: ...” and address to Dr Lim Chern Hong
    - If you don’t follow this rule, your email might be missed!
  - I/we will normally answer your email within 2 working days
- Contact your lecturer, tutor for your concerns as soon as possible.
- Strongly recommend to use Ed discussion as much as possible!

◀ Consultation Times

Forums

❖ This section contains:

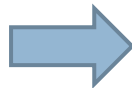
- Announcement
- Discussion Forums

❖ Announcements

❖  Unit Announcements 

❖ Discussions

❖  Ed Discussion 



ed FIT5215 SSA 2021 – Discussion

New Thread

Search

Filter

COURSES

FIT3181 S2 2021

FIT5215 S2 2021

FIT5215 SSA 2021

FIT Education Worksh... 3

FIT Lecturer Network

CATEGORIES

General

Lectures


Labs

Quizzes

Assignments

Social

Welcome! #1

Trung Le  2 days ago in General

1

Hi everyone,

We're using Ed Discussion for class Q&A.

This is the best place to ask questions about the course, whether curricular or administrative. You will get faster answers here from staff and peers than through email.

Here are some tips:

- Search before you post
- Heart questions and answers you find useful
- Answer questions you feel confident answering
- Share interesting course related content with staff and peers

For more information on Ed Discussion, you can refer to the Quick Start Guide.

All the best this semester!

Trung

Comment Edit Delete

# Learning objectives

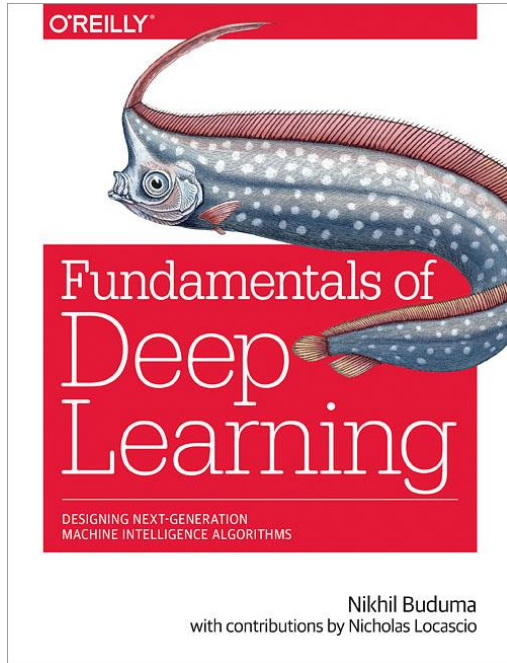
- Fundamental elements of deep learning
- Basic and advanced concepts of machine learning, AI and deep learning
- Hand-on experience with practical deep learning models and framework

# Course structure

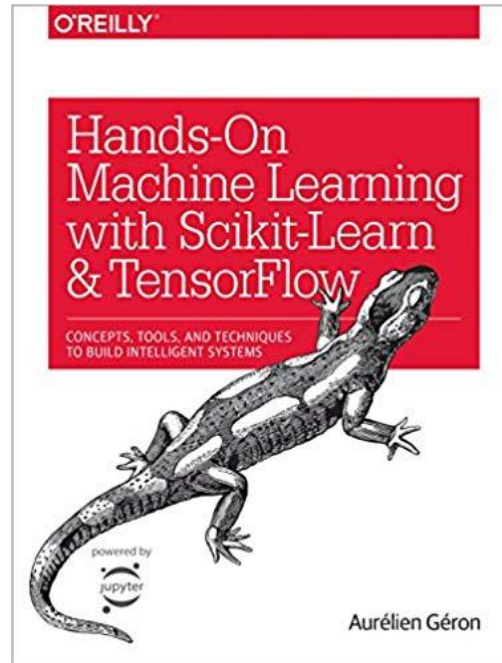
Wk	Lectures	Assessment
0	Revision and Preparation: Students are required to install Anaconda/Tensorflow before going to the tutorial 1	
1	Mathematics and Machine learning revisited	Assignment 1 released
2	Introduction to Deep Learning, from multilayer feedforward networks to modern deep neural networks (DNNs)	
3	Stochastic Gradient Descent and Optimization for Deep Learning	
4	Deep Learning for Vision (I): convolutional neural networks (CNNs)	
5	Training (very) deep networks:: optimizers and practical skills	
6	Deep Learning for Vision (II): network architectures, visualization, interpretability and robustness	Online Quiz Test 1 (10%) , Assignment 1 due (20%) Assignment 2 released
7	Deep learning for time-series and temporal data: RNNs and LSTMs	
8	Representation and deep learning for language: word2vec, Slim and Glove	
Semester Break		
9	Seq2Seq and End-to-End Deep Models	
10	Autoencoder and Variational Autoencoder (VAE)	
11	Deep generative models and GANs	Online Quiz Test 2 (10%), Assignment 2 due (20%)
12	Revision and Exam Information	

# Unit Logistics

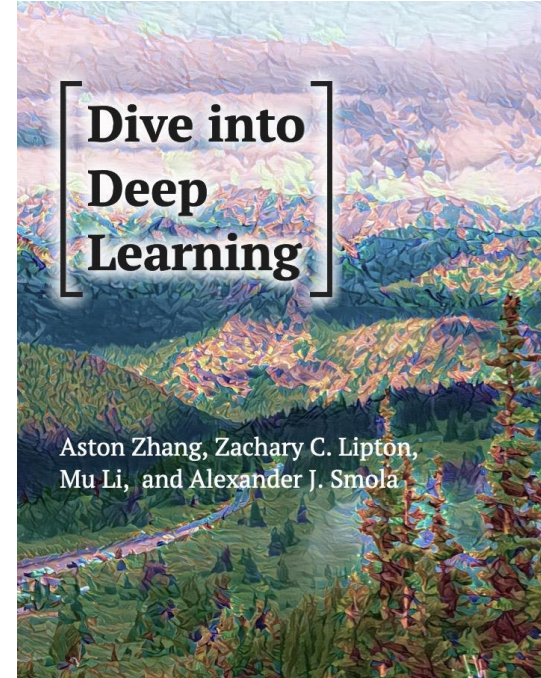
## Textbook and References



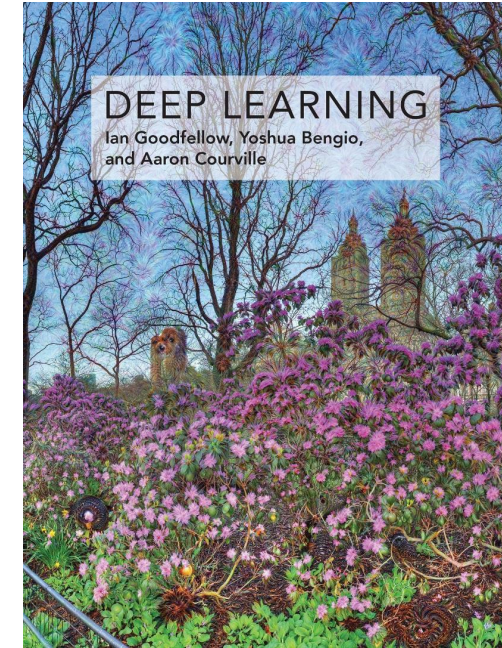
Introductory and hand-on



Intermediate and hand-on, both  
1<sup>st</sup> and 2<sup>nd</sup> edition



Intermediate and hand-on  
(but don't use tensorflow)

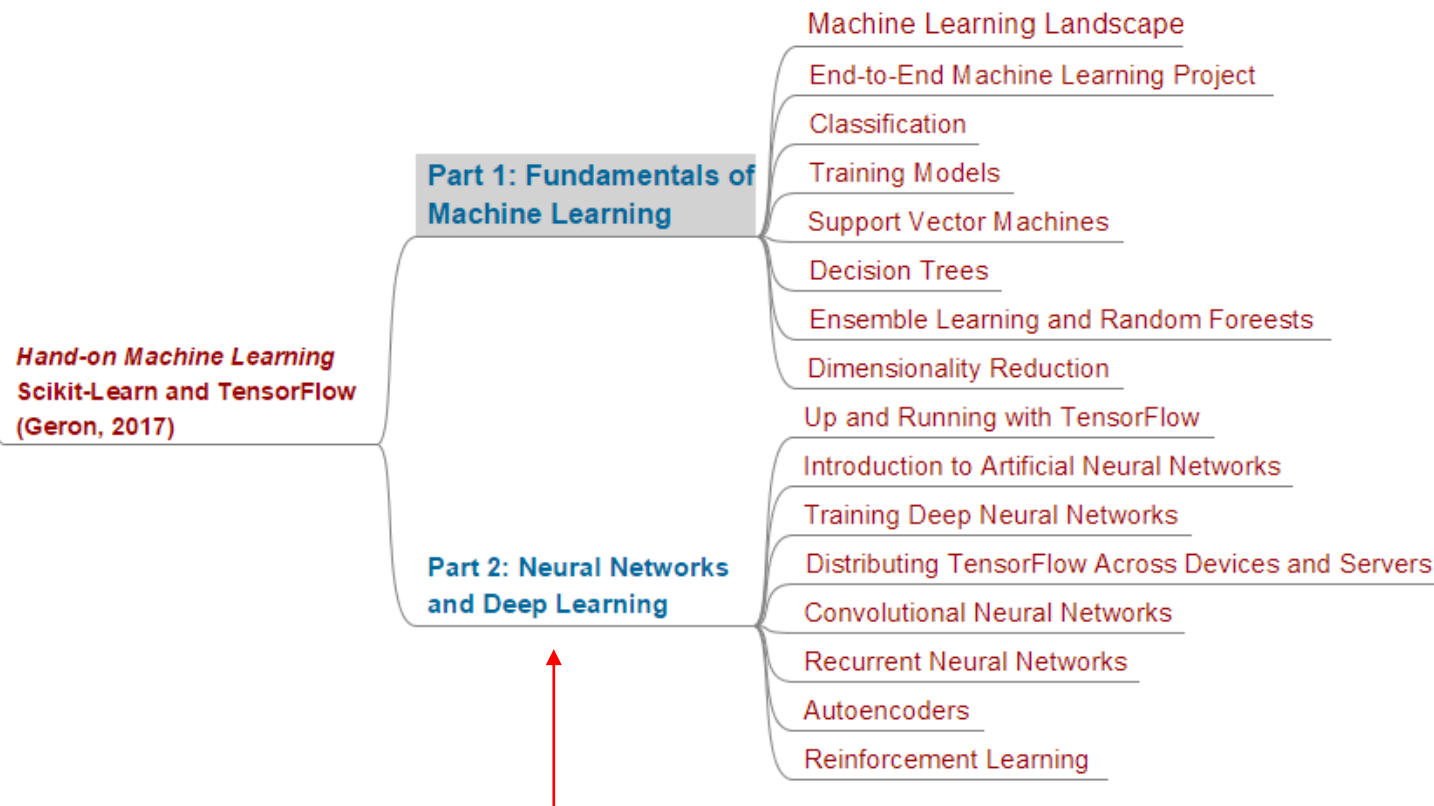


Advanced and  
theoretical

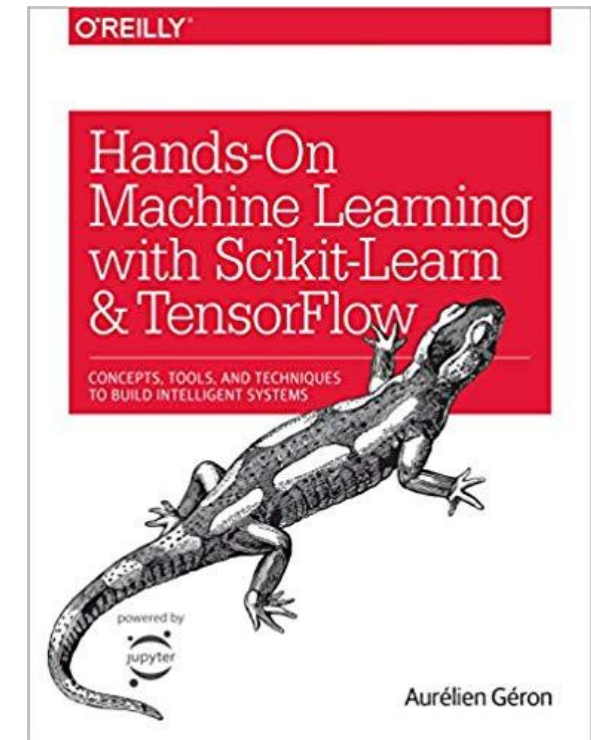
- ❑ No single textbook, some tutorial/lectures parts might use these books
- ❑ All of these books are available as online resources from Monash library.

