



# Introduction

Amir H. Payberah  
payberah@kth.se  
2020-10-27





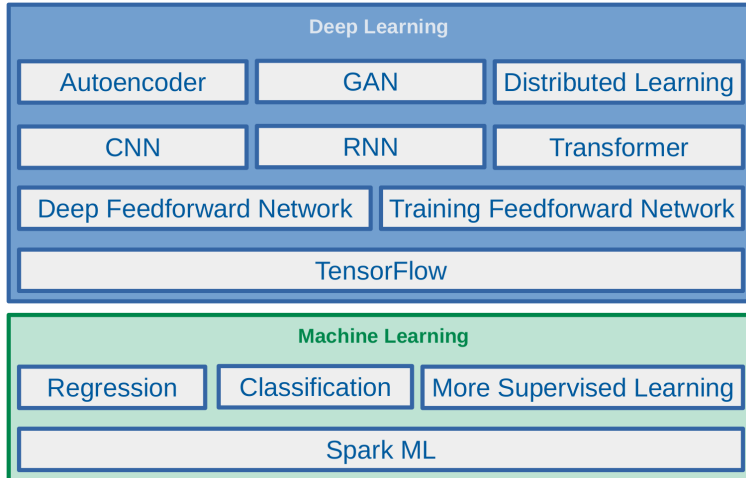
# Course Information



## Course Objective

- ▶ This course has a **system-based** focus.
- ▶ Learn the **theory** of **machine learning and deep learning**.
- ▶ Learn the **practical aspects** of building **machine learning and deep learning** algorithms using data **parallel programming platforms**, such as **Spark and TensorFlow**.

# Topics of Study

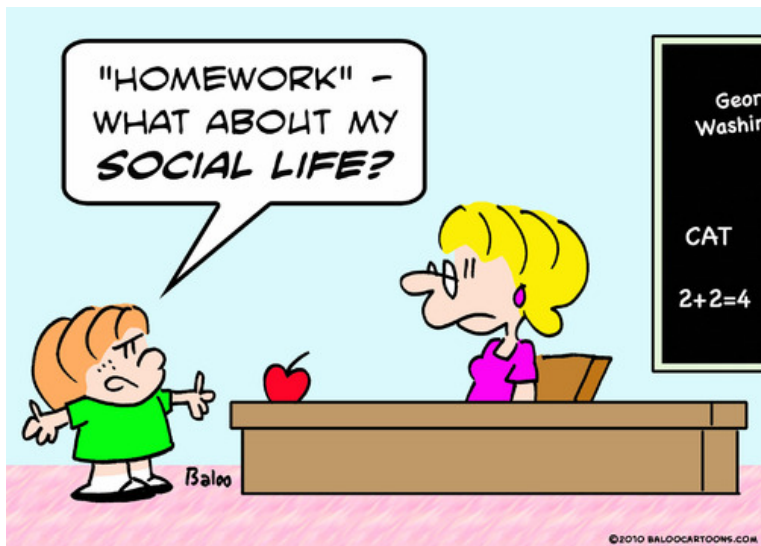






## Intended Learning Outcomes (ILOs)

- ▶ **ILO1:** explain the **principles** of **ML/DL algorithms** and apply their techniques to solve problems.
- ▶ **ILO2:** explain different **DNN architectures**, such as CNN, RNN, etc., and know how to build and train such networks.
- ▶ **ILO3:** explain the **principles** of **distributed learning**.
- ▶ **ILO4:** implement **ML/DL algorithms** using **Spark** and **TensorFlow**.





# The Course Assessment

- ▶ **Task1**: the **review** questions (P/F)
- ▶ **Task2**: the **lab** assignments (A-F)
- ▶ **Task3**: the final **project** (A-F)



## How Each ILO is Assessed?

	Task1	Task2	Task3
<b>ILO1</b>	x		
<b>ILO2</b>	x		
<b>ILO3</b>	x		
<b>ILO4</b>	x	x	x



## Task1: The Review Questions (A-F)

- ▶ One review question **per week**.
- ▶ Questions about the **lectures**.
- ▶ The review questions are **graded (A-F)**.



## Task2: The Lab Assignments (A-C-E)

- ▶ Two lab assignments: **source code** and **oral presentation**.
- ▶ **E**: source code
- ▶ **C**: source code + basic questions
- ▶ **A**: source code + advanced questions



## Task3: The Final Project (A-D)

- ▶ One final project: **source code** and **oral presentation**.
- ▶ **Proposed by students** and confirmed by the teacher: **A-level** or **C-level** proposals.
- ▶ **D**: source code C-level proposal
- ▶ **C**: source code C-level proposal + questions
- ▶ **B**: source code A-level proposal
- ▶ **A**: source code A-level proposal + questions



# The Final Grade

- ▶ The **final grade** is the **average** of the **two labs**, the **project**, and the **review questions**.
- ▶ To compute it, map **A-F** to **5-1**, and take the average.
- ▶ The floating values are **rounded up**, if they are **more than half**, otherwise they are **rounded down**.
  - E.g., **3.6** will be rounded to **4**, and **4.2** will be rounded to **4**.
- ▶ The half grades will be **rounded up**, if you submit the assignments **before their deadlines**, otherwise they will be **rounded down**.



