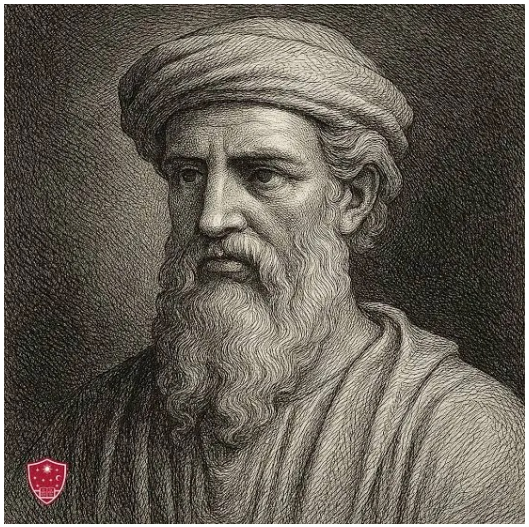
Review of Mathematics for Deep Learning

AI with Deep Learning
EE4016

Prof. PO Lai-Man
Department of Electrical Engineering
City University of Hong Kong

All Things Are Number (Pythagoras, 570 – c. 495 BC)

- This famous philosophical proposition is attributed to **Pythagoras**, the ancient Greek philosopher and mathematician.
- Pythagoras and his followers (the Pythagorean school) believed that **numbers are the fundamental essence of all things** in the universe.
- They held that **reality is ultimately mathematical**—harmony, form, and even moral principles can be understood through numerical relationships.
- This idea profoundly influenced later thinkers, including Plato, and laid early groundwork for the mathematization of nature in Western science.



Galilei, Galileo (1564 – 1642)

[The universe] cannot be read until we have learnt the language and become familiar with the characters in which it is written. **It is written in mathematical language**, and the letters are triangles, circles and other geometrical figures, without which means it is humanly impossible to comprehend a single word.

OpereIl Saggiatore p. 171.

Demis Hassabis (DeepMind CEO, 2025)

- Demis Hassabis operates on the foundational belief that "**everything is computationally tractable**," meaning the entire universe and all its phenomena can, in theory, be modeled and understood by a sufficiently powerful computer.

https://x.com/i/status/2000994920753144043

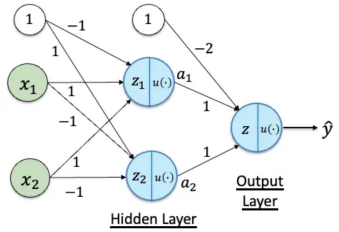
Four-Level of Understanding

1. Natural Language Understanding

- **Employ human language** to describe the concepts, problems and solutions in AI.
  - e.g. Deep learning uses multi-layered artificial neural networks to recognize complex patterns in data, resulting in state-of-the-art performance in domains such as CV and NLP.

2. Visual Understanding

- **Utilize figures** to visually represent concepts, problems and solutions in AI.



3. Mathematical Understanding

- **Utilize mathematical equations** to represent the concepts in AI.
  - e.g. A MLP model can be expressed as $P_\theta(\mathbf{x}) = \text{softmax}\left(\mathbf{W}^{(2)}\text{ReLU}\left(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\right) + \mathbf{b}^{(2)}\right)$

4. Implementation of AI System

- Implement deep learning research and applications **using Python programming language and frameworks like PyTorch.**

EE4016 Prerequisites

- **Prerequisites:**
  - Linear Algebra, Multi-variable Calculus and Probabilities, **and**
  - Object-Oriented Programme (e.g. Python)

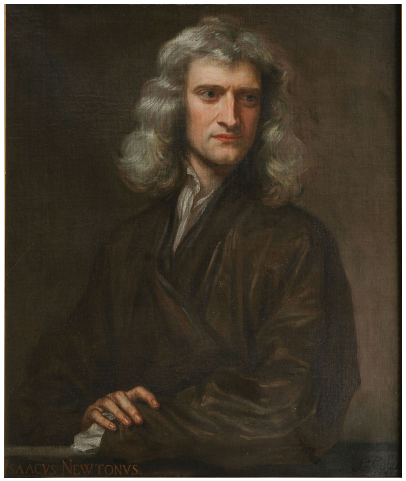
- Please be advised, **EE4016 is a course with a STRONG mathematical and programming components.**
  - **Focus**: Deep learning architectures trainable via gradient-based approaches.
    - Linear Algebra and Matrix Calculus.
    - Probability and Statistics.
  - **Skills gained through assignments and projects**:
    - Proficiency in Python and familiarity with deep learning libraries like PyTorch.

EE4016 Prerequisites

- **Linear Algebra**: Matrix Multiplication
  - Linear algebra is a fundamental concept that is essential for understanding neural network algorithms, which are often formulated as matrix computations.
- **Differential Calculus**: Derivative and Vector Calculus
  - Differential multivariable calculus plays a key role in optimization techniques, specifically differential programming using gradient descent.
- **Probability**: Conditional Probability, Random Variable, Expectation, Variance
  - Probability theory is another important area of math for deep learning as many applications involve dealing with probabilistic models.
- **Programming**: Python, Object-Oriented Programming, PyTorch

First Industrial Revolution (16th -17th Century)

- The First Industrial Revolution is linked to the First Scientific Revolution, which established modern science.
- **Newton's laws of motion provided a framework for understanding physical forces**, guiding engineering and technological advancements.
- **Newton's Second Law of Motion:**



Isaac Newton

$$F = ma = \frac{d}{dt}(m \cdot v) = m\frac{d^2x}{dt^2}$$

- This equation describes the relationship between force (F), mass (m), and acceleration (a). It is fundamental to classical mechanics.
- **Force** is equal to **rate of change of momentum** (mass · velocity)

Second Industrial Revolution (18th - 19th Century)

- The Second Scientific Revolution saw the development of **electromagnetism**.
- **Maxwell's Equations**:

Gauss's law for electricity

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0}$$

Gauss's law for magnetism

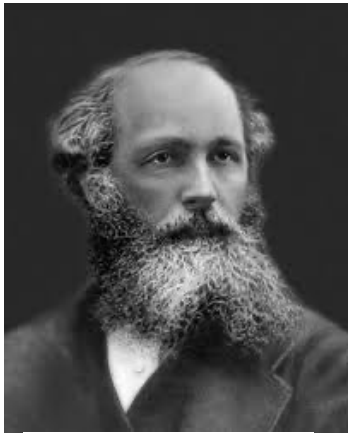
$$\nabla \cdot \mathbf{B} = 0$$

Faraday's law of induction

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

Ampère-Maxwell law

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t}$$



James C. Maxwell

- These four equations, formulated by James Clerk Maxwell, describe the behavior of electric ($\mathbf{E}$) and magnetic ($\mathbf{B}$) fields and their interactions with charges (ρ) and currents ($\mathbf{J}$).