# Unit Logistics

## Textbook and References



Deep learning part, using TF 2.x

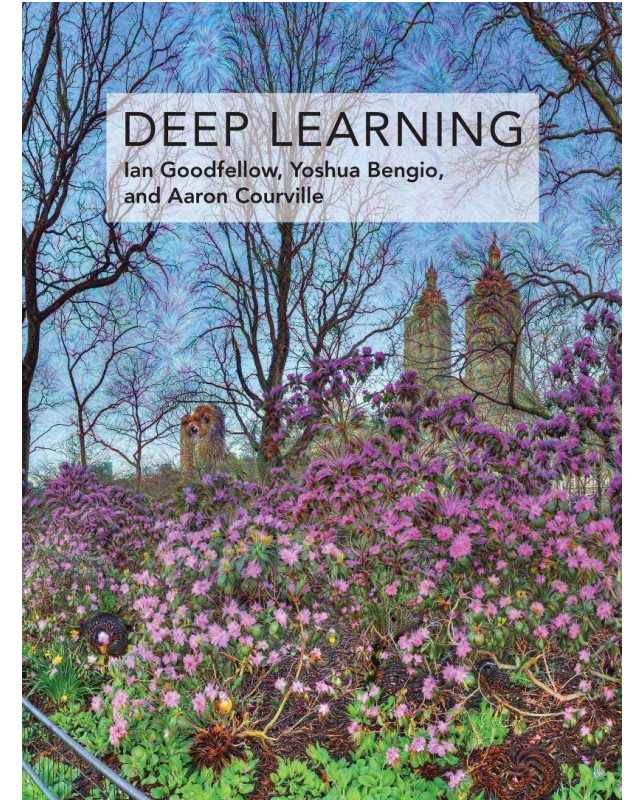
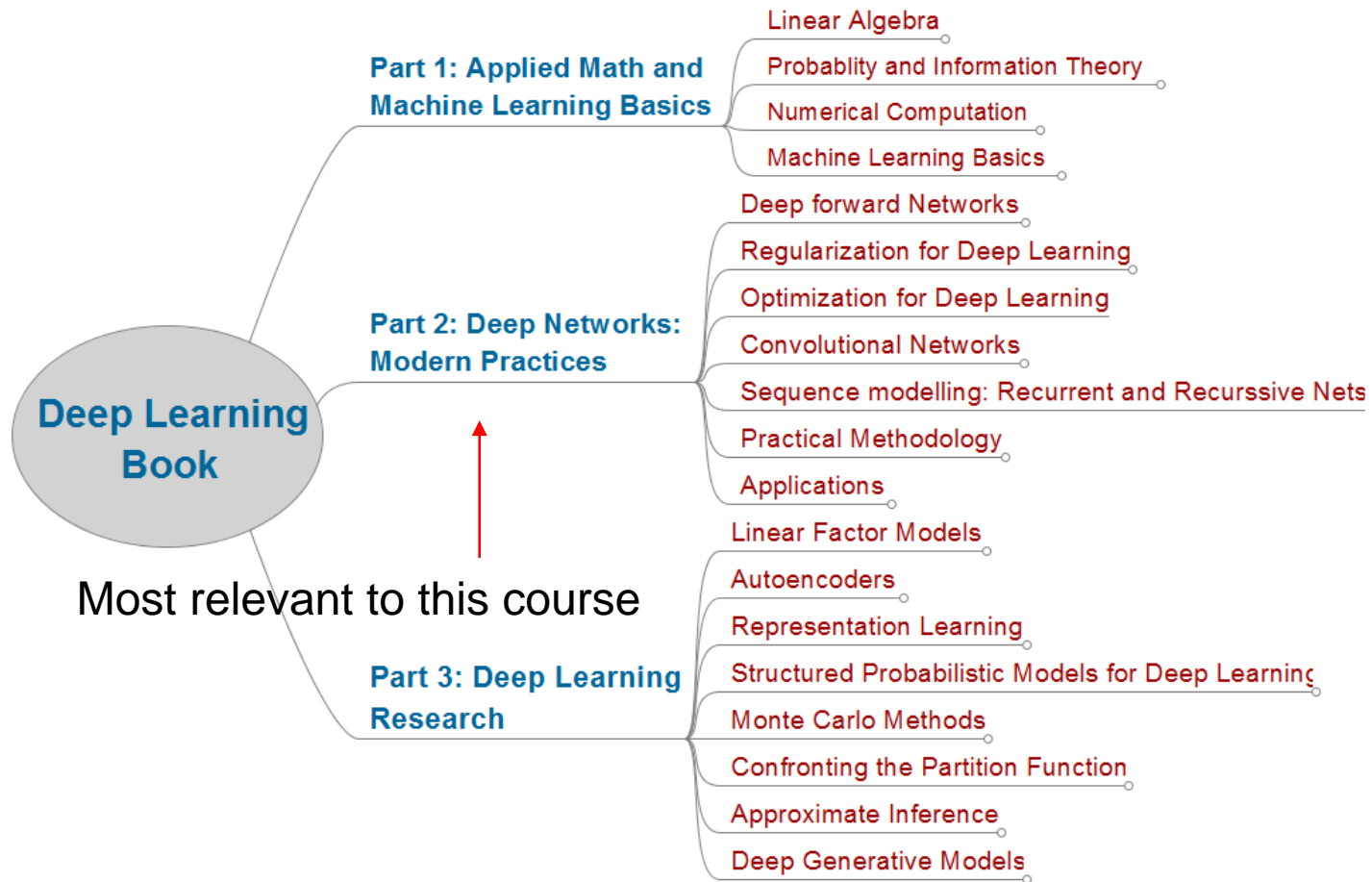


Intermediate and hand-on



# Unit Logistics

## Textbook and References



Advanced and theoretical

# Unit materials

## □ Materials available in Moodle:

- Unit handbook
- Lecture slides and subsequent recorded lectures
- News and occasional required readings
- Tutorial and hand-on practical materials:
  - See Week 00 for preparation and revision: installation, anaconda, python, numpy, scikit-learn, etc.
  - Deep learning algorithms and models (DNN, CNNs, LSTM, Word2Vec, etc). Code is given for some of them.
  - Jupyter notebooks
  - ...
- Formal assessment instructions

# Unit Logistics

## □ How to succeed in this unit?

- Work on the hand-on tutorials and master them as quickly as you can
- Where applicable, attempt all exercises and questions during lectures and tutorials
- Revise key points after each lesson to prepare for the in-semester test
- Start on your assignments early!
- Build your knowledge systematically from lectures and recommended readings
  - If it helps, use some mindmap tool such as xmind

## □ How to become extra ... ordinary?

- Read and learn beyond what is being taught
- Work on project/ideas that inspire you
- Be proactive, github and build DL profile



# Plagiarism Notice

- Plagiarism:

*“the practice of taking someone else's work or ideas and passing them off as one's own” (Oxford Dictionary)*

- It is a serious academic offence and you can be expelled from the University!





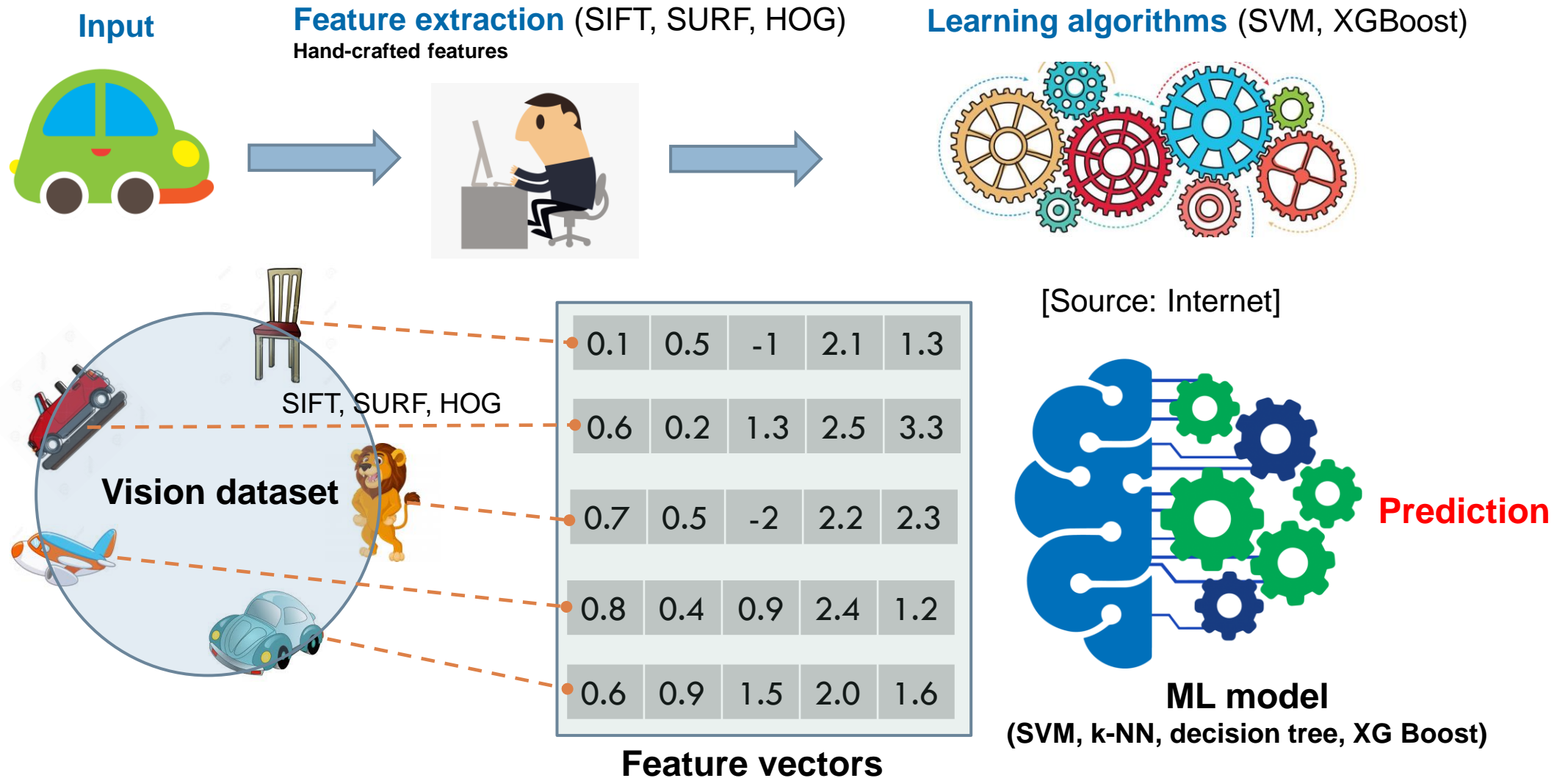
## Mathematical Background Revisited

## Linear algebra revision

# Why do we need vectors? (Forum discussion)

Hand-crafted feature

## □ Traditional approach (hand-crafted feature learning)



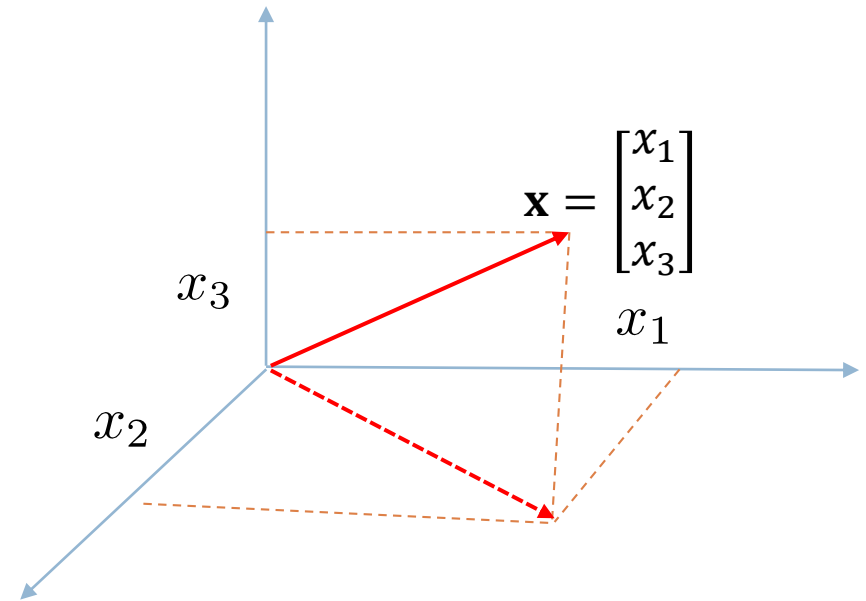
# Vector

n-dimensional vector

dimension

$$\mathbf{x} \in \mathbb{R}^n, \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$x_i : i\text{-th element}$



# Vector

## Operations on vector

transpose: column vector to row vector

$$\mathbf{x}^T = [x_1 \ x_2 \ \dots \ x_n]$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$\mathbf{x} + \mathbf{y} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_n + y_n \end{bmatrix}$$

addition

$$\mathbf{x}^T \mathbf{y} = x_1 y_1 + x_2 y_2 + \dots + x_n y_n$$

inner product

# Vector

## p-norm

$$\|\mathbf{x}\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}} \text{ with } p \geq 0$$

- $p = 0$  : how many elements in  $\mathbf{x}$  are non-zeros (sparsity)
- $p = 1$  : sum of absolute values of elements
- $p = 2$  : length of the vector
  
- $\|\mathbf{x}\|_p \geq 0$  and  $\|\mathbf{x}\|_p = 0$  if only if  $\mathbf{x} = \mathbf{0}$ .
- $\|k\mathbf{x}\|_p = |k| \|\mathbf{x}\|_p$  for any scalar  $k \in \mathbb{R}$ .
- $\|\mathbf{x} + \mathbf{y}\|_p \leq \|\mathbf{x}\|_p + \|\mathbf{y}\|_p$  (triangle inequality).