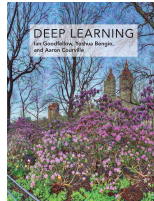
# How to Submit the Assignments?

- ▶ Through the [Canvas](#) site.
- ▶ Students will work in **groups of two** on all the [Tasks 1-4](#).



# The Course Material

- ▶ **Hands-on machine learning with Scikit-Learn and TensorFlow, 2nd Edition**, A. Geron, O'Reilly Media, 2019
- ▶ **Deep learning**, I. Goodfellow et al., Cambridge: MIT press, 2016
- ▶ **Spark - The Definitive Guide**, M. Zaharia et al., O'Reilly Media, 2018.





## The Course Web Page

`https://id2223kth.github.io`



## The Questions-Answers Page

<https://tinyurl.com/y6kcpmzy>



# The Course Overview

# Sheepdog or Mop



# Chihuahua or Muffin



# Barn Owl or Apple





# Raw Chicken or Donald Trump





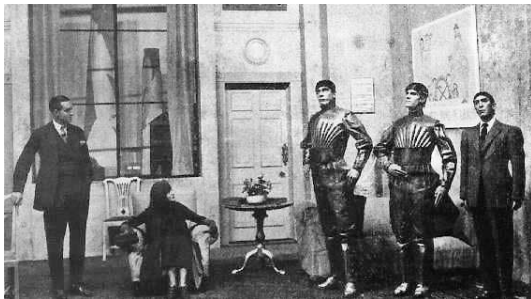
# Artificial Intelligence Challenge

- ▶ **Artificial intelligence (AI)** can solve problems that can be described by a **list of formal mathematical rules**.
- ▶ The **challenge** is to solve the tasks that are **hard for people to describe formally**.
- ▶ Let computers to **learn** from **experience**.

# History of AI

## 1920: Rossum's Universal Robots (R.U.R.)

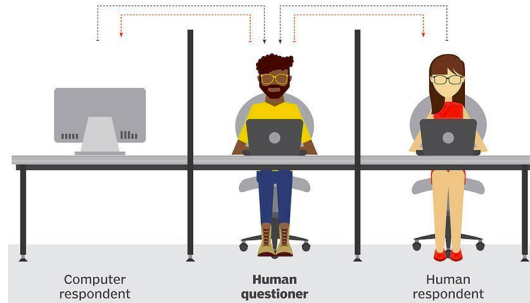
- ▶ A science fiction play by Karel Čapek, in 1920.
- ▶ A factory that creates artificial people named robots.



[<https://dev.to/lshultebraucks/a-short-history-of-artificial-intelligence-7hm>]

# 1950: Turing Test

- ▶ In 1950, **Turing** introduced the **Turing test**.
- ▶ An attempt to define **machine intelligence**.



[<https://searchenterpriseai.techtarget.com/definition/Turing-test>]

## 1956: The Dartmouth Workshop

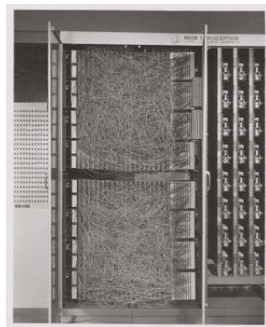
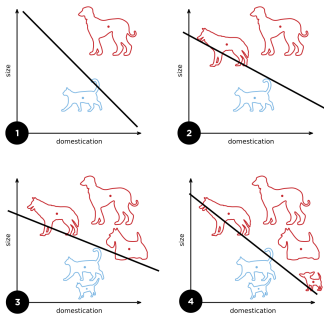
- ▶ Probably the first workshop of AI.
- ▶ Researchers from CMU, MIT, IBM met together and founded the AI research.



[<https://twitter.com/lordsaicom/status/898139880441696257>]

# 1958: Perceptron

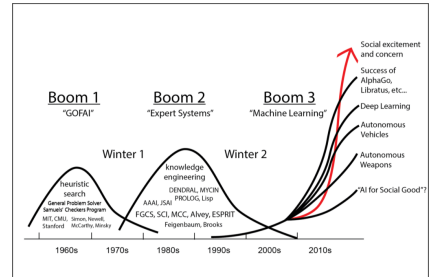
- ▶ A supervised learning algorithm for binary classifiers.
- ▶ Implemented in custom-built hardware as the **Mark 1 perceptron**.