Maxwell's Equations

- **Electric Fields and Charges ($\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0}$)**: Electric fields are generated by electric charges, as described by Gauss's Law for Electricity.
- **Magnetic Fields ($\nabla \cdot \mathbf{B} = 0$)** : Magnetic fields are generated by moving charges (currents) and changing electric fields, as described by Ampère's Law with Maxwell's addition.
- **Induced Electric Fields ($\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$)**: Changing magnetic fields induce electric fields, as described by Faraday's Law of Induction.
- **No Magnetic Monopoles ($\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t}$)**: Magnetic fields are continuous and have no starting or ending points, as described by Gauss's Law for Magnetism.

Third Industrial Revolution (20th Century)

- The Third Industrial Revolution, or Digital Revolution, started in the mid-20th century, marked by digital technology, computing, and automation.
- **Shannon's Information Entropy (Information Theory):**

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i)$$

Claude Shannon

  - The entropy $H(X)$ of a random variable X measures the uncertainty or information content in a system, where $P(x_i)$ is the probability of event x_i .
- **Significance**: This equation is the foundation of information theory, **enabling the quantification of information** and the development of data compression, error correction, and communication systems.

Fourth Industrial Revolution (21st Century)??

- **First Industrial Revolution**: *What is physical force?*
  This era focused on harnessing mechanical power, driven by advancements in understanding forces like gravity and motion, as defined by Newtonian physics, to revolutionize manufacturing and transportation.

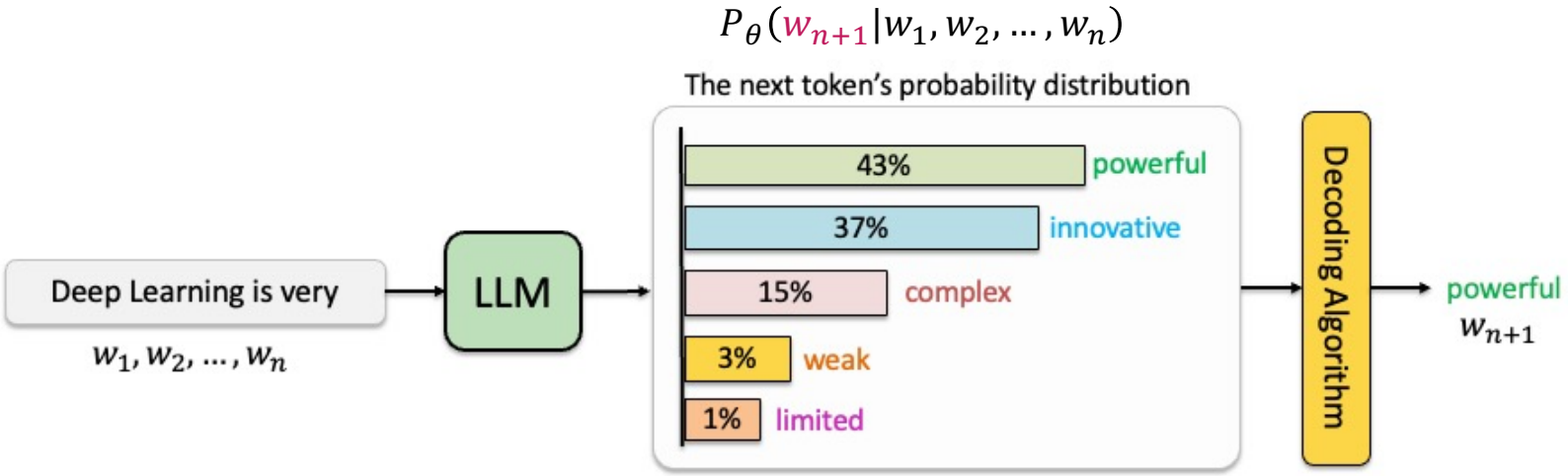
- **Second Industrial Revolution**: *What is electromagnetism?*
  This period was defined by the mastery of electricity and magnetism, leading to breakthroughs in energy generation, communication, and industrial automation, transforming society and infrastructure.

- **Third Industrial Revolution**: *What is information?*
  Marked by the rise of digital technology, computing, and the internet, this revolution centered on processing, storing, and transmitting information, reshaping economies and global connectivity.

- **Fourth Industrial Revolution**: *What questions are we trying to answer?*
  This current era explores the integration of advanced technologies like AI, biotechnology, and quantum computing, seeking to address complex challenges about humanity's future, ethics, and the relationship between humans and machines.

Fourth Industrial Revolution (21st Century)??