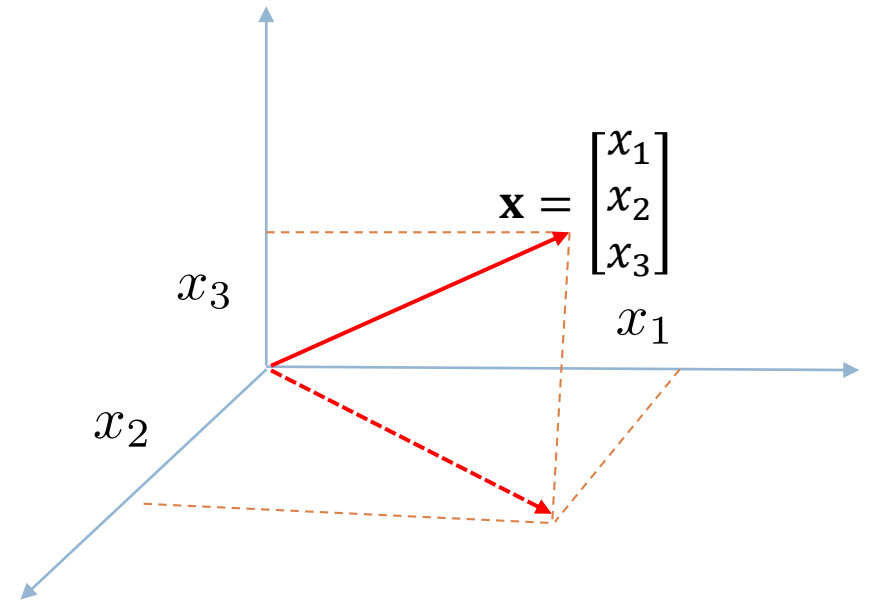


# Vector

## Length of a vector

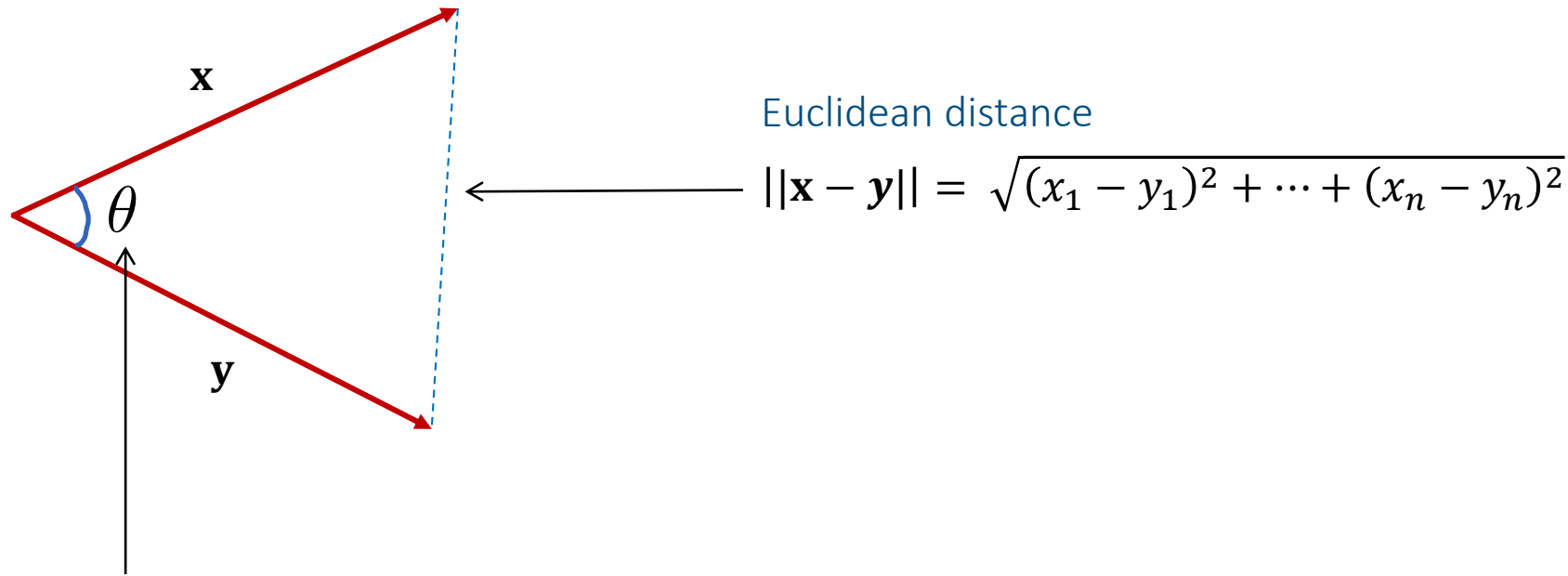
$$\text{length}(\mathbf{x}) = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

$$\|\mathbf{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$



# Vector

## Distance between two vectors



Cosine similarity

$$\cos(\theta) = \cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \mathbf{y}}{||\mathbf{x}|| ||\mathbf{y}||} = \frac{\mathbf{x}^T \mathbf{y}}{\sqrt{\mathbf{x}^T \mathbf{x}} \sqrt{\mathbf{y}^T \mathbf{y}}} \rightarrow$$

$$-1 \leq \cos(\mathbf{x}, \mathbf{y}) \leq 1$$

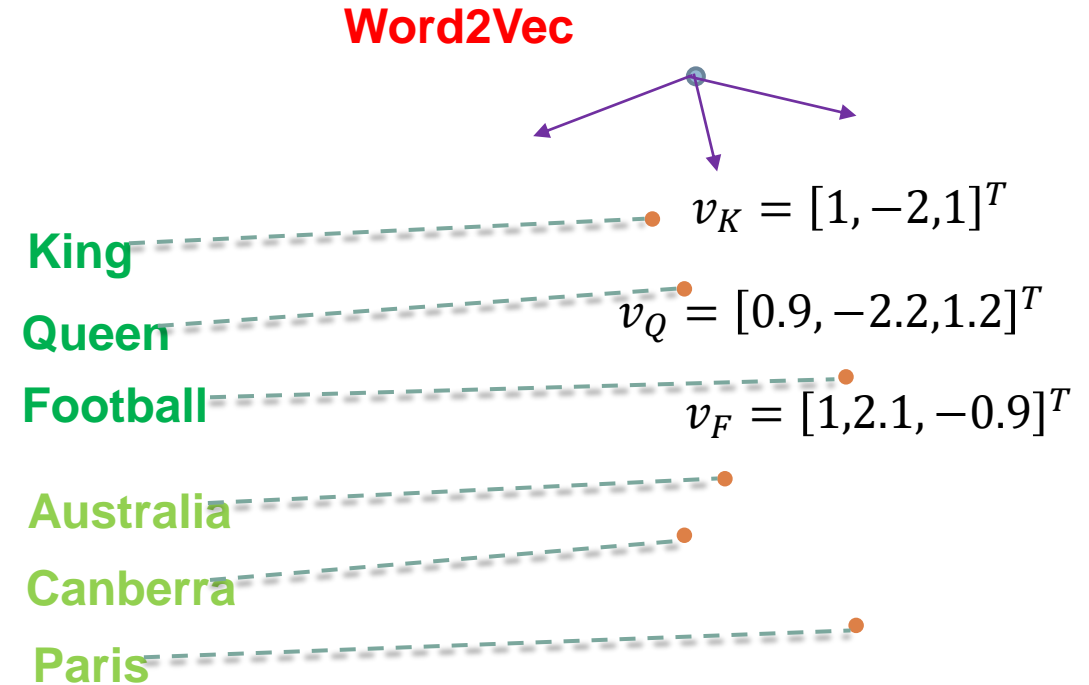
Cosine distance is defined as

$$d_{\cosine}(\mathbf{x}, \mathbf{y}) = 1 - \cos(\mathbf{x}, \mathbf{y})$$

Cosine distance can be computed via Euclidean distance if vectors are made unit vectors! (why?)

# Example of cosine similarity (Forum discussion)

- Assume that a **Word2Vec** model transforms three words **King**, **Queen**, and **Football** to three vectors  $v_K$ ,  $v_Q$ , and  $v_F$  as shown in the figure. Between two words King and Football, which word is **more similar** to word Queen?



- $$\cos\_sim(King, Queen) = \frac{v_K^T v_Q}{\|v_K\| \|v_Q\|} = \frac{1 \times 0.9 + (-2) \times (-2.2) + 1 \times 1.2}{\sqrt{1^2 + (-2)^2 + 1^2} \sqrt{0.9^2 + (-2.2)^2 + 1.2^2}} = 0.996$$
- $$\cos\_sim(Football, Queen) = \frac{v_F^T v_Q}{\|v_F\| \|v_Q\|} = \frac{1 \times 0.9 + 2.1 \times (-2.2) + (-0.9) \times 1.2}{\sqrt{1^2 + 2.1^2 + (-0.9)^2} \sqrt{0.9^2 + (-2.2)^2 + 1.2^2}} = -0.72$$
- King is **more similar** to Queen than Football.

# Matrix (2D tensor)

$$A \in \mathbb{R}^{m \times n}$$

number of columns

number of rows

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{in} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

number of columns

number of rows

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{in} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

row vector

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1j} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{m1} & \cdots & a_{nj} & \cdots & a_{mn} \end{bmatrix}$$

column vector

# Matrix operation

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{in} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

$$B = \begin{bmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{i1} & \cdots & b_{in} \\ \vdots & \ddots & \vdots \\ b_{m1} & \cdots & b_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

$$C = A^T = \begin{bmatrix} a_{11} & \cdots & a_{m1} \\ \vdots & \ddots & \vdots \\ a_{1j} & \cdots & a_{mj} \\ \vdots & \ddots & \vdots \\ a_{1n} & \cdots & a_{mn} \end{bmatrix} \in \mathbb{R}^{n \times m}$$

$C = A^T$  if only if  $C_{ij} = A_{ji}$  for any  $1 \leq i \leq n, 1 \leq j \leq m$

$$\alpha A = \begin{bmatrix} \alpha a_{11} & \cdots & \alpha a_{1n} \\ \vdots & \ddots & \vdots \\ \alpha a_{i1} & \cdots & \alpha a_{in} \\ \vdots & \ddots & \vdots \\ \alpha a_{m1} & \cdots & \alpha a_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

$$C = A + B = \begin{bmatrix} a_{11} + b_{11} & \cdots & a_{1n} + b_{1n} \\ \vdots & \ddots & \vdots \\ a_{i1} + b_{i1} & \cdots & a_{in} + b_{in} \\ \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & \cdots & a_{mn} + b_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

# Matrix multiplication

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{in} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & \cdots & b_{1j} & \cdots & b_{1p} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nj} & \cdots & b_{np} \end{bmatrix}$$

$$c_{ij} = A[i, :]B[:, j] = \sum_{k=1}^n a_{ik} b_{kj}$$

- Given **two matrices**  $A = [a_{ij}] \in \mathbb{R}^{m \times n}$  and  $B = [b_{ij}] \in \mathbb{R}^{n \times p}$ , we can compute the **multiplication**  $C = AB$  of  $A$  and  $B$  as
  - $C = [c_{ij}] \in \mathbb{R}^{m \times p}$
  - $c_{ij} = \sum_{k=1}^n a_{ik} b_{kj}$  for  $1 \leq i \leq m$  and  $1 \leq j \leq p$
  - $c_{ij} = A[i, :]B[:, j]$  which is the **dot product** of the **row  $i$**  in  **$A$**  and the **column  $j$**  in  **$B$** .

# Matrix multiplication with numpy

```
A = np.array([
    [10, 20, 30],
    [40, 50, 60]
])
A
```

array([[10, 20, 30],  
 [40, 50, 60]])

Declare matrix  $A$  with the size  $2 \times 3$

```
D = np.array([
    [ 2,  3,  5,  7],
    [11, 13, 17, 19],
    [23, 29, 31, 37]
])
D
```

array([[ 2, 3, 5, 7],  
 [11, 13, 17, 19],  
 [23, 29, 31, 37]])

Declare matrix  $D$  with the size  $3 \times 4$

```
E = A.dot(D)
E
```

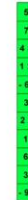
array([[ 930, 1160, 1320, 1560],  
 [2010, 2510, 2910, 3450]])

$E = A \times D$  has the size  $2 \times 4$

# Tensor

- A **tensor** is simply a **multidimensional array**
  - 0D tensor: a scalar
  - 1D tensor: a vector
  - 2D tensor: a matrix
  - 3D tensor: a 3D hyperrectangle of numerals

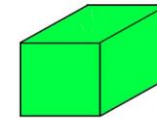
1D TENSOR /  
VECTOR



2D TENSOR /  
MATRIX

-9	4	2	5	7
3	0	12	8	61
1	23	-6	45	2
22	3	-1	72	6

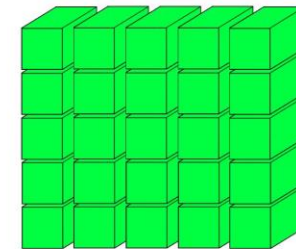
3D TENSOR /  
CUBE



-9	4	2	5	7
3	0	12	8	61
1	23	-6	45	2
22	3	-1	72	6



4D TENSOR  
VECTOR OF CUBES



5D TENSOR  
MATRIX OF CUBES

Examples of tensors. [Source: Internet]



## Calculus revision

# A small detour to calculus

● Calculus = **mathematics of change** (very important for deep learning)

● Properties of derivative:

- $f'(x) = \nabla f(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$
- $(uv)' = u'v + uv'$
- $\left(\frac{u}{v}\right)' = \frac{u'v - uv'}{v^2}$
- $(e^u)' = u'e^u$
- $(\log u)' = \frac{u'}{u}$

● Multi-variate function  $f: \mathbb{R}^n \rightarrow \mathbb{R}$  with  $y = f(x) = f(x_1, \dots, x_n)$ .

- Gradient/derivative:  $\frac{\partial f}{\partial x}(a) = \nabla_x f(a) = [\nabla_{x_1} f(a), \nabla_{x_2} f(a), \dots, \nabla_{x_n} f(a)]$ .

● Chain rule  $\otimes$  :

- $\frac{\partial u}{\partial x} = \frac{\partial u}{\partial v} \times \frac{\partial v}{\partial x}$

# Example

- Consider the function  $f(x, y, z) = \log(\exp(x) + \exp(y) + \exp(z))$ . What are  $f'_x = \nabla_x f$ ,  $f'_y = \nabla_y f$ , and  $f'_z = \nabla_z f$ ?

- $f(x, y, z) = \log(u)$  with  $u = \exp(x) + \exp(y) + \exp(z)$ .