[<https://en.wikipedia.org/wiki/Perceptron>]

# 1974–1980: The First AI Winter

- ▶ The over **optimistic settings**, which were not occurred
- ▶ The **problems**:
  - Limited **computer power**
  - Lack of **data**
  - Intractability and the **combinatorial explosion**

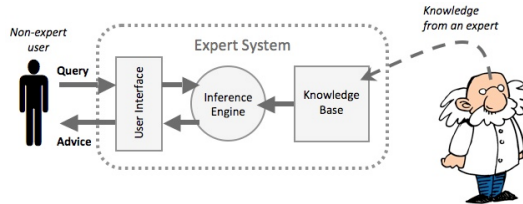


[<http://www.technologystories.org/ai-evolution>]



# 1980's: Expert systems

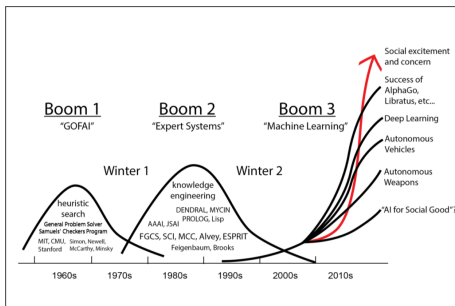
- ▶ The programs that solve problems in a **specific domain**.
- ▶ **Two** engines:
  - **Knowledge engine**: **represents** the **facts and rules** about a specific topic.
  - **Inference engine**: **applies** the **facts and rules** from the knowledge engine to new facts.



[[https://www.igcseict.info/theory/7\\_2/expert](https://www.igcseict.info/theory/7_2/expert)]

# 1987–1993: The Second AI Winter

- ▶ After a series of **financial setbacks**.
- ▶ The fall of **expert systems** and **hardware companies**.



[<http://www.technologystories.org/ai-evolution>]

## 1997: IBM Deep Blue

- ▶ The first chess computer to beat a world chess champion Garry Kasparov.



[<http://marksist.org/icerik/Tarihte-Bugun/1757/11-Mayis-1997-Deep-Blue-adli-bilgisayar>]



## 2012: AlexNet - Image Recognition

- ▶ The ImageNet competition in image classification.
- ▶ The AlexNet Convolutional Neural Network (CNN) won the challenge by a large margin.

IMGENET

## 2016: DeepMind AlphaGo

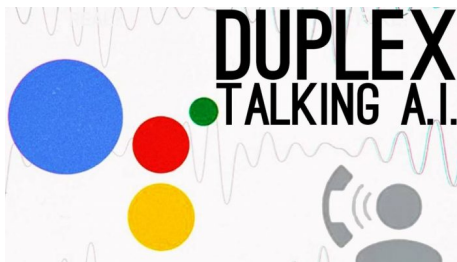
- ▶ DeepMind **AlphaGo** won **Lee Sedol**, one of the best players at Go.
- ▶ In 2017, DeepMind published **AlphaGo Zero**.
  - The **next generation** of AlphaGo.
  - It learned Go by playing **against itself**.



[<https://www.zdnet.com/article/google-alphago-caps-victory-by-winning-final-historic-go-match>]

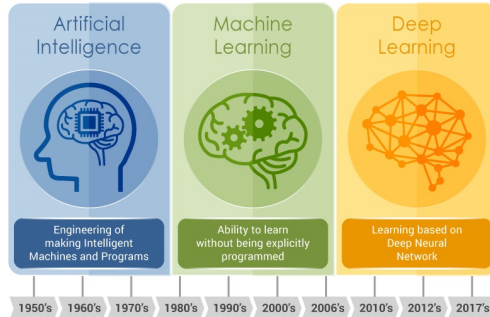
## 2018: Google Duplex

- ▶ An AI system for accomplishing **real-world tasks** over the phone.
- ▶ A **Recurrent Neural Network (RNN)** built using **TensorFlow**.



# AI Generations

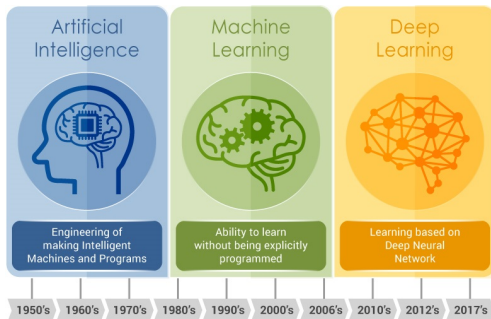
- ▶ Rule-based AI
- ▶ Machine learning
- ▶ Deep learning



[<https://bit.ly/2woLEzs>]

# AI Generations - Rule-based AI

- ▶ **Hard-code** knowledge
- ▶ Computers reason using **logical inference rules**

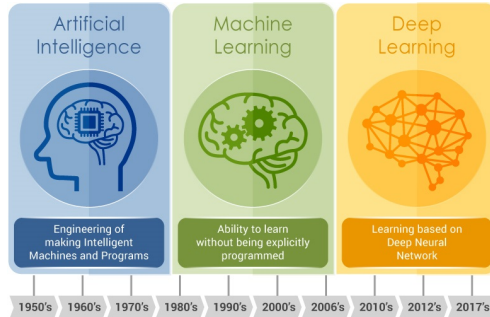


[<https://bit.ly/2woLEzs>]