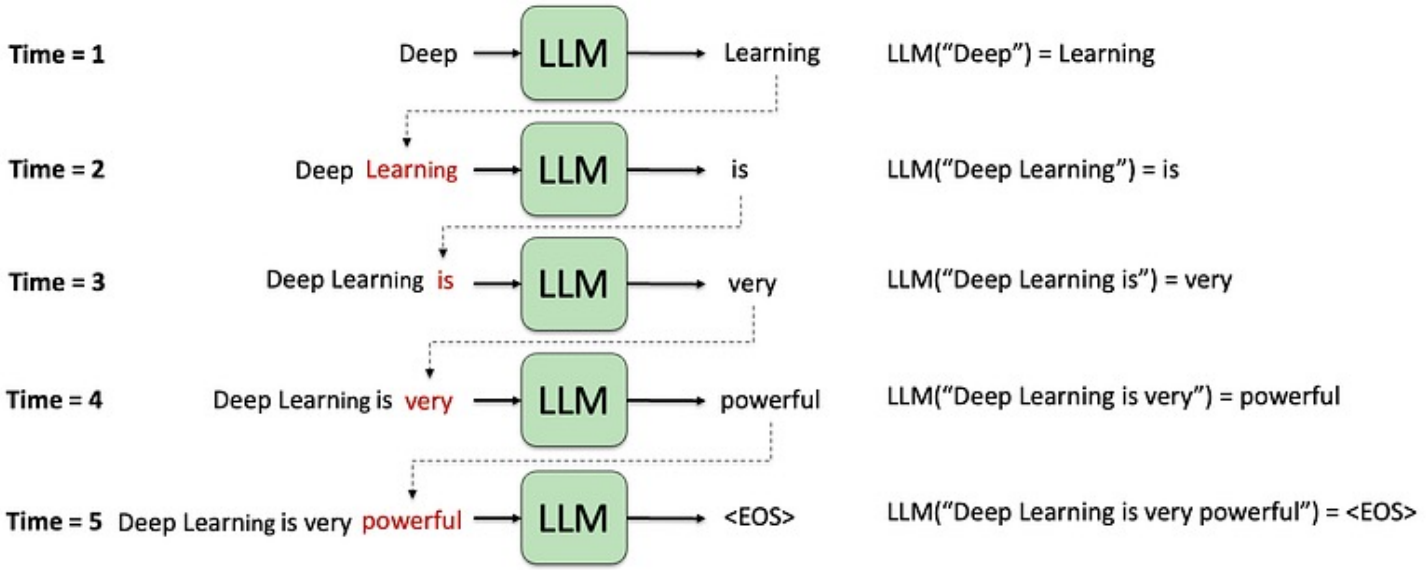
- In the AI era, we may seek to answer: **What is Intelligence?**
  - One current interpretation lies in **autoregressive language models** like ChatGPT, which predict and assign probabilities to word sequences.
  - These AI models compute the **probability distribution of the next word** w_{n+1} given previous words $w_1, w_2, \ldots, w_n$:



Fourth Industrial Revolution (21st Century)??

- **Autoregressive Language Modeling** is the task of **predicting the next word**
  - 文字接龙
  - Deep Learning is very _____.

powerful (43%) innovative (37%) complex (15%) weak (3%) limited (1%)



Hong Kong TV Game: GPT Game (文字接龙)

https://www.youtube.com/watch?v=pwTKrvqZOMo

Linear Algebra

Scalar, Vector, Matrices, and Tensors : Notations

Scalar

Rank-0 tensor

$x \in \mathbb{R}$

$x = 5438$

Vector

Rank-1 tensor

$\mathbf{x} \in \mathbb{R}^n$ or $\mathbf{x} \in \mathbb{R}^{n \times 1}$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\mathbf{x}^{\mathrm{T}} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}$$

where $\mathbf{x}^{\mathrm{T}} \in \mathbb{R}^{1 \times n}$

Matrix

Rank-2 tensor

$\mathbf{X} \in \mathbb{R}^{n \times m}$ or $\mathbf{X} \in \mathbb{R}^n \times \mathbb{R}^m$

- An m×n Matrix $\mathbf{X}$ with m rows and n columns:

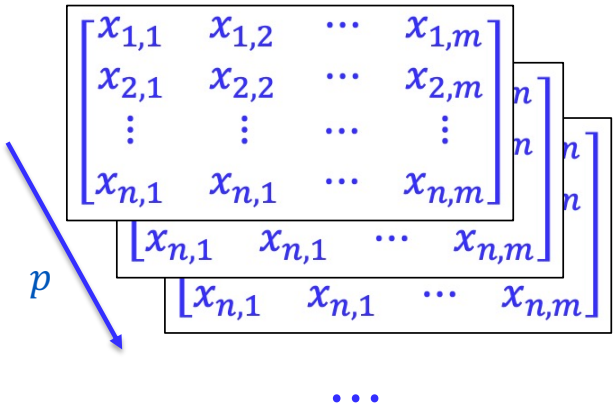
$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,m} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,m} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n,1} & x_{n,1} & \cdots & x_{n,m} \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} 1 & 3 & 4 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Rank-3 Tensors

- In general, a **tensor** is a **multidimensional array** with more than 2 axes (e.g. an RGB image)
- We write tensors in upper case, **bold typeface**
- For a **rank-3** tensor of shape $n \times m \times p$:

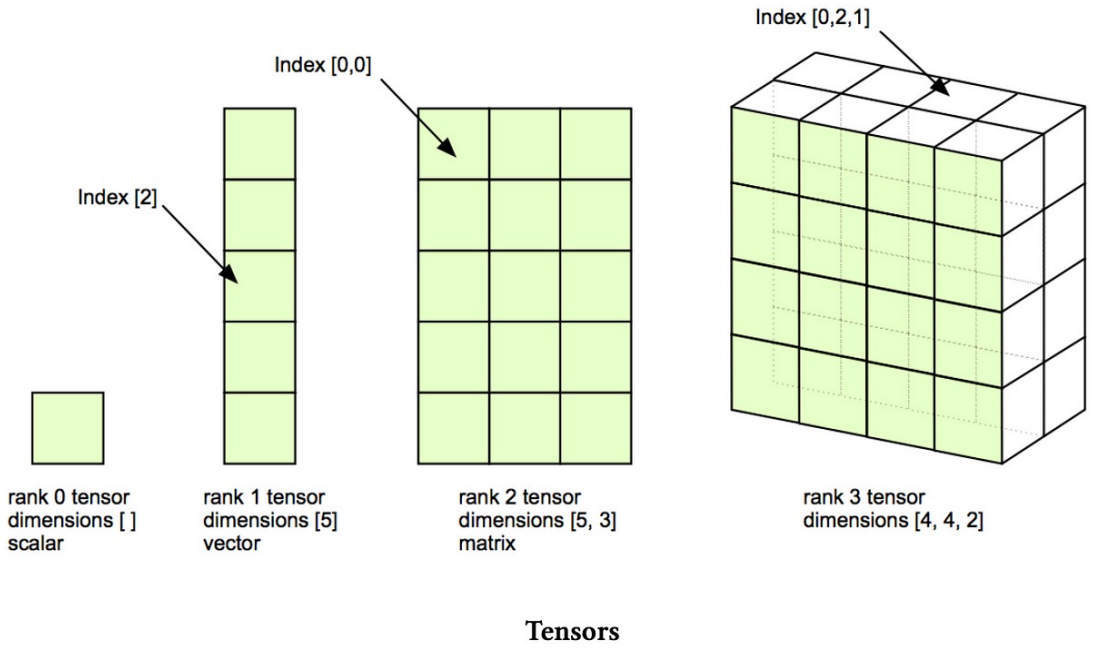
$$\mathbf{X} \in \mathbb{R}^{n \times m \times p} \text{ or } \mathbf{X} \in \mathbb{R}^{n} \times \mathbb{R}^{m} \times \mathbb{R}^{p}$$