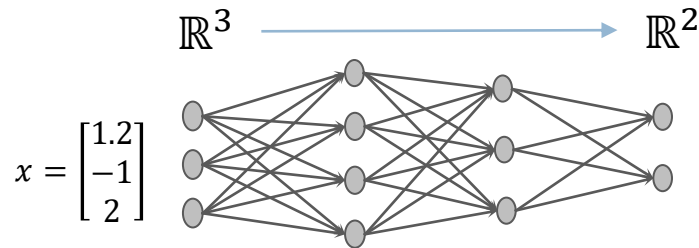
- $f'_x = \frac{u'_x}{u} = \frac{\exp(x)}{\exp(x) + \exp(y) + \exp(z)}.$

- $f'_y = \frac{u'_y}{u} = \frac{\exp(y)}{\exp(x) + \exp(y) + \exp(z)}.$

- $f'_z = \frac{u'_z}{u} = \frac{\exp(z)}{\exp(x) + \exp(y) + \exp(z)}.$

- $\nabla f = [f'_x, f'_y, f'_z] = \text{softmax}([x, y, z]).$

# Derivative for multi-variate functions



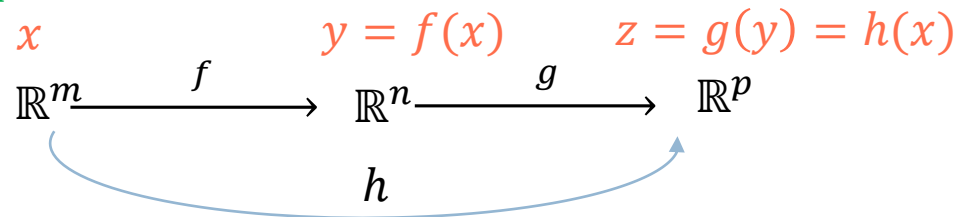
- Given a function  $f: \mathbb{R}^m \rightarrow \mathbb{R}^n$
- $f(x) = (f_1(x), \dots, f_n(x))$  where  $f_1, \dots, f_n: \mathbb{R}^m \rightarrow \mathbb{R}$  and  $x = (x_1, \dots, x_m)$ . Let denote  $y = f(x)$ .
  - The **derivative** of  $f$  at the point  $a \in \mathbb{R}^m$ , denoted by  $\nabla f(a)$  (function related notion) or  $\frac{\partial y}{\partial x}(a)$  (variable related notion) is a matrix  $n$  by  $m$  (i.e., the Jacobian matrix).

$$\frac{\partial y}{\partial x}(a) = \nabla f(a) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(a) & \dots & \dots & \frac{\partial f_1}{\partial x_j}(a) & \dots & \frac{\partial f_1}{\partial x_m}(a) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{\partial f_i}{\partial x_1}(a) & \dots & \dots & \frac{\partial f_i}{\partial x_j}(a) & \dots & \frac{\partial f_i}{\partial x_m}(a) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1}(a) & \dots & \dots & \frac{\partial f_n}{\partial x_j}(a) & \dots & \frac{\partial f_n}{\partial x_m}(a) \end{bmatrix}^n_m$$

**Jacobian matrix**

## □ Chain rule ∞

- Given a function  $f: \mathbb{R}^m \rightarrow \mathbb{R}^n$ ,  $g: \mathbb{R}^n \rightarrow \mathbb{R}^p$ , denote  $h = g \circ f: \mathbb{R}^m \rightarrow \mathbb{R}^p$ , meaning that  $h(x) = g(f(x))$ . We further define  $y = f(x)$  and  $z = g(y) = g(f(x)) = h(x)$ .
- For  $x \in \mathbb{R}^m$ ,  $\nabla h(x) = \nabla g(f(x)) \times \nabla f(x)$  or equivalently  $\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \cdot \frac{\partial y}{\partial x}$ .



# Example

$$\square \quad y = f(x) = f(x_1, x_2, x_3) = (x_1^2 + x_2^2, x_2^2 + x_3^2 x_2)$$

- $f: \mathbb{R}^3 \rightarrow \mathbb{R}^2$
- $f_1(x) = f_1(x_1, x_2, x_3) = x_1^2 + x_2^2$
- $f_2(x) = f_2(x_1, x_2, x_3) = x_2^2 + x_3^2 x_2$
- $\frac{\partial y}{\partial x} = \nabla f \in \mathbb{R}^{2 \times 3}$

$$\frac{\partial y}{\partial x} = \nabla_x f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 2x_1 & 2x_2 & 0 \\ 0 & 2x_2 + x_3^2 & 2x_2 x_3 \end{bmatrix}$$

# Example

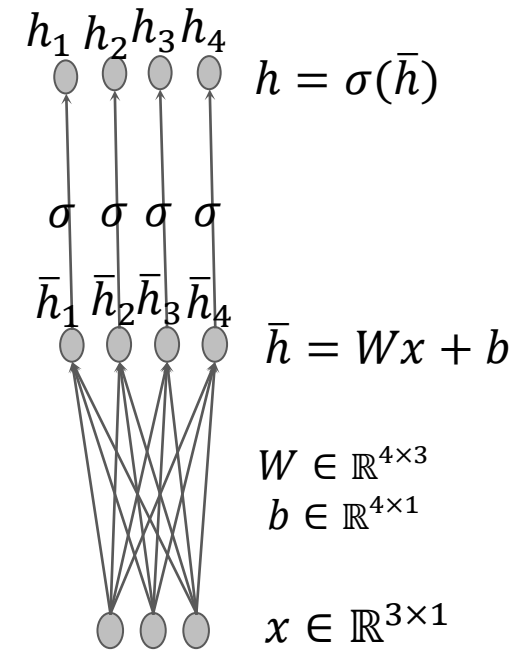
$$\square \quad \bar{h} = Wx + b \text{ and } h = \sigma(\bar{h})$$

- $h = \sigma(Wx + b)$
- $\sigma$  is the **activation function**

$$\square \quad \frac{\partial h}{\partial x} = \frac{\partial h}{\partial \bar{h}} \times \frac{\partial \bar{h}}{\partial x} = \text{diag}(\sigma'(\bar{h})) W$$

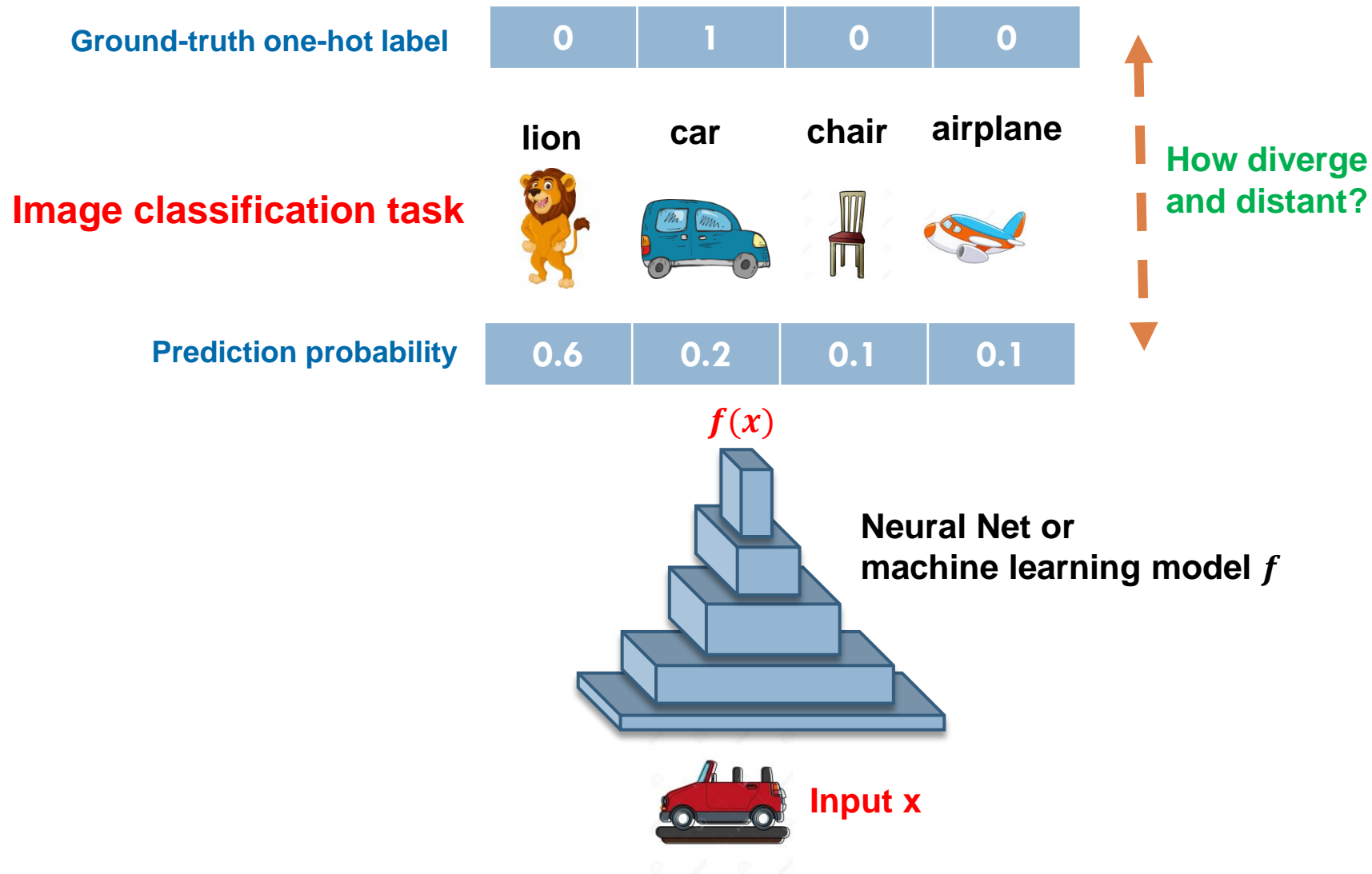
$$\square \quad \frac{\partial h}{\partial \bar{h}} = \begin{bmatrix} \frac{\partial h_1}{\partial \bar{h}_1} & \frac{\partial h_1}{\partial \bar{h}_2} & \frac{\partial h_1}{\partial \bar{h}_3} & \frac{\partial h_1}{\partial \bar{h}_4} \\ \frac{\partial h_2}{\partial \bar{h}_1} & \frac{\partial h_2}{\partial \bar{h}_2} & \frac{\partial h_2}{\partial \bar{h}_3} & \frac{\partial h_2}{\partial \bar{h}_4} \\ \frac{\partial h_3}{\partial \bar{h}_1} & \frac{\partial h_3}{\partial \bar{h}_2} & \frac{\partial h_3}{\partial \bar{h}_3} & \frac{\partial h_3}{\partial \bar{h}_4} \\ \frac{\partial h_4}{\partial \bar{h}_1} & \frac{\partial h_4}{\partial \bar{h}_2} & \frac{\partial h_4}{\partial \bar{h}_3} & \frac{\partial h_4}{\partial \bar{h}_4} \end{bmatrix} = \begin{bmatrix} \sigma'(\bar{h}_1) & 0 & 0 & 0 \\ 0 & \sigma'(\bar{h}_2) & 0 & 0 \\ 0 & 0 & \sigma'(\bar{h}_3) & 0 \\ 0 & 0 & 0 & \sigma'(\bar{h}_4) \end{bmatrix} = \text{diag}(\sigma'(\bar{h}))$$

$$\square \quad \frac{\partial \bar{h}}{\partial x} = W$$



Information theory revision

# Why need to work with discrete distributions?





# Revision of basic informatic quantities

- Given two **discrete distributions**  $p = [p_i]_{i=1}^d$  ( $p_i \geq 0$  and  $\sum_{i=1}^d p_i = 1$ ) and  $q = [q_i]_{i=1}^d$  ( $q_i \geq 0$  and  $\sum_{i=1}^d q_i = 1$ ).
  
- **Kullback-Leibler (KL) divergence between  $p, q$** 
  - $KL(p, q) = \sum_{i=1}^d p_i \log \frac{p_i}{q_i}$
  - $KL(p, q) \geq 0$  and  $KL(p, q) = 0$  if only if  $p = q$
  
- **Entropy of the distribution  $p$** 
  - $H(p) = \sum_{i=1}^d p_i \log \frac{1}{p_i} = -\sum_{i=1}^d p_i \log p_i$  where  $0 \times \log 0 = 0$
  
- **Cross-entropy (CE) divergence between  $p, q$** 
  - $CE(p, q) = -\sum_{i=1}^d p_i \log q_i$ .
  - $CE(p, q) = KL(p, q) + H(p)$ .

# Example

- Consider  $p = [0.1, 0.4, 0.5]$  and  $q = [0.2, 0.5, 0.3]$ . Compute  $KL(p, q)$ ,  $CE(p, q)$ , and  $H(p)$ .

- $KL(p, q) = 0.1 \times \log \frac{0.1}{0.2} + 0.4 \times \log \frac{0.4}{0.5} + 0.5 \times \log \frac{0.5}{0.3} \approx 0.042$
- $CE(p, q) = -0.1 \times \log 0.2 - 0.4 \times \log 0.5 - 0.5 \times \log 0.3 \approx 0.452$
- $H(p) = 0.1 \times \log \frac{1}{0.1} + 0.4 \times \log \frac{1}{0.4} + 0.5 \times \log \frac{1}{0.5} \approx 0.41$

- Consider  $p = [0, 1, 0]$  and  $q = [0.2, 0.3, 0.5]$ . Compute  $CE(p, q)$ .

- $CE(p, q) = -0 \times \log 0.2 - 1 \times \log 0.3 - 0 \times \log 0.5 \approx -\log 0.3 = 0.523$

❓ **Question:** when the entropy is maximized or minimized?