# AI Generations - Machine Learning

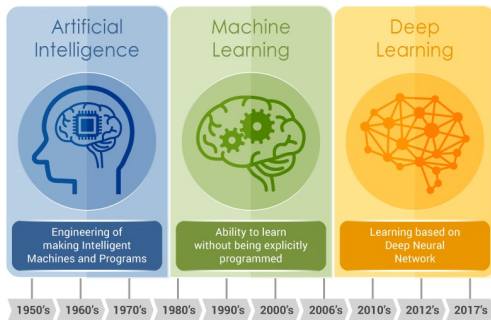
- ▶ If AI systems acquire **their own knowledge**
- ▶ **Learn from data** without being explicitly programmed



[<https://bit.ly/2woLEzs>]

# AI Generations - Deep Learning

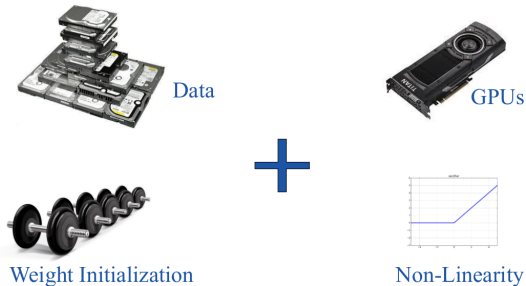
- ▶ For many tasks, it is difficult to know what features should be extracted
- ▶ Use machine learning to discover the mapping from representation to output



[<https://bit.ly/2woLEzs>]

# Why Does Deep Learning Work Now?

- ▶ Huge quantity of data
- ▶ Tremendous increase in computing power
- ▶ Better training algorithms





# Machine Learning and Deep Learning

# Learning Algorithms

- ▶ A **ML algorithm** is an algorithm that is able to **learn from data**.
- ▶ 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 **E**. (Tom M. Mitchell)



# Learning Algorithms - Example 1

- ▶ A **spam filter** that can learn to flag **spam** given examples of **spam emails** and examples of **regular emails**.
- ▶ **Task T**: flag spam for new emails
- ▶ **Experience E**: the training data
- ▶ **Performance measure P**: the ratio of correctly classified emails



[<https://bit.ly/2oip1YM>]

## Learning Algorithms - Example 2

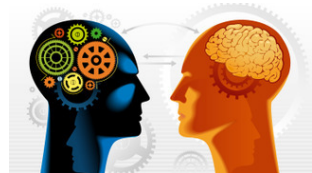
- ▶ Given dataset of prices of 500 houses, how can we learn to **predict the prices** of other houses, as a **function of the size of their living areas**?
- ▶ **Task T**: predict the price
- ▶ **Experience E**: the dataset of living areas and prices
- ▶ **Performance measure P**: the difference between the predicted price and the real price



[<https://bit.ly/2MyiJUy>]

# Types of Machine Learning Algorithms

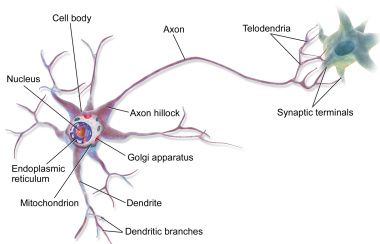
- ▶ **Supervised** learning
  - Input data is **labeled**, e.g., spam/not-spam or a stock price at a time.
  - **Regression vs. classification**
  
- ▶ **Unsupervised** learning
  - Input data is **unlabeled**.
  - Find **hidden structures** in data.





# From Machine Learning to Deep Learning

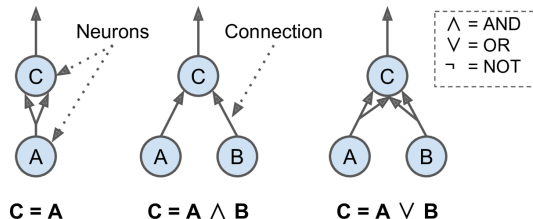
- ▶ **Deep Learning (DL)** is part of **ML** methods based on learning **data representations**.
- ▶ Mimic the **neural networks of our brain**.



[A. Geron, O'Reilly Media, 2017]

# Artificial Neural Networks

- ▶ **Artificial Neural Network (ANN)** is inspired by **biological neurons**.
- ▶ **One or more binary inputs** and **one binary output**
- ▶ **Activates its output** when more than a **certain number of its inputs** are active.



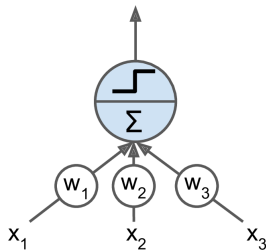
[A. Geron, O'Reilly Media, 2017]

# The Linear Threshold Unit (LTU)

- ▶ Inputs of a LTU are **numbers** (not binary).
- ▶ Each **input connection** is associated with a **weight**.
- ▶ Computes a **weighted sum of its inputs** and applies a **step function** to that **sum**.