- We identify each element of a tensor via its indices ($x_{i,j,k}$)
  - Elements of a tensor are scalars and written in lower case non-boldface font

Tensor Notations

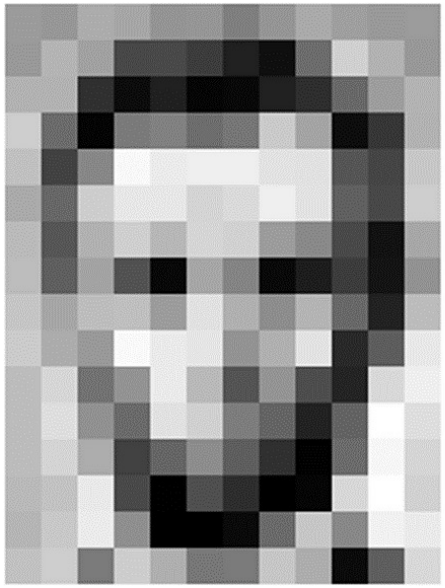
- Rank 0 Tensor, Scalar, $\mathbb{R}$
- Rank 1 Tensor, Scalar, $\mathbb{R}^n$
- Rank 2 Tensor, Scalar, $\mathbb{R}^{n \times m}$
- Rank 3 Tensor, Scalar, $\mathbb{R}^{n \times m \times p}$



https://ai.plainenglish.io/what-is-a-tensor-2715746a4785

An Example of Rank-2 Tensor (Matrix)

- **A rank 2 tensor**, commonly recognized as a matrix, encompasses arrays of values arranged in two dimensions, denoted as $\mathbb{R}^{n \times m}$.
- **Example:** In image processing, a grayscale image can be represented as a matrix of pixel values, where each element represents the intensity of light at a specific location. For instance, a 28x28 matrix could represent a grayscale image with dimensions of 28 pixels by 28 pixels.

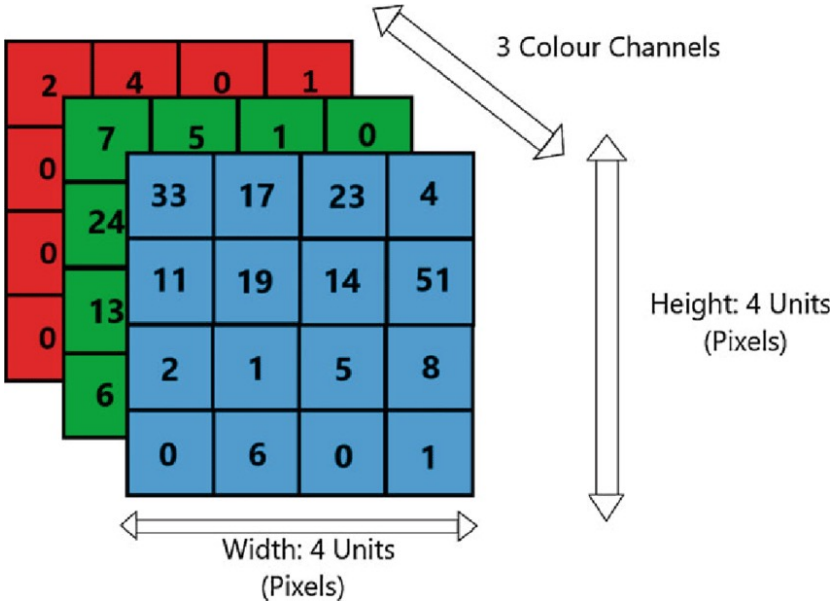


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

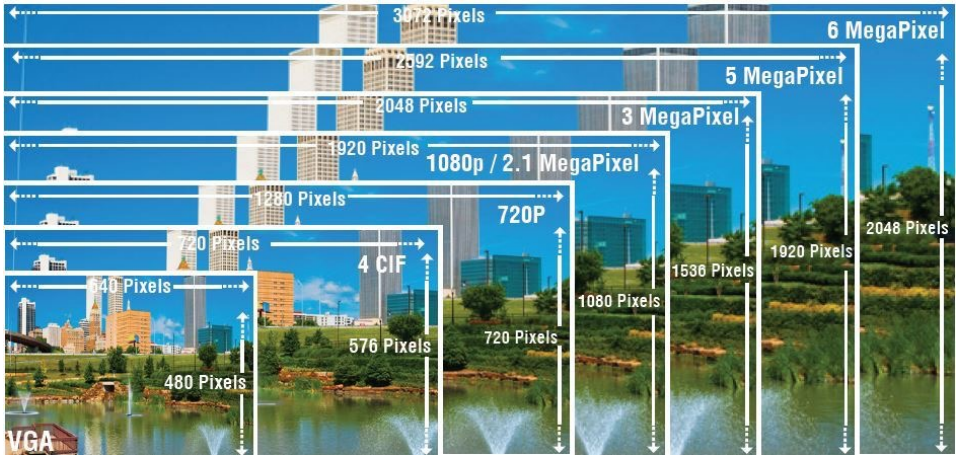
157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

An Example of a 3D Tensor in Deep Learning

- A 3D tensor of a Red-Green-Blue (RGB) image of a dimension of 4x4x3:



A 4x4 RGB color image: $\mathbf{X} \in \mathbb{R}^{4\times4\times3}$



RGB color images with different resolutions can be represented by 3D tensors

Pytorch Example: Element-wise Operations

```
>>> import torch
>>>
>>> a = torch.Tensor([26])
>>>
>>> print(type(a))
<class 'torch.Tensor'>
>>>
>>> print(a.shape)
torch.Size([1])
>>>
>>> # Creates a Random Torch Variable of size 5x3
>>> t = torch.rand(5,3)
>>> print(t)
tensor([[0.1942, 0.1000, 0.0924],
        [0.5892, 0.0238, 0.8701],
        [0.0924, 0.5316, 0.8005],
        [0.5326, 0.0825, 0.0833],
        [0.7445, 0.0825, 0.1249]])
>>>
>>>
>>> print(t.shape)
torch.Size([5, 3])
```

```
>>> import torch
>>>
>>> p = torch.rand(4,4)
>>> q = torch.rand(4,4)
>>> ones = torch.ones(4,4)
>>>
>>> print(p)
tensor([[0.3725, 0.6177, 0.1797, 0.1530],
        [0.7200, 0.7792, 0.2947, 0.1345],
        [0.1337, 0.0337, 0.7615, 0.9457],
        [0.7205, 0.3430, 0.2765, 0.5166]])
>>> print(q)
tensor([[0.2651, 0.6778, 0.1580, 0.7291],
        [0.3449, 0.8570, 0.4514, 0.3144],
        [0.4027, 0.6261, 0.9280, 0.1111],
        [0.7718, 0.7152, 0.9063, 0.6271]])
>>> print(ones)
tensor([[1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.]])
```

```
>>> print(f"Addition:{p + q}")
Addition:tensor([[0.6376, 1.2955, 0.3377, 0.8820],
        [1.0649, 1.6363, 0.7461, 0.4489],
        [0.5364, 0.6597, 1.6895, 1.0568],
        [1.4922, 1.0582, 1.1829, 1.1437]])
>>>
>>> print(f"Subtraction:{p - ones}")
Subtraction:tensor([[-0.6275, -0.3823, -0.8203, -0.8470],
        [-0.2800, -0.2208, -0.7053, -0.8655],
        [-0.8663, -0.9663, -0.2385, -0.0543],
        [-0.2795, -0.6570, -0.7235, -0.4834]])
>>>
>>> print(f"Multiplication:{p * ones}")
Multiplication:tensor([[0.3725, 0.6177, 0.1797, 0.1530],
        [0.7200, 0.7792, 0.2947, 0.1345],
        [0.1337, 0.0337, 0.7615, 0.9457],
        [0.7205, 0.3430, 0.2765, 0.5166]])
>>>
>>> print(f"Division:{p / q}")
Division:tensor([[1.4048, 0.9113, 1.1378, 0.2098],
        [2.0872, 0.9092, 0.6529, 0.4276],
        [0.3321, 0.0538, 0.8206, 8.5151],
        [0.9335, 0.4796, 0.3051, 0.8238]])
```

https://colab.research.google.com/drive/1ZrjtAJWZ1HgNTQ98nMs_SHtjA6QAbsBj

Representation of Matrix by Vectors

- **Column Vector**

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} \qquad \mathbf{c}^T = [c_1 \quad c_2 \quad \cdots \quad c_n]$$

- **Row Vector**

$$\mathbf{r} = [r_1 \quad r_2 \quad \cdots \quad r_n]$$

- **Matrix as combination of column or row vectors**

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} \leftarrow \mathbf{r}_1 \rightarrow \\ \leftarrow \mathbf{r}_2 \rightarrow \\ \leftarrow \mathbf{r}_3 \rightarrow \end{bmatrix} = \begin{bmatrix} \uparrow & \uparrow & \uparrow \\ \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 \\ \downarrow & \downarrow & \downarrow \end{bmatrix}$$

Transpose

- The transpose $\mathbf{x}^T$ of a column vector $\mathbf{x}$ become a row vector:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \qquad \mathbf{x}^T = [x_1 \quad x_2 \quad \cdots \quad x_m]$$

- The transpose $\mathbf{X}^T$ of a matrix $\mathbf{X}$ mirrors it at its main diagonal:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{bmatrix} \Rightarrow \mathbf{X}^T = \begin{bmatrix} x_{11} & x_{21} & x_{31} \\ x_{12} & x_{22} & x_{32} \end{bmatrix}$$

Adding and Subtracting Vectors and Matrices

- We **add/subtract** vectors or matrices by **adding/subtracting** them elementwise
- Examples

$$\mathbf{a} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} \Rightarrow \mathbf{a} + \mathbf{b} = \begin{bmatrix} 1+4 \\ 2+5 \\ 3+6 \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \Rightarrow \mathbf{A} + \mathbf{B} = \begin{bmatrix} 1+5 & 2+6 \\ 3+7 & 4+8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

Scalar Addition/Multiplication of Vectors and Matrices

- We can also **add a scalar to a vector/matrix** or **multiply a vector/matrix by a scalar**:

$$\mathbf{d} = a\mathbf{b} + c \qquad \mathbf{D} = a\mathbf{B} + c$$

- Example for scalar addition and scalar multiplication:

$$2\begin{bmatrix}5 & 6\\ 7 & 8\end{bmatrix} + 1 = \begin{bmatrix}2\times5+1 & 2\times6+1\\ 2\times7+1 & 2\times8+1\end{bmatrix} = \begin{bmatrix}11 & 13\\ 15 & 17\end{bmatrix}$$

- Example for **broadcasting** (shape of vector determines which type):

$$\begin{bmatrix}5 & 6\\ 7 & 8\end{bmatrix} + \begin{bmatrix}1\\ 2\end{bmatrix} = \begin{bmatrix}5+1 & 6+1\\ 7+2 & 8+2\end{bmatrix} = \begin{bmatrix}6 & 7\\ 9 & 10\end{bmatrix}$$

$$\begin{bmatrix}5 & 6\\ 7 & 8\end{bmatrix} + \begin{bmatrix}1 & 3\end{bmatrix} = \begin{bmatrix}5+1 & 6+3\\ 7+1 & 8+3\end{bmatrix} = \begin{bmatrix}6 & 9\\ 8 & 11\end{bmatrix}$$