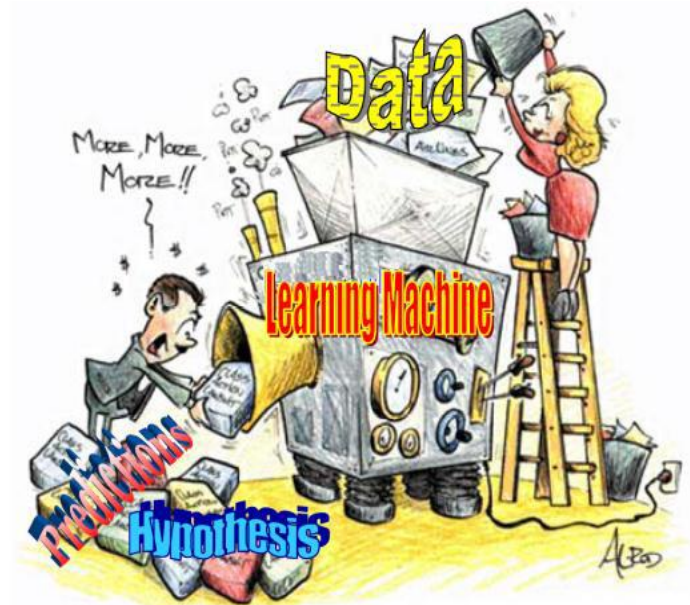


# Machine Learning Revisited

# Machine learning

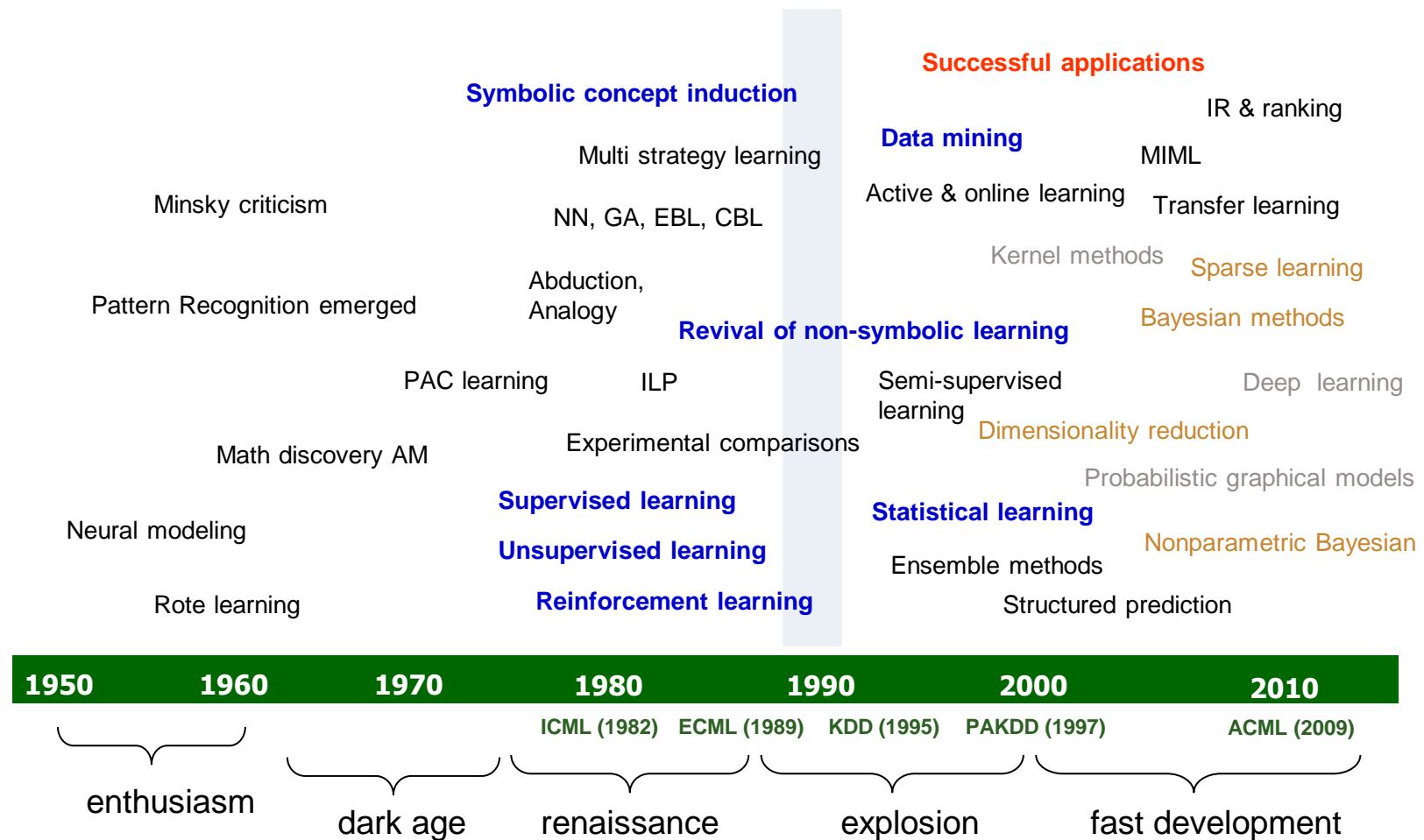
- “Field of study that gives computers the ability to learn without being explicitly programmed” (Arthur Samuel, 1959).
- “Machine learning sits at the crossroads of computer science, statistics and a variety of other disciplines concerned with automatic improvement over time, and inference and decision-making under uncertainty.” (Jordan & Michell, 2015).
- “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  $E$ ”, (Tom Mitchell, 1997)

M.I. Jordan and T. Mitchell, “Machine Learning: Trends, perspectives, and prospects”, *Science*, 17 July 2015.



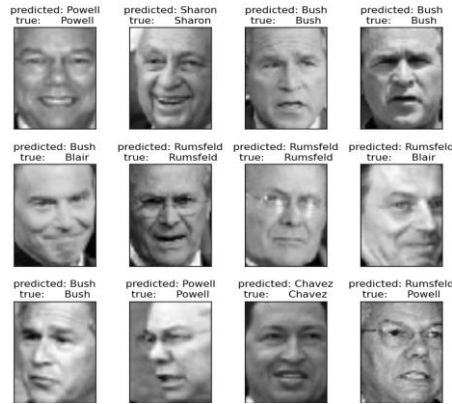
(from Eric Xing lecture notes)

# A brief history of ML



[credit: Prof Bao Ho, JAIST]

# Machine learning



Face recognition system



Detecting Spam Emails



Detect credit card fraud

## Experience E

dataset of thousands of known faces

## Task T

given a new photo, recognise the name of the face

Label set: names of all people.

## Performance P

how **accurate** the recognition is  
Measure: **Accuracy**



## Experience E

Examples of spam emails and not-spam emails

## Task T

To assign a label "spam" or "not-spam" to an email  
Label set: **spam** and **not spam**

## Performance P

how **accurate** spam email can be detected  
Measure: **accuracy**

## Experience E

Data collected for credit-card transactions deemed as fraud and not-fraud

## Task T

To assign a label 'fraud' or "not fraud" to a given credit-card transaction  
Label set: **fraud** and **not fraud**

## Performance P

how accurate a credit-card fraud transaction can be detected.  
Measure: **accuracy**

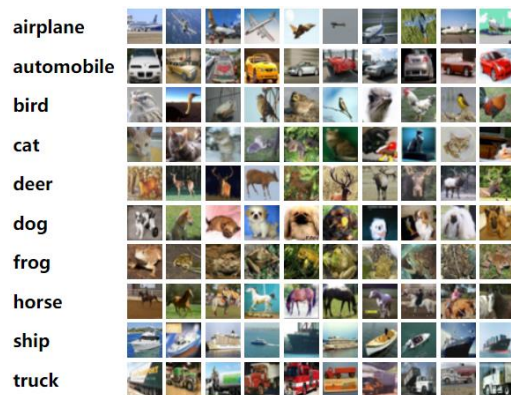


# Feature extraction (Forum discussion)

- Most machine learning algorithms (e.g, Support Vector Machine or XG Boosts) only **work with numeric data**

- Given **structural data** such as images or texts, we **extract numeric features** that preserve characteristics of raw data
- **Feature extraction methods**
  - **Images:** SIFT, SURF, and HOG.
  - **Texts:** bag of words and TF-IDF.

**Cifar10**



SIFT, SURF, HOG

TF-IDF

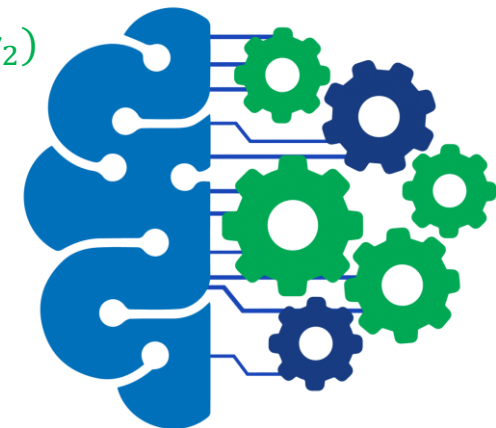
**Numeric Dataset**

Feature vectors ( $x$ ) + Labels ( $y$ )

0.1	0.5	-1	2.1	1.3	$(x_1, y_1)$
0.6	0.2	1.3	2.5	3.3	$(x_2, y_2)$
0.7	0.5	-2	2.2	2.3	.
0.8	0.4	0.9	2.4	1.2	.
0.6	0.9	1.5	2.0	1.6	.
0.1	0.8	-1	-2	1.3	.
0.2	0.3	-4	2.1	1.6	$(x_N, y_N)$

**ML model**

(SVM, k-NN, decision tree, XG Boost)



**Prediction**

Loves the German bakeries in Sydney. Together with my imported honey it feels like home	Positive
@VivaLaLauren Mine is broken too! I miss my sidekick	Negative
Finished fixing my twitter...I had to unfollow and follow everyone again	Negative
@DinahLady I too, liked the movie! I want to buy the DVD when it comes out	Positive
@frugaldougal So sad to hear about @OscarTheCat	Negative
@Mofette brilliant! May the fourth be with you #starwarsday #starwars	Positive
Good morning thespians a bright and sunny day in UK, Spring at last	Positive
@DowneyisDOWNEY Me neither! My laptop's new, has dvd burning/ripping software but I just can't copy the files somehow!	Negative

**Sentiment analysis**



# Elements of machine learning

## ① Data

Vector  $x \in \mathbb{R}^d$

Image  $x$

Sentence  $x$

Speech  $x$

Ground-truth label  $y \in Y$

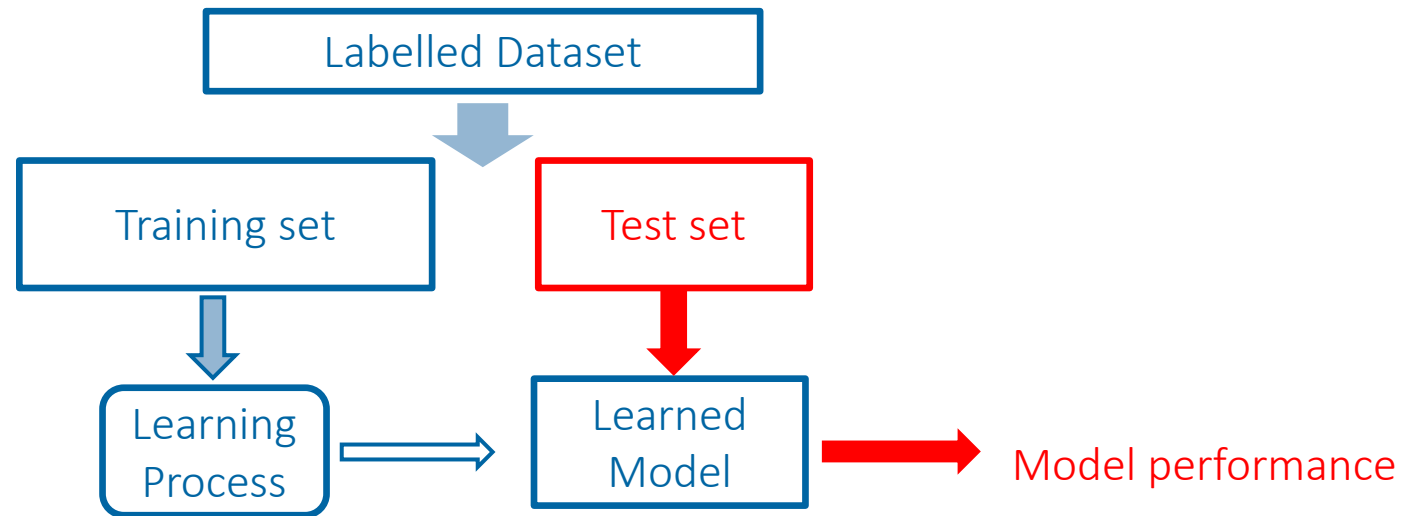


## ② Model

- $f: X \rightarrow Y$
- $X$  is **data space**,  $Y$  is **label space**
- **Classification**:  $Y = \{1, 2, \dots, M\}$
- **Regression**:  $Y = \mathbb{R}$
- **Prediction**:  $f(x) = \hat{y} \in Y$

## ③ Assessment

- How **good** is the model doing its tasks?
- **Performance metrics**
  - Accuracy, Recall, Precision, F-score
  - $Accuracy = \frac{\#Correct\ Predictions}{\#All\ Predictions}$   
: % of correctly classified examples



Machine learning pipeline.

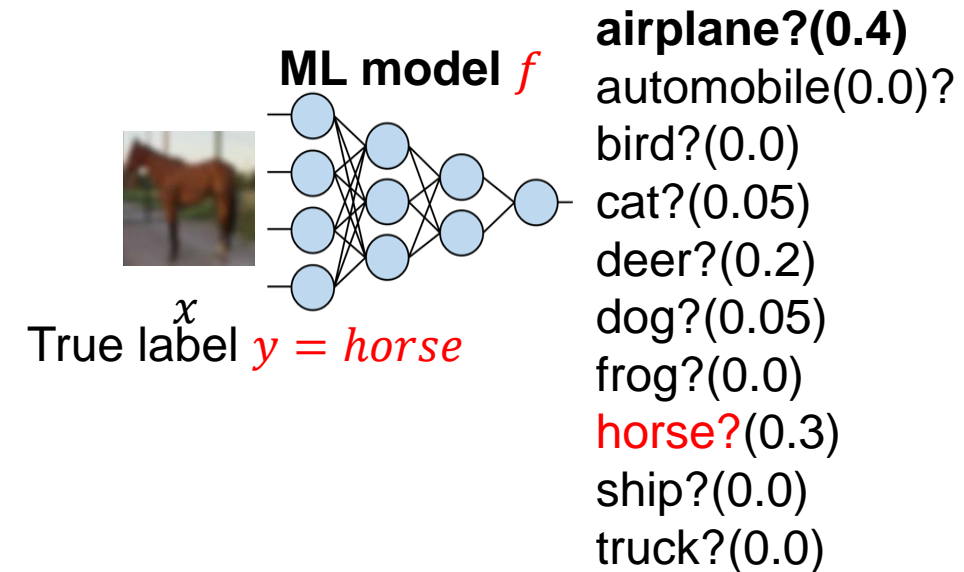
# Classification and Regression

## □ Two main tasks in machine learning

1. **Classification** and **regression**
2. Other tasks in ML include density estimation, clustering, anomaly detection, recommendation, forecasting, and ranking

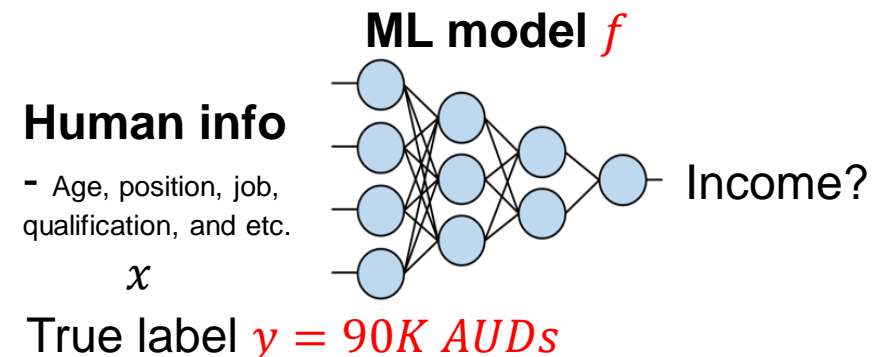
## □ Classification

1. The label set  $Y = \{1, 2, \dots, M\}$  and the data space  $X$  (vectors, images).
2. Training set  $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$  where  $x_i \in \mathbb{R}^d$  and  $y_i \in Y = \{1, 2, \dots, M\}$
3. Learn a model  $f: X \rightarrow Y$
4. Given data example  $x$ , render the prediction  $\hat{y} = f(x) \in \{1, 2, \dots, M\}$



## □ Regression

1. The label set  $Y = \mathbb{R}$  and the data space  $X$  (vectors, images).
2. Training set  $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$  where  $x_i \in \mathbb{R}^d$  and  $y_i \in Y = \mathbb{R}$
3. Learn a model  $f: X \rightarrow Y$
4. Given data example  $x$ , render the prediction  $\hat{y} = f(x) \in \mathbb{R}$



# Supervised learning (Forum discussion)

**Problem:** Learn a **function** from **data** to **relate the inputs** with **outputs**. Training data include output information (labels).