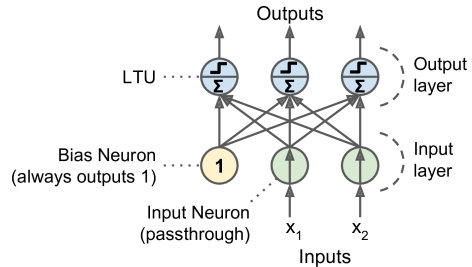
- ▶  $z = w_1x_1 + w_2x_2 + \dots + w_nx_n = \mathbf{w}^T\mathbf{x}$

- ▶  $\hat{y} = \text{step}(z) = \text{step}(\mathbf{w}^T\mathbf{x})$



# The Perceptron

- ▶ The **perceptron** is a **single layer** of LTUs.
- ▶ The **input neurons** output whatever **input they are fed**.
- ▶ A **bias neuron**, which just **outputs 1 all the time**.



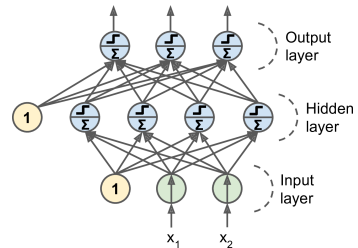


# Deep Learning Models

- ▶ Deep Neural Network (**DNN**)
- ▶ Convolutional Neural Network (**CNN**)
- ▶ Recurrent Neural Network (**RNN**)
- ▶ Autoencoders
- ▶ Generative Adversarial Network (**GAN**)

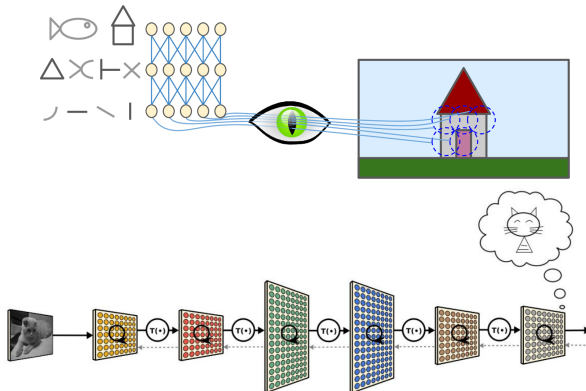
# Deep Neural Networks

- ▶ Multi-Layer Perceptron (MLP)
  - One **input layer**.
  - One or more layers of **LTUs** (hidden layers).
  - One **final layer of LTUs** (output layer).
- ▶ Deep Neural Network (DNN) is an ANN with **two or more hidden layers**.
- ▶ **Backpropagation** training algorithm.



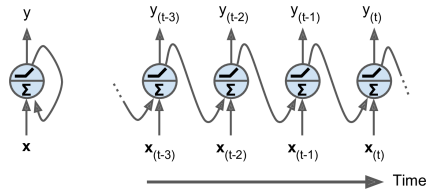
# Convolutional Neural Networks

- ▶ Many neurons in the **visual cortex** react only to a **limited region** of the visual field.
- ▶ The **higher-level** neurons are based on the outputs of **neighboring lower-level** neurons.



# Recurrent Neural Networks

- ▶ The **output** depends on the **input** and the **previous computations**.



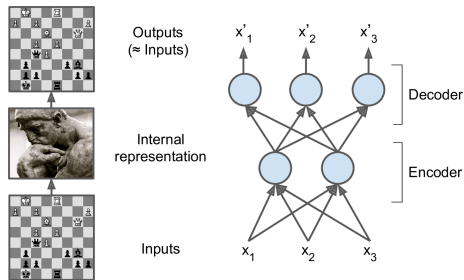
- ▶ Analyze **time series data**, e.g., stock market, and autonomous driving systems.
- ▶ Work on sequences of **arbitrary lengths**, rather than on **fixed-sized inputs**.





# Autoencoders and Generative Models

- ▶ Learn **efficient representations** of the input data, **without any supervision**.
  - With a **lower dimensionality** than the input data.
- ▶ **Generative model**: generate **new data** that looks very similar to the training data.
- ▶ Preserve **as much information as possible**.



[A. Geron, O'Reilly Media, 2017]



# Linear Algebra Review



# Vector

- ▶ A **vector** is an array of numbers.
- ▶ Notation:
  - Denoted by **bold lowercase letters**, e.g.,  $\mathbf{x}$ .
  - $x_i$  denotes the  $i$ th entry.

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



# Matrix and Tensor

- ▶ A **matrix** is a 2-D array of numbers.
- ▶ A **tensor** is an array with more than two axes.
- ▶ **Notation:**
  - Denoted by **bold uppercase letters**, e.g., **A**.
  - $a_{ij}$  denotes the entry in  $i$ th row and  $j$ th column.
  - If **A** is  $m \times n$ , it has  $m$  rows and  $n$  columns.

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \dots & a_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & a_{m,3} & \dots & a_{m,n} \end{bmatrix}$$



# Matrix Addition and Subtraction

- ▶ The **matrices** must have the same dimensions.