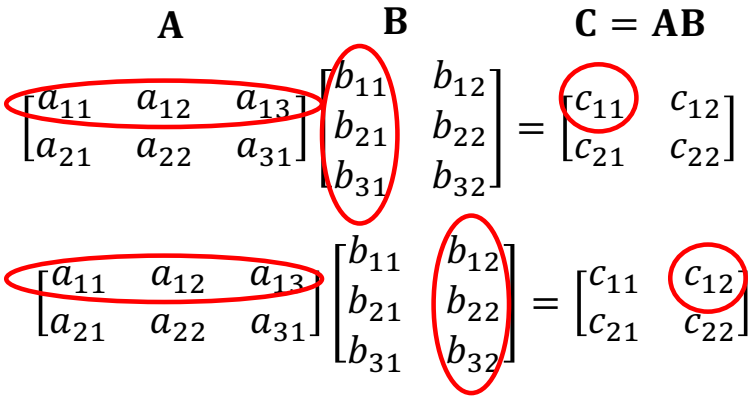
Vector and Matrix Multiplications

- Two vectors or matrices $\mathbf{A}$ and $\mathbf{B}$ can be multiplied if $\mathbf{A}$ has the same number columns as $\mathbf{B}$ has rows (i.e. $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times m}$):

$$\mathbf{C} = \mathbf{AB}$$

- The matrix product is defined by

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



Examples

- Example for **matrix product**:

$$\mathbf{A} = \begin{bmatrix} 3 & 1 \\ 2 & 1 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix} \Rightarrow \mathbf{AB} = \begin{bmatrix} 3\times1 + 1\times3 & 3\times2 + 1\times1 \\ 2\times1 + 1\times3 & 2\times2 + 1\times1 \end{bmatrix} = \begin{bmatrix} 6 & 7 \\ 5 & 5 \end{bmatrix}$$

- Example for an **inner product** between two vectors (dot product):

$$\mathbf{a} = \begin{bmatrix} 3 \\ 2 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 1 \\ 3 \end{bmatrix} \Rightarrow \mathbf{a}^T\mathbf{b} = \begin{bmatrix} 3 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} = 3\times1 + 2\times3 = 9$$

- Example for an **outer product** between two vectors: (= rank-1 matrix)

$$\mathbf{a} = \begin{bmatrix} 3 \\ 2 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 1 \\ 3 \end{bmatrix} \Rightarrow \mathbf{ab}^T = \begin{bmatrix} 3 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 3 \end{bmatrix} = \begin{bmatrix} 3\times1 & 3\times3 \\ 2\times1 & 2\times3 \end{bmatrix} = \begin{bmatrix} 3 & 9 \\ 2 & 6 \end{bmatrix}$$

Basic Matrix Operations

- **Addition**

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

Element-wise Addition

- **Subtraction**

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

Element-wise Subtract

- **Multiplication**

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae+bg & af+bh \\ ce+dg & cf+dh \end{bmatrix}$$

Dot product of each row by each column

Elementwise Vector and Matrix Multiplications

- For vectors $\mathbf{x}$ and $\mathbf{y}$ given by $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

- The elementwise multiplication between these two vectors is

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

Special Matrices

Identity Matrix, $\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Zero Matrix, $\mathbf{0} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

Matrix Transpose,

$$\mathbf{X} = \begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{bmatrix} \qquad \mathbf{X}^T = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix}$$

Symmetric Matrix,

$$\mathbf{X} = \begin{bmatrix} a_1 & b_1 & c_1 \\ b_1 & b_2 & c_2 \\ c_1 & c_2 & c_3 \end{bmatrix} = \mathbf{X}^T$$

Diagonal Matrix,

$$\mathbf{D} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix}$$

Useful Matrix Properties

- **Commutative Property**
  - $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$
  - $\mathbf{AB} \neq \mathbf{BA}$
- **Associative Property**
  - $(\mathbf{A} + \mathbf{B}) + \mathbf{C} = \mathbf{A} + (\mathbf{B} + \mathbf{C})$
  - $(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$
- **Distributive Property**
  - $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$

- **Multiplicative Identity**
  - $\mathbf{AI} = \mathbf{A}$
- **Additive Property of Zero**
  - $\mathbf{A} + \mathbf{0} = \mathbf{A}$
- **Transpose Property**
  - $(\mathbf{AB})^T = \mathbf{A}^T\mathbf{B}^T$

Vector Norm

- The norm of a vector is a function that maps a vector to a positive value
- p-norm for $p \geq 1$

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

$$\|\mathbf{x}\|_p = \left(\sum_{j=1}^{d} |x_j|^p \right)^{1/p} \qquad \text{where } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

$$\|\mathbf{x}\|_p^p = \sum_{j=1}^{d} |x_j|^p$$

L1-Norm and L2-Norm

- **L1-Norm (Euclidean norm)**

$$\|\mathbf{x}\|_1 = \sum_{j=1}^{d} |x_j|$$

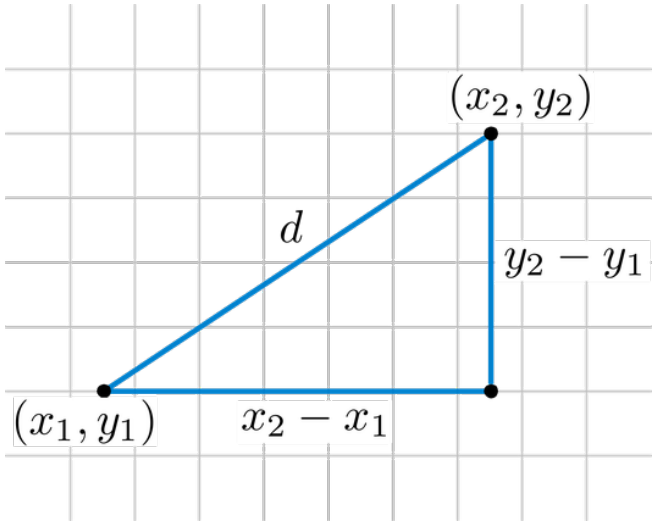
- **L2-Norm (Euclidean norm)**

$$\|\mathbf{x}\|_2 = \left(\sum_{j=1}^{d} (x_j)^2 \right)^{1/2} = \sqrt{\sum_{j=1}^{d} x_j^2} = (\mathbf{x}^T \mathbf{x})^{1/2}$$