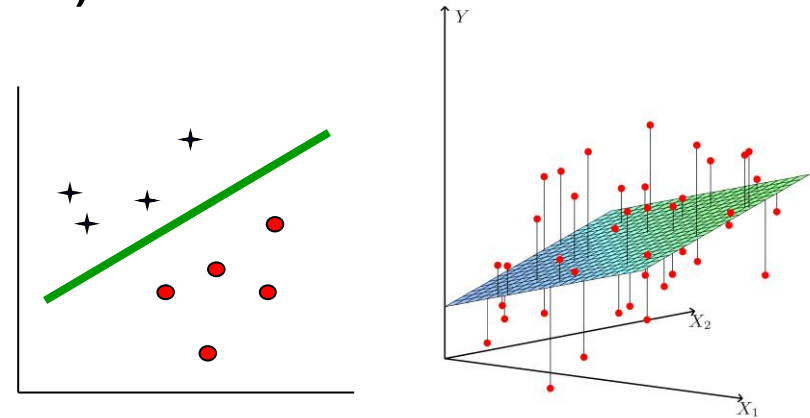
**Data:**  $D = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$

**Function:**  $f: X \rightarrow Y$

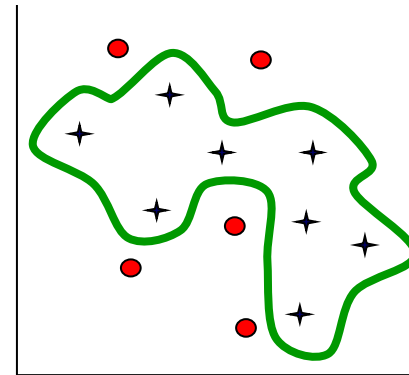
$x \in \mathbb{R}^d$  = feature

$y \in \{1, 2, \dots, M\}$  = a **discrete** label (**classification**),

$y \in \mathbb{R}$  = a **continuous** value (**regression**)



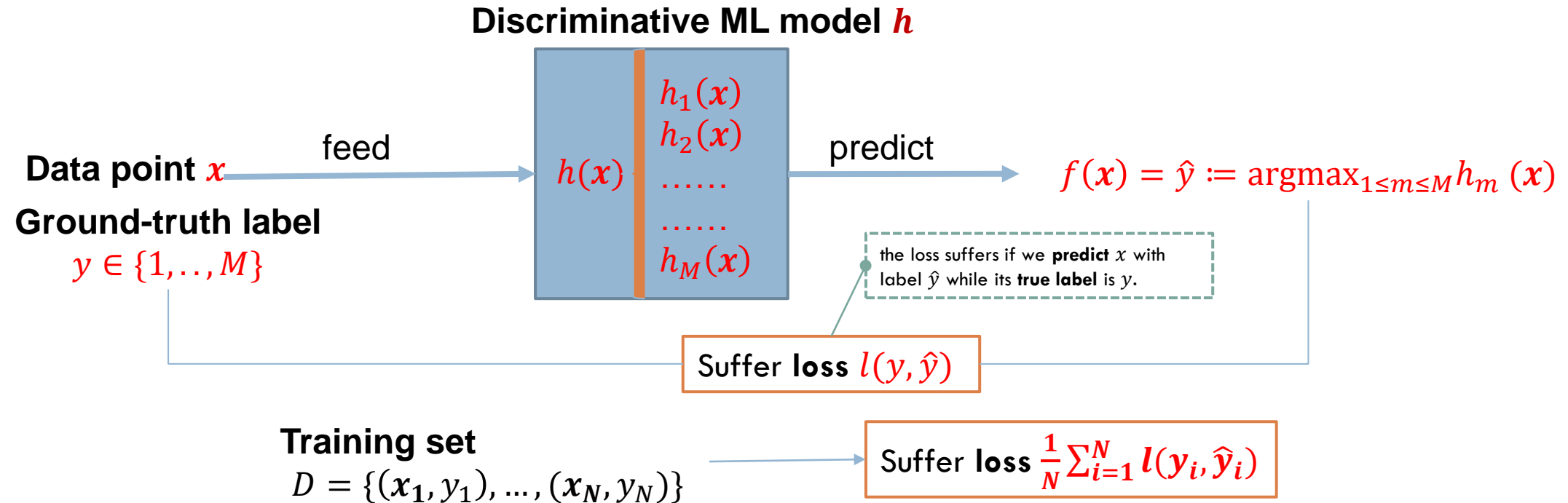
e.g., linear functions



e.g., nonlinear functions

# Discriminative machine learning

## Classification



- A data point  $x$  receives **discriminative values**  $h_1(x), \dots, h_M(x)$  from the model
  - $h_m(x)$  represents the **possibility** to **classify**  $x$  to **class**  $m$  for  $1 \leq m \leq M$
- We use these discriminative values to predict the label  $\hat{y}$  as
  - $\hat{y} = \operatorname{argmax}_{1 \leq m \leq M} h_m(x)$ , meaning the class with highest discriminative value
- The prediction  $x$  with the predicted label  $\hat{y}$  suffers a loss
  - $l(\hat{y}, y)$  where  $l$  is a **loss function** (if  $\hat{y} = y$  then  $l(\hat{y}, y) = 0$ ).
- Given a training set  $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$ , the loss incurred is
  - $\frac{1}{N} \sum_{i=1}^N l(y_i, \hat{y}_i)$

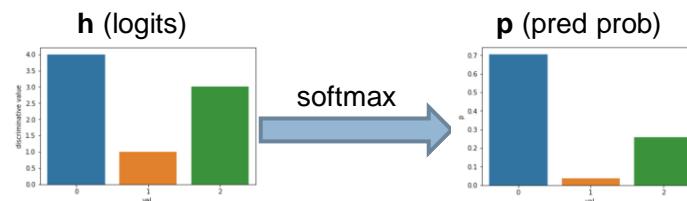
# Softmax activation function

- Use to transform **real-valued discriminative scores** to **discrete probabilities**

- $h = [h_m]_{m=1}^M \rightarrow p = \text{softmax}(h) := \left[ \frac{\exp\{h_m\}}{\sum_{i=1}^M \exp\{h_i\}} \right]_{m=1}^M$
- $p = [p_m]_{m=1}^M$  becomes a **distribution** over **classes**  $\{1, \dots, M\}$ 
  - $p_m \geq 0$  ( $1 \leq m \leq M$ ) and  $\sum_{m=1}^M p_m = 1$ .

- Assume that we do **sentiment classification** (analysis)

- Data point  $x$  is a sentence (e.g.,  $x = \text{"it is a beautiful day today"}$ ), we need to predict **sentiment** of this sentence
- Three **sentiment labels**: **positive** (happy, class 1), **negative** (sad, class 2), **neural** (none happy and sad, class 3)
- The model gives three **discriminative values** for  $x$ 
  - $h_1 = 2, h_2 = -1, h_3 = 1$  means that **highest possibility** to **classify**  $x$  to the **class 1** with the label **positive**.
- We apply **softmax** function on the **discriminative scores**  $h$ 
  - $p = \text{softmax}(h)$ 
    - $p_1 = \frac{\exp\{2\}}{\exp\{2\} + \exp\{-1\} + \exp\{1\}} \approx 0.705$ ,  $p_2 = \frac{\exp\{-1\}}{\exp\{2\} + \exp\{-1\} + \exp\{1\}} \approx 0.035$ ,  $p_3 = \frac{\exp\{1\}}{\exp\{2\} + \exp\{-1\} + \exp\{1\}} \approx 0.259$
  - $p = [0.70538451, 0.03511903, 0.25949646]$  are the **probabilities** to **classify**  $x$  to the **classes 1, 2, 3** respectively.



**Softmax** transforms real-valued discriminative scores (ranged in  $(-\infty, +\infty)$ ) to probability values (ranged in  $[0, 1]$ ) but preserving the order.

# Cross-entropy loss

- Given  $M$  labels  $1, 2, \dots, M$ , a label  $y \in \{1, \dots, M\}$  has two equivalent forms:
  - **Numeric form:**  $y \in \{1, \dots, M\}$  is an **index** in  $\{1, \dots, M\}$ .
  - **One-hot vector form:**  $\mathbf{1}_y = [0, \dots, 0, \mathbf{1}_y, 0, \dots, 0]$  - the vector of **zeros** except **an unique 1** in the  $y$ -th position.
  
- Label set  $\{happy, sad, neural\}$  with 1: *happy*, 2: *sad*, and 3: *neural*
  - $[1, 0, 0] \rightarrow$  happy (numeric label 1),  $[0, 1, 0] \rightarrow$  sad (numeric label 2),  $[0, 0, 1] \rightarrow$  neural (numeric label 3)
  
- Given two **discrete distributions** over classes:  $\mathbf{p} = [p_m]_{m=1}^M$  and  $\mathbf{q} = [q_m]_{m=1}^M$ , the **cross-entropy divergence** between  $p$  and  $q$ 
  - $CE(p, q) := -\sum_{m=1}^M p_m \log q_m$  which measures how **far (divergent)** it is between  $p$  and  $q$ .
  - $CE(p, q) \geq H(p) := -\sum_{m=1}^M p_m \log p_m$  ( $H$  is the **Shannon entropy**).

# Cross-entropy loss

## □ Cross-entropy loss

- $l(y, \hat{y}) = \text{CE}(1_y, p) = -\log p_y$
- $y \in \{1, \dots, M\}$  is the **ground-truth label** of a given  $x$  and  $p$  is the **prediction probabilities** of  $x$  to classes

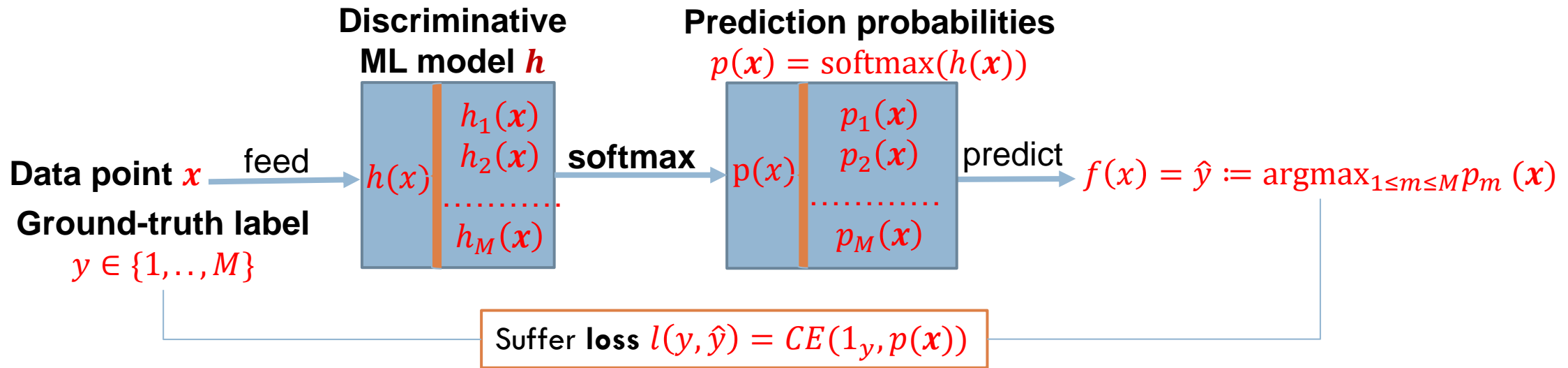
- Given a **sentence**  $x$  with the **label positive/happy** (the class 1), assume that our model predicts it with **prediction probabilities**  $p = [0.3, 0.4, 0.3]$ , the **cross-entropy loss** for this prediction

- $l(y, \hat{y}) = \text{CE}([1, 0, 0], [0.3, 0.4, 0.3]) = -1 \cdot \log 0.3 - 0 \cdot \log 0.4 - 0 \cdot \log 0.3 = -\log 0.3 \approx 1.204$

❓ **Question:** why is the **cross-entropy loss** always **non-negative** and when it is **0**?