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a + e & b + f \\ c + g & d + h \end{bmatrix}$$

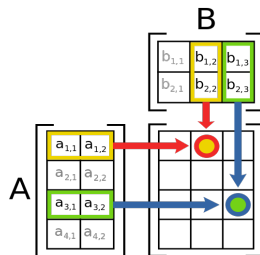
# Matrix Product

- ▶ The **matrix product** of matrices **A** and **B** is a third matrix **C**, where **C = AB**.
- ▶ If **A** is of shape  $m \times n$  and **B** is of shape  $n \times p$ , then **C** is of shape  $m \times p$ .

$$c_{ij} = \sum_k a_{ik} b_{kj}$$

## ▶ Properties

- Associative: **(AB)C = A(BC)**
- Not commutative: **AB ≠ BA**



[[https://en.wikipedia.org/wiki/Matrix\\_multiplication](https://en.wikipedia.org/wiki/Matrix_multiplication)]



# Matrix Transpose

- ▶ Swap the **rows and columns** of a matrix.

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} \Rightarrow \mathbf{A}^T = \begin{bmatrix} a & c & e \\ b & d & f \end{bmatrix}$$

- ▶ Properties

- $\mathbf{A}_{ij} = \mathbf{A}_{ji}^T$
- If  $\mathbf{A}$  is  $m \times n$ , then  $\mathbf{A}^T$  is  $n \times m$
- $(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$
- $(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T$



## Inverse of a Matrix

- ▶ If  $\mathbf{A}$  is a **square** matrix, its **inverse** is called  $\mathbf{A}^{-1}$ .

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$$

- ▶ Where  $\mathbf{I}$ , the **identity** matrix, is a **diagonal matrix** with all **1's on the diagonal**.

$$\mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbf{I}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$





## $L^p$ Norm for Vectors

- ▶ We can measure the **size of vectors** using a **norm** function.
- ▶ Norms are functions **mapping vectors to non-negative values**.

- ▶  $L^1$  norm

$$\|\mathbf{x}\|_1 = \sum_i |x_i|$$

- ▶  $L^2$  norm

$$\|\mathbf{x}\|_2 = \left( \sum_i |x_i|^2 \right)^{\frac{1}{2}} = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

- ▶  $L^p$  norm

$$\|\mathbf{x}\|_p = \left( \sum_i |x_i|^p \right)^{\frac{1}{p}}$$



# Probability Review



# Random Variables

- ▶ **Random variable:** a **variable** that can take on **different values** randomly.
- ▶ Random variables may be **discrete** or **continuous**.
  - **Discrete** random variable: **finite or countably infinite** number of states
  - **Continuous** random variable: **real value**
- ▶ **Notation:**
  - Denoted by an **upper case letter**, e.g.,  $X$
  - Values of a random variable  $X$  are denoted by **lower case letters**, e.g.,  $x$  and  $y$ .



# Probability Distributions

- ▶ **Probability distribution**: how likely a random variable is to take on each of its possible states.
  - E.g., the random variable  $X$  denotes the outcome of a coin toss.
  - The probability distribution of  $X$  would take the value 0.5 for  $X = \text{head}$ , and 0.5 for  $Y = \text{tail}$  (assuming the coin is fair).
- ▶ The way we describe probability distributions depends on whether the variables are discrete or continuous.



# Discrete Variables

- ▶ **Probability mass function (PMF)**: the probability distribution of a discrete random variable  $X$ .
- ▶ **Notation**: denoted by a lowercase  $p$ .
  - E.g.,  $p(x) = 1$  indicates that  $X = x$  is certain
  - E.g.,  $p(x) = 0$  indicates that  $X = x$  is impossible
- ▶ **Properties**:
  - The domain  $D$  of  $p$  must be the set of all possible states of  $X$
  - $\forall x \in D(X), 0 \leq p(x) \leq 1$
  - $\sum_{x \in D(X)} p(x) = 1$



# Independence

- ▶ Two random variables  $X$  and  $Y$  are **independent**, if their **probability distribution** can be expressed as their **products**.

$$\forall x \in D(X), y \in D(Y), p(X = x, Y = y) = p(X = x)p(Y = y)$$

- ▶ E.g., if a **coin is tossed** and a single **6-sided die is rolled**, then the probability of landing on the **head** side of the coin and **rolling a 3** on the die is:

$$p(X = \text{head}, Y = 3) = p(X = \text{head})p(Y = 3) = \frac{1}{2} \times \frac{1}{6} = \frac{1}{12}$$



# Conditional Probability

- ▶ **Conditional probability:** the probability of an event given that another event has occurred.

$$p(Y = y \mid X = x) = \frac{p(Y = y, X = x)}{p(X = x)}$$

- ▶ E.g., if 60% of the class passed both labs and 80% of the class passed the first labs, then what percent of those who passed the first lab also passed the second lab?
  - E.g.,  $X$  and  $Y$  random variables for the first and the second labs, respectively.

$$p(Y = \text{lab2} \mid X = \text{lab1}) = \frac{p(Y = \text{lab2}, X = \text{lab1})}{p(X = \text{lab1})} = \frac{0.6}{0.8} = \frac{3}{4}$$