Manhattan Distance L1 and Euclidian Distance L2

- **Manhattan Distance L1**

$$\|\mathbf{x}-\mathbf{y}\|_1 = \sum_{j=1}^{d} |x_j - y_j|$$

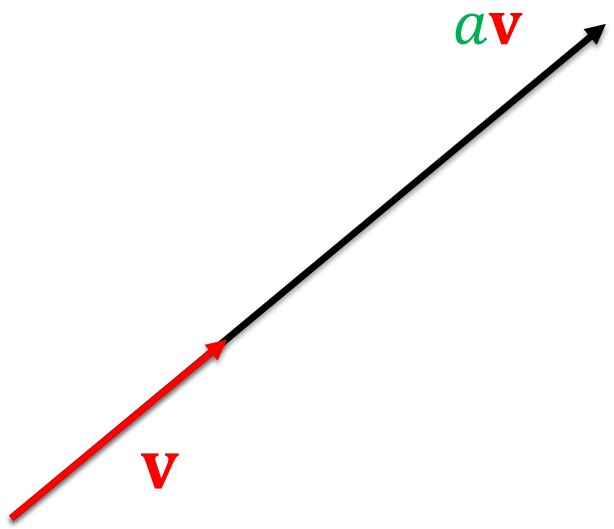


- **Euclidian Distance L2**

$$\|\mathbf{x}-\mathbf{y}\|_2 = \sqrt{\sum_{j=1}^{d} (x_j - y_j)^2}$$

Physical Meanings of Scale Product : $a\mathbf{v}$

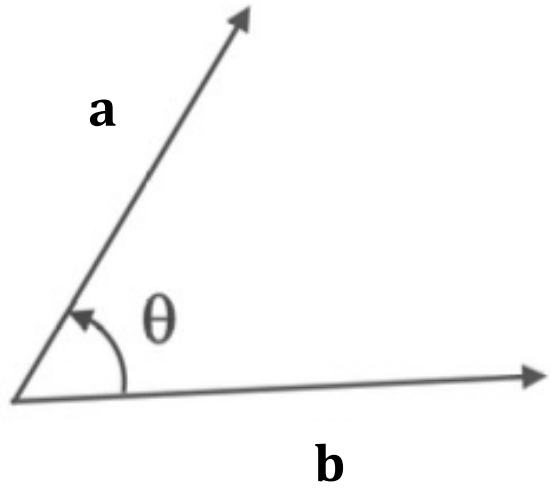
- a is scale and $\mathbf{v}$ is a vector
- $a\mathbf{v} = a\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} a \cdot v_1 \\ a \cdot v_2 \end{bmatrix}$
- Change only the length (scaling), but keep direction fixed



Inner or Dot Product

- The inner product of two vectors $\mathbf{a}$ and $\mathbf{b}$,

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$



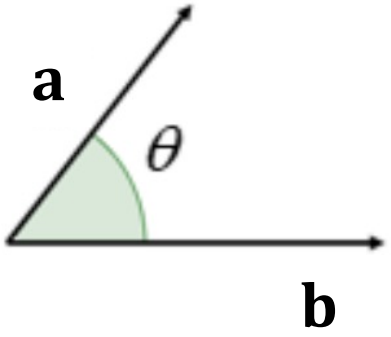
is defined as $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^{\mathrm{T}}\mathbf{b} = \mathbf{b}^{\mathrm{T}}\mathbf{a} = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$

$$\mathbf{a} \cdot \mathbf{b} = \sum_{j=1}^{n} a_j \cdot b_j = \|\mathbf{a}\| \|\mathbf{b}\| \cos(\theta)$$

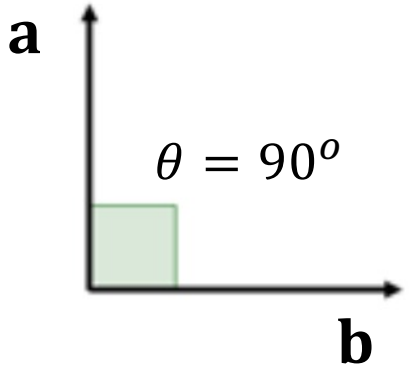
Note that the inner product is a scalar.

Cosine Distance Between Two Vectors

- If θ is the angle between $\mathbf{a}$ and $\mathbf{b}$ then
  - $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T\mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\cos(\theta)$



- $\mathbf{a} \cdot \mathbf{b}$ are orthogonal (perpendicular) if and only if
  - $\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\cos(\pm 90^o) = 0$



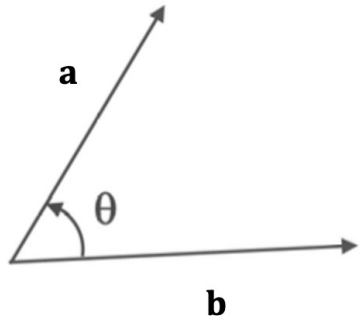
- **Cosine Distance** of two vectors $\mathbf{a}$ and $\mathbf{b}$ is defined as

$$\cos(\theta) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$$

Cosine Similarity (or Distance)

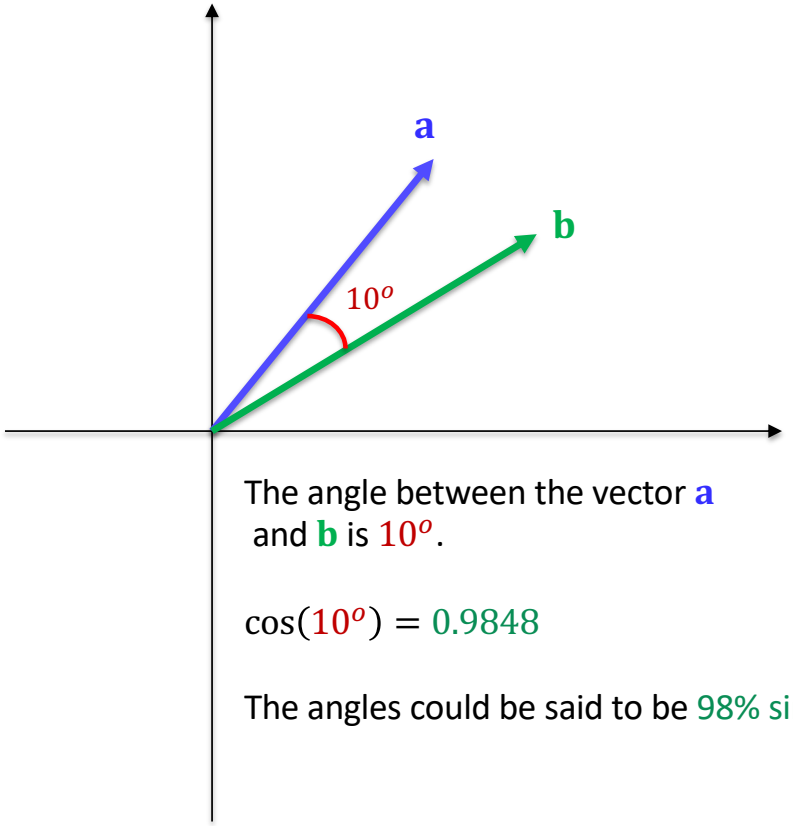
- Cosine similarity is **a measure of similarity between two non-zero vectors.** It is calculated as the cosine of the angle θ between the two vectors.
- Given two n-dimensional vectors $\mathbf{a}$ and $\mathbf{b}$, their cosine similarity is defined as:

$$\text{sim}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} = \cos(\theta)$$