# Discriminative machine learning

Classification with the cross-entropy loss

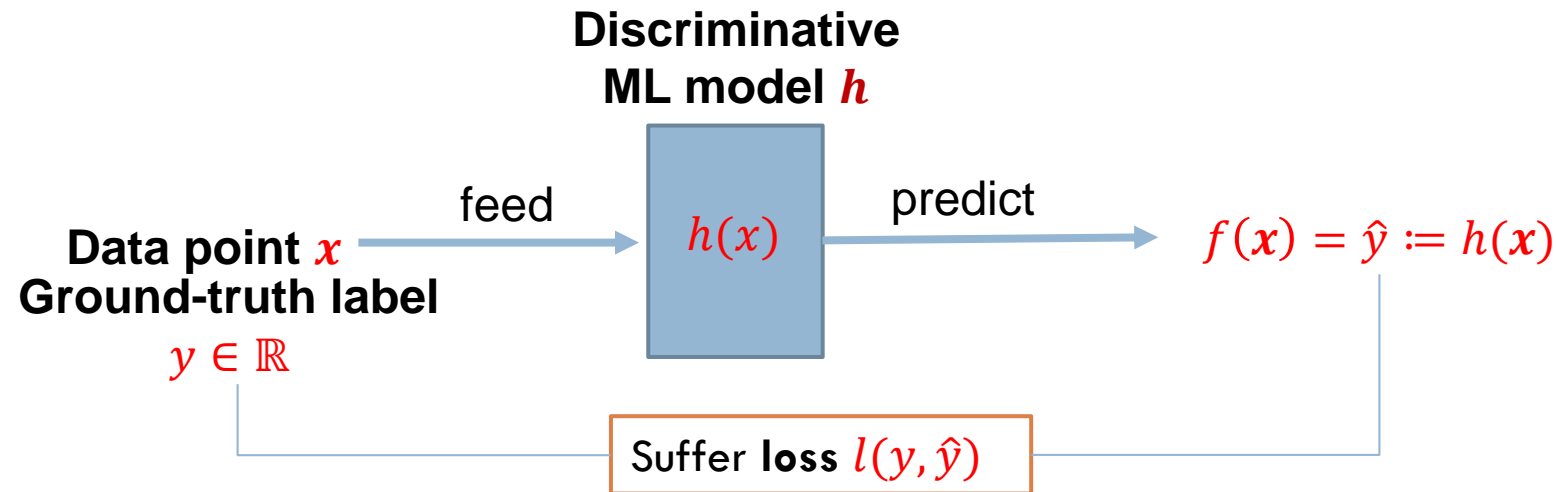


- We apply the **softmax function**:  $\mathbf{p}(x) = \text{softmax}(\mathbf{h}(x))$  to make  $p(x)$  **prediction probabilities**
  - $p_m(x) = p(y = m|x)$  is the **probability** to classify  $x$  to the class  $m$  for  $1 \leq m \leq M$
- The loss incurred for this prediction
  - $l(y, \hat{y}) = CE(1_y, p(x)) = -\log p_y(x)$  (**negative log likelihood**)
- The **loss for the training set**  $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$ 
  - $L(D, h) = \frac{1}{N} \sum_{i=1}^N CE(1_{y_i}, p(x_i)) = -\frac{1}{N} \sum_{i=1}^N \log p_{y_i}(x_i)$



# Discriminative machine learning

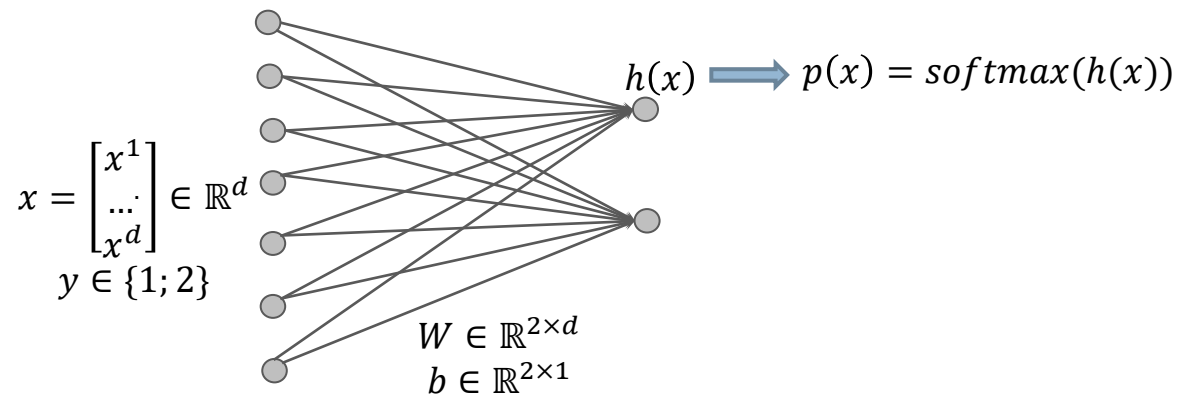
## Regression



- ❑ Only **one discriminative value**  $h(x) \in \mathbb{R}$ , we use this to predict  $\hat{y} = f(x) := h(x)$ .
- ❑ The **potential losses** could be
  - ❑ **L2 loss:**  $l(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$
  - ❑ **L1 loss:**  $l(y, \hat{y}) = |y - \hat{y}|$
  - ❑  **$\epsilon$ -insensitive loss:**  $l(y, \hat{y}) = \max\{0, |y - \hat{y}| - \epsilon\}$  where  $\epsilon \geq 0$ .

# Logistic regression (Forum discussion)

A simple feed-forward neural network



## □ Computational process

- $h(x) = Wx + b$ 
  - $h(x) \in \mathbb{R}^{2 \times 1}$  contains **discriminative scores** with respect to classes 1 and 2.
- $p(x) = \text{softmax}(h(x))$ 
  - $p(x)$  is the **prediction probabilities** with respect to classes 1 and 2.
- $\hat{y} = \begin{cases} 1, & p_1(x) \geq p_2(x) \\ 2, & p_1(x) < p_2(x) \end{cases}$
- $l(y, \hat{y}) = CE(1_y, p(x)) = -\log p_y(x) \rightarrow$  the **cross entropy loss**.

## □ Training process

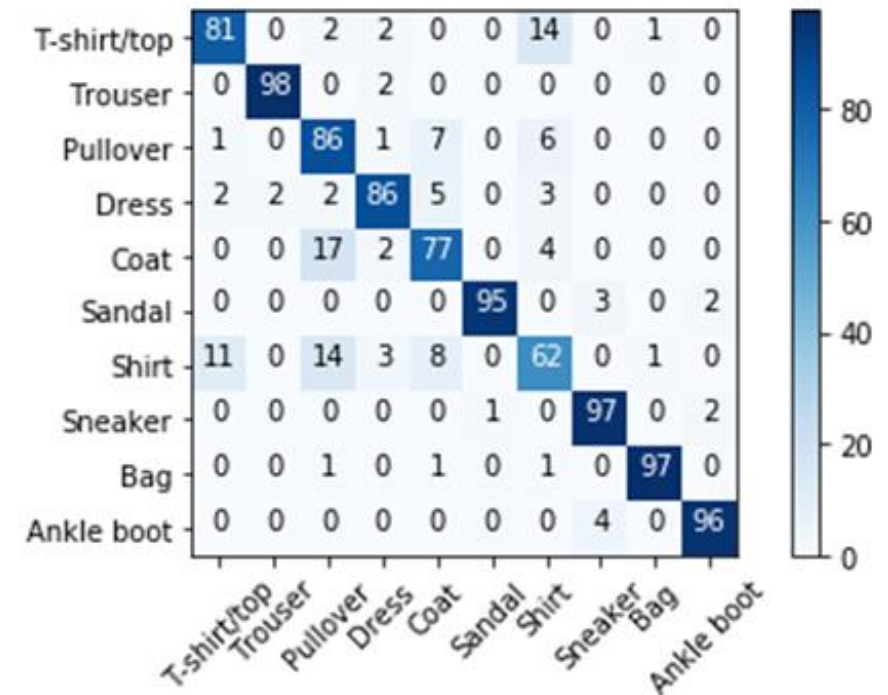
- **Training set**  $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$
- $L(D; W, b) = \frac{1}{N} \sum_{i=1}^N CE(1_{y_i}, p(x_i)) = -\frac{1}{N} \sum_{i=1}^N \log p_{y_i}(x_i)$
- $(W^*, b^*) = \text{argmin}_{W, b} L(D; W, b)$



# Confusion matrix

- How **confusing** a machine learning classifier is?
  - On-diagonal entries mean **correct** predictions.
  - Off-diagonal entries mean **incorrect** predictions.

Predicted class



Actual class

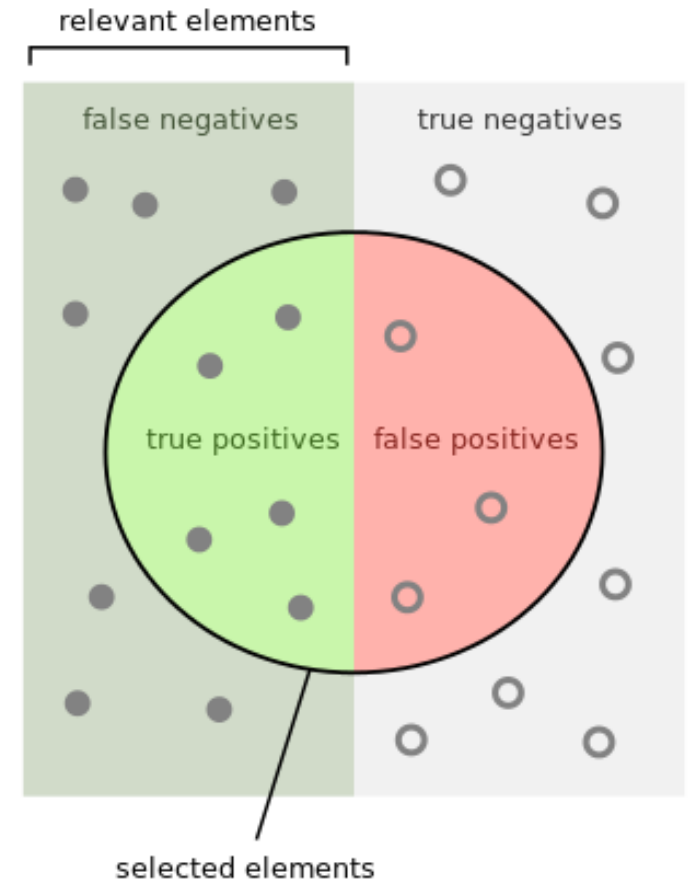
[Source: pythonhealthcare.org]

# Other metrics for measuring performance

## □ TP, FP, TN, FN and confusion matrix

		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

- We may think of **positive class** as “class 1” and **Negative class** as “class 0”.
- **The second letter** says what we predicted and **the first letter** says whether it was true or false.



# Other metrics for measuring performance

## □ Precision, recall, and F-score

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{F-score} = 2 \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Accuracy} = \frac{\text{\#corrects}}{\text{\#examples}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

# Example

## □ Example: Email Spam Detection

- In test set: 10 spam, 20 non-spam
- Positive = spam

### True Labels

Predicted as

		True Labels	
		SPAM (1)	NON-SPAM (0)
Predicted as	SPAM (1)	7	5
	NON-SPAM (0)	3	15

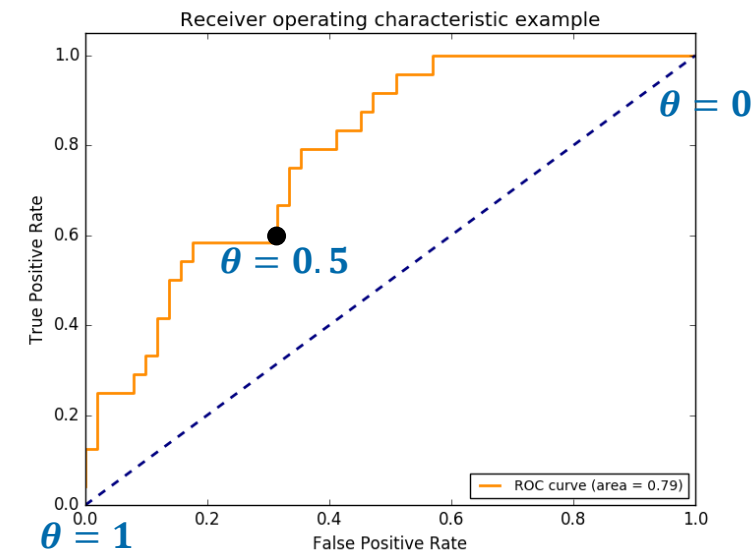
- $TP = 7, TN = 15$
- $FP = 5, FN = 3$
- $\text{Accuracy} = 22/30$
- $\text{Recall} = 7/10$
- $\text{Precision} = 7/12$
- $\text{F-score} = ??$

		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

# ROC curve (Forum discussion)

- For binary classification
  - **True positive rate (TPR) (Sensitivity)**
    - $TPR = TP / (TP + FN)$
  - **False positive rate (FPR)**
    - $FPR = FP / (FP + TN)$
- When output also includes probability (e.g., logistic regression), we can compute **ROC** by varying the detection threshold.
  - **ROC curve = true-positive rate (TPR) versus false-positive rate (FPR)**
  - Consider a **threshold**  $\theta \in [0; 1]$ . Given a data point  $x$ , the ML model returns  $p(x) = \mathbb{P}(y = 1|x)$  (i.e., the **probability to predict**  $x$  as the label 1). We predict  $x$  as the **label 1** if  $p(x) = \mathbb{P}(y = 1|x) \geq \theta$  and the **label -1** otherwise.
  - We compute **true positive rate (TPR)** and **false positive rate (FPR)** for the entire training set and plot **the point** [TPR, FPR].
  - We vary  $\theta \in [0; 1]$  to gain the ROC curve.

		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)



[Source: Wikipedia]



# Summary

## ❖ Linear algebra :

- ❖ Vectors, matrices and operations

## ❖ Calculus

- ❖ Derivative and chain rule

## ❖ Information theory

- ❖ Discrete distribution, KL divergence, entropy, and cross-entropy.