## Expectation

- ▶ The **expected value** of a random variable  $X$  with respect to a probability distribution  $p(X)$  is the **average** value that  $X$  takes on when it is drawn from  $p(X)$ .

$$E_{x \sim p}[X] = \sum_x p(x)x$$

- ▶ E.g., If  $X : \{1, 2, 3\}$ , and  $p(X = 1) = 0.3$ ,  $p(X = 2) = 0.5$ ,  $p(X = 3) = 0.2$ 
  - $E[X] = 0.3 \times 1 + 0.5 \times 2 + 0.2 \times 3 = 1.9$



## Variance and Standard Deviation

- ▶ The **variance** gives a measure of how much the **values of a random variable  $X$**  vary as we sample it from its **probability distribution  $p(X)$** .

$$\text{Var}(X) = E[(X - E[X])^2]$$
$$\text{Var}(X) = \sum_x p(x)(x - E[X])^2$$

- ▶ E.g., If  $X : \{1, 2, 3\}$ , and  $p(X = 1) = 0.3$ ,  $p(X = 2) = 0.5$ ,  $p(X = 3) = 0.2$ 
  - $E[X] = 0.3 \times 1 + 0.5 \times 2 + 0.2 \times 3 = 1.9$
  - $\text{Var}(X) = 0.3(1 - 1.9)^2 + 0.5(2 - 1.9)^2 + 0.2(3 - 1.9)^2 = 0.49$
- ▶ The **standard deviation**, shown by  $\sigma$ , is the **square root of the variance**.



## Covariance (1/2)

- ▶ The **covariance** gives some sense of **how much two values are linearly related** to each other.

$$\begin{aligned}\text{Cov}(X, Y) &= E[(X - E[X])(Y - E[Y])] \\ \text{Cov}(X, Y) &= \sum_{(x,y)} p(x, y)(x - E[X])(y - E[Y])\end{aligned}$$

## Covariance (2/2)

			Y		
	p(X, Y)	1	2	3	p(X)
	1	1/4	1/4	0	1/2
X	2	0	1/4	1/4	1/2
	p(Y)	1/4	1/2	1/4	1

$$E[X] = \frac{1}{2} \times 1 + \frac{1}{2} \times 2 = \frac{3}{2} \qquad E[Y] = \frac{1}{4} \times 1 + \frac{1}{2} \times 2 + \frac{1}{4} \times 3 = 2$$

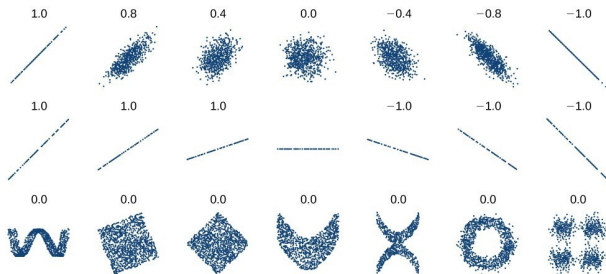
$$\text{Cov}(X, Y) = \sum_{(x,y)} p(x, y)(x - E[X])(y - E[Y])$$

$$= \frac{1}{4}(1 - \frac{3}{2})(1 - 2) + \frac{1}{4}(1 - \frac{3}{2})(2 - 2) + 0(1 - \frac{3}{2})(3 - 2) \\ + 0(2 - \frac{3}{2})(1 - 2) + \frac{1}{4}(2 - \frac{3}{2})(2 - 2) + \frac{1}{4}(2 - \frac{3}{2})(3 - 2) = \frac{1}{4}$$

# Correlation Coefficient

- ▶ The **Correlation coefficient** is a quantity that measures the **strength** of the **association** (or **dependence**) between two random variables, e.g., **X** and **Y**.

$$\rho(X, Y) = \frac{\text{Cov}(X, Y)}{\sigma(X)\sigma(Y)}$$





## Probability and Likelihood (1/2)

- ▶ Let  $X : \{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$  be a **discrete random variable** drawn **independently** from a **distribution probability  $p$**  depending on a **parameter  $\theta$** .
  - For six tosses of a coin,  $X : \{h, t, t, t, h, t\}$ , **h**: head, and **t**: tail.
  - Suppose you have a **coin** with probability  $\theta$  to land heads and  $(1 - \theta)$  to land tails.
- ▶  $p(X | \theta = \frac{2}{3})$  is the **probability** of  $X$  given  $\theta = \frac{2}{3}$ .
- ▶  $p(X = h | \theta)$  is the **likelihood** of  $\theta$  given  $X = h$ .
- ▶ **Likelihood (L)**: a function of the **parameters ( $\theta$ )** of a probability model, given **specific observed data**, e.g.,  $X = h$ .

$$L(\theta | X) = p(X | \theta)$$



## Probability and Likelihood (2/2)

- ▶ The **likelihood** differs from that of a **probability**.
- ▶ A **probability**  $p(X | \theta)$  refers to the occurrence of **future events**.
- ▶ A **likelihood**  $L(\theta | X)$  refers to **past events** with known outcomes.

# Maximum Likelihood Estimator

- ▶ If samples in  $\mathbf{X}$  are **independent** we have:

$$\begin{aligned} L(\theta | \mathbf{X}) &= p(\mathbf{X} | \theta) = p(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(m)} | \theta) \\ &= p(\mathbf{x}^{(1)} | \theta) p(\mathbf{x}^{(2)} | \theta) \cdots p(\mathbf{x}^{(m)} | \theta) = \prod_{i=1}^m p(\mathbf{x}^{(i)} | \theta) \end{aligned}$$

- ▶ The **maximum likelihood estimator (MLE)**: what is the **most likely value** of  $\theta$  given the training set?

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\theta} L(\theta | \mathbf{X}) = \arg \max_{\theta} \prod_{i=1}^m p(\mathbf{x}^{(i)} | \theta)$$

## Maximum Likelihood Estimator - Example

- ▶ Six tosses of a coin, with the following model:
  - Possible outcomes: **h** with probability of  $\theta$ , and **t** with probability  $(1 - \theta)$ .
  - Results of coin tosses are independent of one another.
- ▶ Data:  $X : \{\mathbf{h}, \mathbf{t}, \mathbf{t}, \mathbf{t}, \mathbf{h}, \mathbf{t}\}$

- ▶ The likelihood is

$$\begin{aligned}L(\theta | X) &= p(X | \theta) \\ &= p(X = \mathbf{h} | \theta)p(X = \mathbf{t} | \theta)p(X = \mathbf{t} | \theta)p(X = \mathbf{t} | \theta)p(X = \mathbf{h} | \theta)p(X = \mathbf{t} | \theta) \\ &= \theta(1 - \theta)(1 - \theta)(1 - \theta)\theta(1 - \theta) \\ &= \theta^2(1 - \theta)^4\end{aligned}$$

- ▶  $\hat{\theta}$  is the value of  $\theta$  that maximizes the likelihood:

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\theta} L(\theta | X) = \frac{2}{2 + 4}$$





# Log-Likelihood

- ▶ The MLE product is prone to numerical underflow.

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\theta} L(\theta | X) = \arg \max_{\theta} \prod_{i=1}^m p(x^{(i)} | \theta)$$

- ▶ To overcome this problem we can use the logarithm of the likelihood.
  - It does not change its arg max, but transforms a product into a sum.

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\theta} \sum_{i=1}^m \log p(x^{(i)} | \theta)$$



## Negative Log-Likelihood

- ▶ **Likelihood:**  $L(\theta | X) = \prod_{i=1}^m p(x^{(i)} | \theta)$
- ▶ **Log-Likelihood:**  $\log L(\theta | X) = \log \prod_{i=1}^m p(x^{(i)} | \theta) = \sum_{i=1}^m \log p(x^{(i)} | \theta)$
- ▶ **Negative Log-Likelihood:**  $-\log L(\theta | X) = -\sum_{i=1}^m \log p(x^{(i)} | \theta)$
- ▶ Negative log-likelihood is also called the **cross-entropy**



# Cross-Entropy

- ▶ **Cross-entropy**: quantify the **difference (error)** between **two probability distributions**.
- ▶ **How close** is the **predicted distribution** to the **true distribution**?

$$H(p, q) = - \sum_x p(x) \log(q(x))$$

- ▶ Where **p** is the **true distribution**, and **q** the **predicted distribution**.



## Cross-Entropy - Example

- ▶ Six tosses of a coin:  $X : \{h, t, t, t, h, t\}$
- ▶ The true distribution  $p$ :  $p(h) = \frac{2}{6}$  and  $p(t) = \frac{4}{6}$
- ▶ The predicted distribution  $q$ :  $h$  with probability of  $\theta$ , and  $t$  with probability  $(1 - \theta)$ .
- ▶ Cross entropy:  $H(p, q) = -\sum_x p(x)\log(q(x))$   
 $= -p(h)\log(q(h)) - p(t)\log(q(t)) = -\frac{2}{6}\log(\theta) - \frac{4}{6}\log(1 - \theta)$
- ▶ Likelihood:  $\theta^2(1 - \theta)^4$
- ▶ Negative log likelihood:  $-\log(\theta^2(1 - \theta)^4) = -2\log(\theta) - 4\log(1 - \theta)$

# Summary



# Summary

- ▶ Logic-based AI, Machine Learning, Deep Learning
- ▶ Deep Learning models
  - Deep Feed Forward
  - Convolutional Neural Network (CNN)
  - Recurrent Neural Network (RNN)
  - Autoencoders
- ▶ Linear algebra and probability
  - Random variables
  - Probability distribution
  - Likelihood
  - Negative log-likelihood and cross-entropy



## References

- ▶ Ian Goodfellow et al., Deep Learning (Ch. 1, 2, 3)

# Questions?

## Acknowledgements

Some of the pictures were copied from the book Hands-On Machine Learning with Scikit-Learn and TensorFlow, Aurelien Geron, O'Reilly Media, 2017.