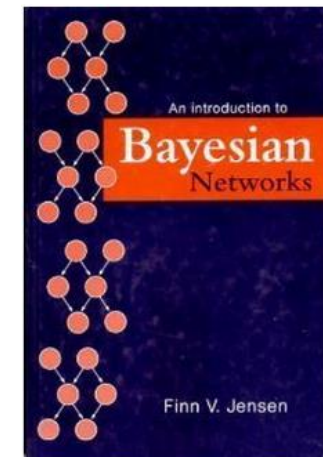
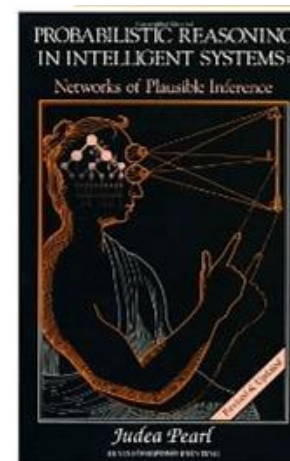
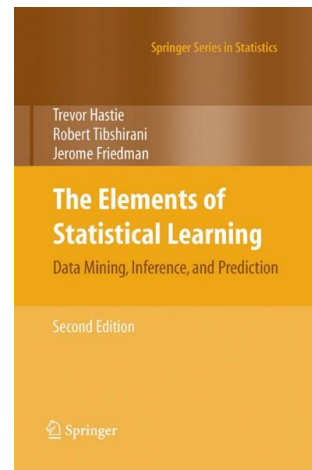
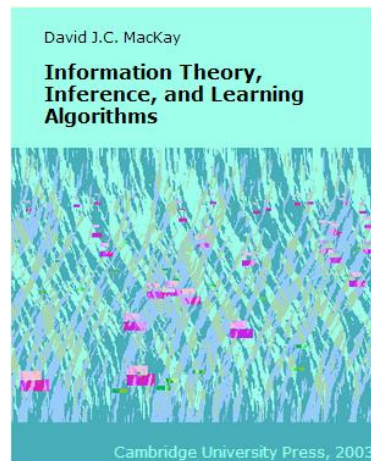
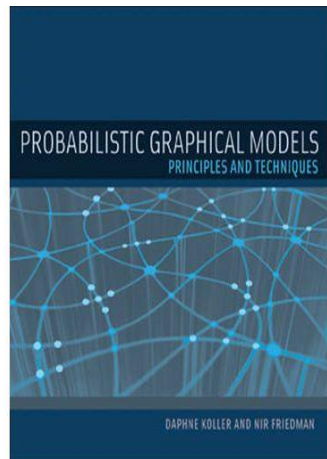
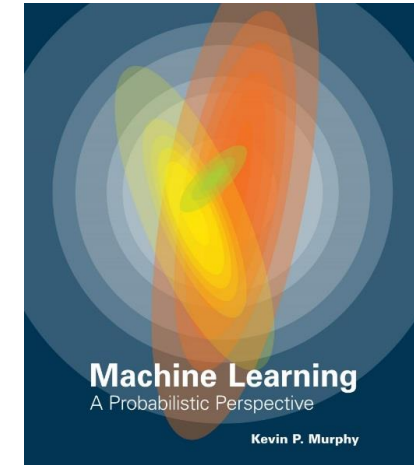
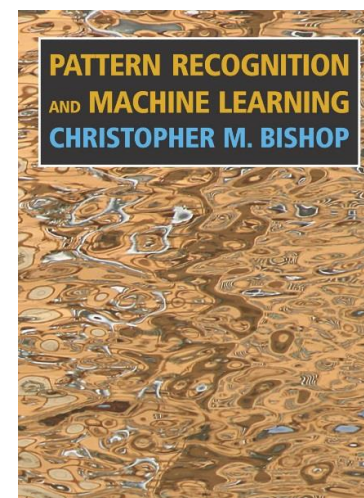
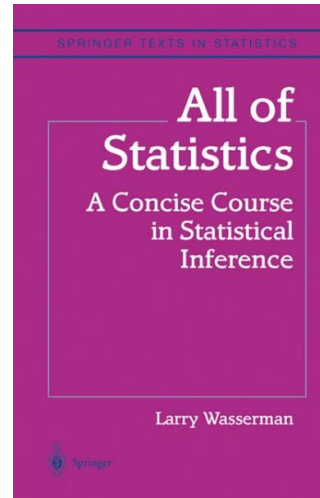
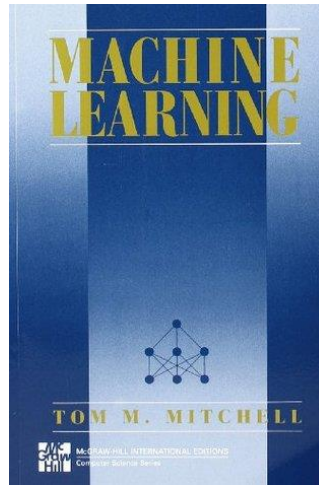
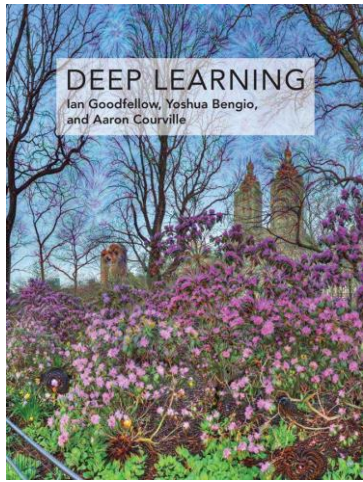
## ❖ Machine learning revisit

- ❖ Classification and regression
- ❖ Discriminative machine learning
- ❖ Confusion table and other metrics of interest (e.g., precision, recall, and F-score).

Thanks for your attention!

## Appendix

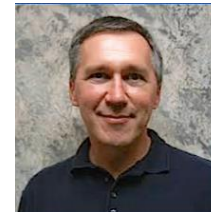
# Other ML Books



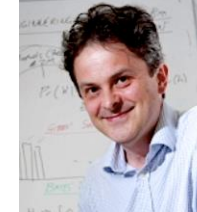
# Machine learning: five tribes

## The Symbolist School

Tribes	Origins	Master Algorithm
Symbolists	Logic, philosophy	<i>Inverse deduction</i>



Tom Mitchell



Steve Muggleton



Ross Quinlan

Deduction	Induction
Socrates is human + Humans are mortal <hr/> = ?	Socrates is human + ? <hr/> = Socrates is mortal

Decision trees, C4.5, first-order propositional logic, rule-based reasoning systems, etc.

# Machine learning: five tribes

Tribes	Origins	Master Algorithm
Symbolists	Logic, philosophy	<i>Inverse deduction</i>
Evolutionary	Evolutionary biology	<i>Genetic programming</i>

## The Evolutionary School



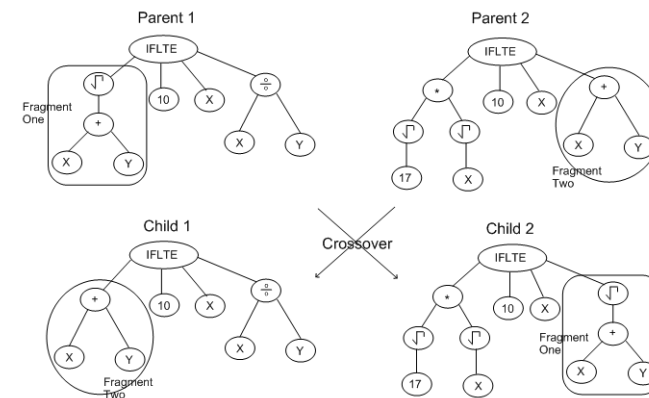
John Koza



John Holland



Hod Lipson



Genetic programming, genetic algorithms, fuzzy systems

# Machine learning: five tribes

## The Connectionist

Tribes	Origins	Master Algorithm
Symbolists	Logic, philosophy	<i>Inverse deduction</i>
Evolutionary	Evolutionary biology	<i>Genetic programming</i>
Connectionists	Neuroscience	<i>Backpropagation</i>



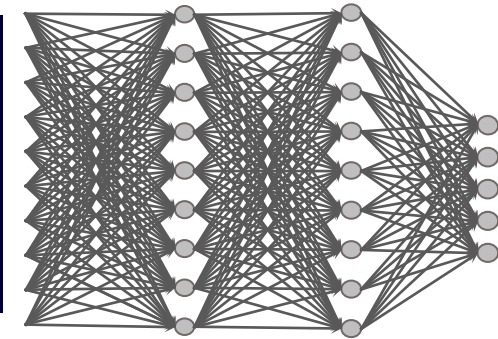
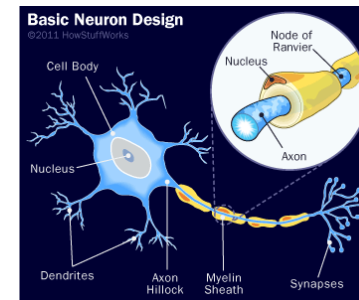
Geoff Hinton



Yann LeCun



Yoshua Bengio



Deep Neural Networks, Convolutional Neural Networks, Deep learning, etc.



# Machine learning: five tribes

Tribes	Origins	Master Algorithm
Symbolists	Logic, philosophy	<i>Inverse deduction</i>
Evolutionary	Evolutionary biology	<i>Genetic programming</i>
Connectionists	Neuroscience	<i>Backpropagation</i>
Bayesians	Statistics	<i>Bayes' theorem</i>

## The Bayesians



David Heckerman



Judea Pearl



Michael Jordan

**Likelihood**

How probable is the evidence  
given that our hypothesis is true?

**Prior**

How probable was our hypothesis  
before observing the evidence?

$$P(H | e) = \frac{P(e | H) P(H)}{P(e)}$$

**Posterior**

How probable is our hypothesis  
given the observed evidence?  
(Not directly computable)

**Marginal**

How probable is the new evidence  
under all possible hypotheses?  
 $P(e) = \sum P(e | H_i) P(H_i)$

Bayesian networks classifiers, graphical models, Hidden Markov models, Conditional Random Fields, most of statistical machine learning approaches ...



# Machine learning: five tribes

Tribes	Origins	Master Algorithm
Symbolists	Logic, philosophy	<i>Inverse deduction</i>
Evolutionary	Evolutionary biology	<i>Genetic programming</i>
Connectionists	Neuroscience	<i>Backpropagation</i>
Bayesian	Statistics	<i>Bayes' theorem</i>
Analogizers	Psychology	<i>Support Vector Machines</i>

## The Analogizers



Vladimir Vapnik



Peter Hart



Douglas Hofstadter

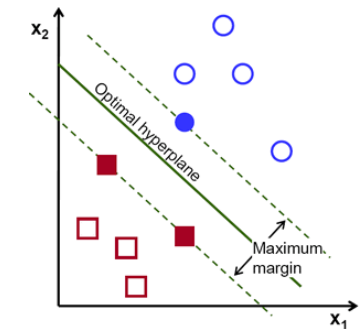
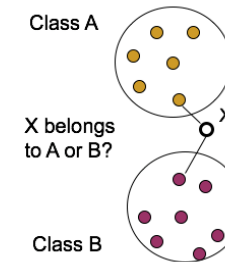
### ■ Instance-based classification

- Using most similar individual instances known in the past to classify a new instance

### ■ Typical approaches

- **k-nearest neighbor approach**

- Instances represented as points in a Euclidean space

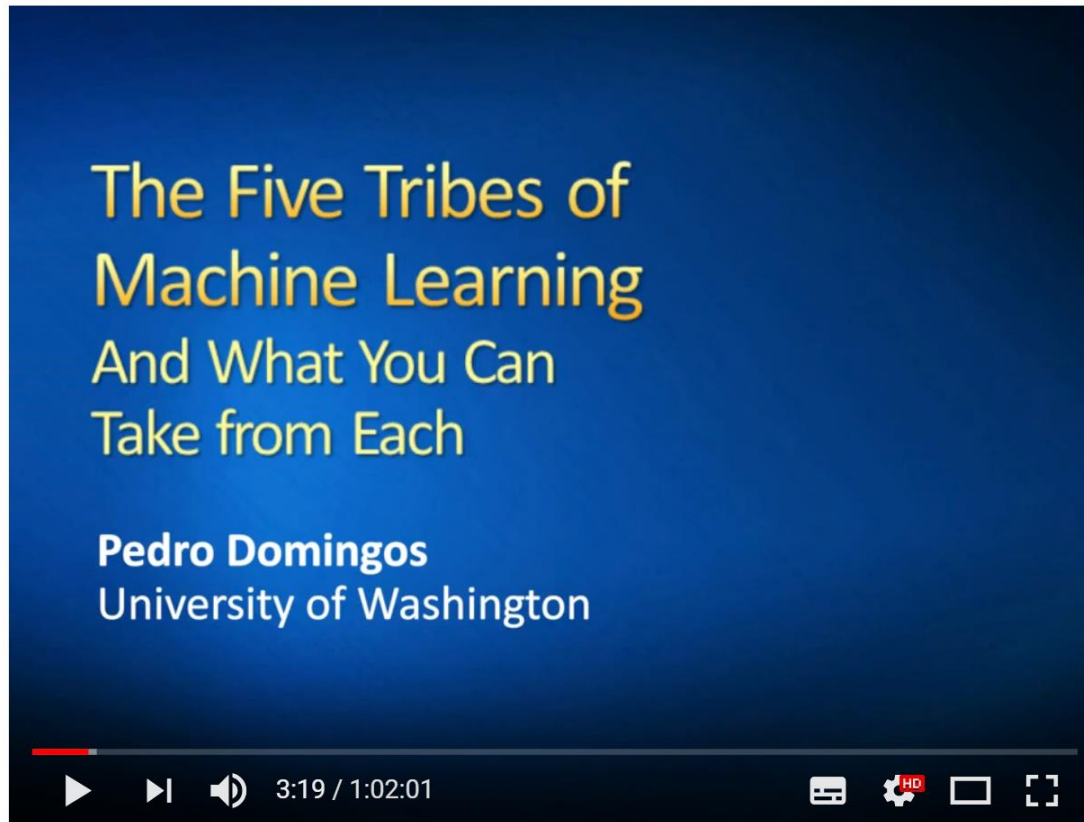


K-NN classifiers, Support vector machines, kernel machines, max-margin machines, instance-based classification, etc.

# Machine learning: five tribes

Approaches	Problem Solved	Solution Origin	Solution
Symbolists	Knowledge acquisition	Logic, philosophy	<i>Inverse deduction</i>
Evolutionary	Structure discovery (nature)	Evolutionary biology	<i>Genetic programming</i>
Connectionists	Credit assignment (nurture)	Neuroscience	<i>Backpropagation and deep learning</i>
Bayesian	Uncertainty	Statistics	<i>Probabilistic Inference</i>
Analogizers	Similarity	Psychology	<i>Kernel machines</i>

# Machine learning: five tribes



<https://www.youtube.com/watch?v=E8rOVwKQ5-8>