  - The cosine similarity calculates the angle between two vecto
  - **Cosine Similarity lies in the ranges** [-1 to 1]:
    - when $\text{sim}(\mathbf{a}, \mathbf{b}) = 1$, the vectors are strongly similar with $\theta = 0^o$
    - when $\text{sim}(\mathbf{a}, \mathbf{b}) = 0$, the two vectors are orthogonal with $\theta = \pm 90^o$
    - when $\text{sim}(\mathbf{a}, \mathbf{b}) = -1$, we have strongly dissimilar vectors with $\theta = 180^o$
    - All the intermediate values indicate the respective degree of similarity

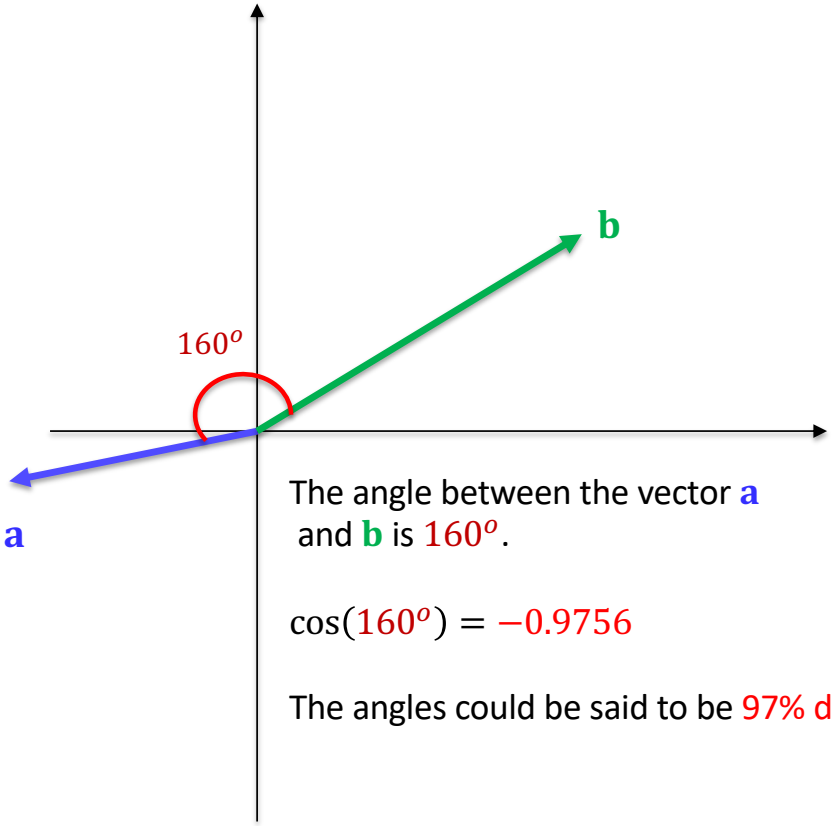
Cosine Similarity – Geometric Interpretation



The angle between the vector **a** and **b** is 10^o.

$\cos(10^o) = 0.9848$

The angles could be said to be 98% similar.



The angle between the vector **a** and **b** is 160^o.

$\cos(160^o) = -0.9756$

The angles could be said to be 97% dissimilar.

Cosine Similarity Example

- Let's suppose that we have two vectors $\mathbf{a} = [4, 3]^{\mathrm{T}}$ and $\mathbf{b} = [8, 6]^{\mathrm{T}}$.
- First, we compute their dot product as follow:
  - $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T\mathbf{b} = [4 \quad 3]\begin{bmatrix}8\\6\end{bmatrix} = 4\times8 + 3\times6 = 50$
  - $\|\mathbf{a}\| = \sqrt{4^2 + 3^2} = \sqrt{25} = 5$ and $\|\mathbf{b}\| = \sqrt{8^2 + 6^2} = \sqrt{100} = 10$
- Then, we compute their cosine similarity as follows:
  - $\mathrm{sim}(\mathbf{a}, \mathbf{b}) = \cos(\theta) = \frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|} = \frac{14}{5\times10} = \frac{14}{50} = 0.28$

Calculus

First Scientific Revolution (16th -17th Century)

- The First Scientific Revolution signalled the shift from medieval to **modern science**, highlighted by Isaac Newton's pivotal contributions.
- **Newton's Second Law of Motion:**

$$F = ma = \frac{d}{dt}(m \cdot v) = m\frac{d^2x}{dt^2}$$

Isaac Newton (1643–1727), the physicist who formulated the laws of motion and invented Calculus

- This equation describes the relationship between force (F), mass (m), and acceleration (a). It is fundamental to classical mechanics.
- **Force** is equal to **rate of change of momentum** (mass · velocity)

Differential Calculus: The Derivative

- **Differential Calculus** is fundamentally about understanding the **rate of change**
- For a function $f: \mathbb{R} \to \mathbb{R}$ (input and output are scalars), the derivative $f'(t)$ is a function describing the **instantaneous rate of change** or the slope of $f(t)$
- Mathematically, the derivative is defined as:

$$f'(t) = \frac{df(t)}{dt} = \lim_{\Delta \to 0} \frac{f(t+\Delta) - f(t)}{\Delta}$$

- For example: $f(t) = 3t^2 - 4t \Rightarrow f'(t) = 6t - 4 \Rightarrow f'(1) = 2$

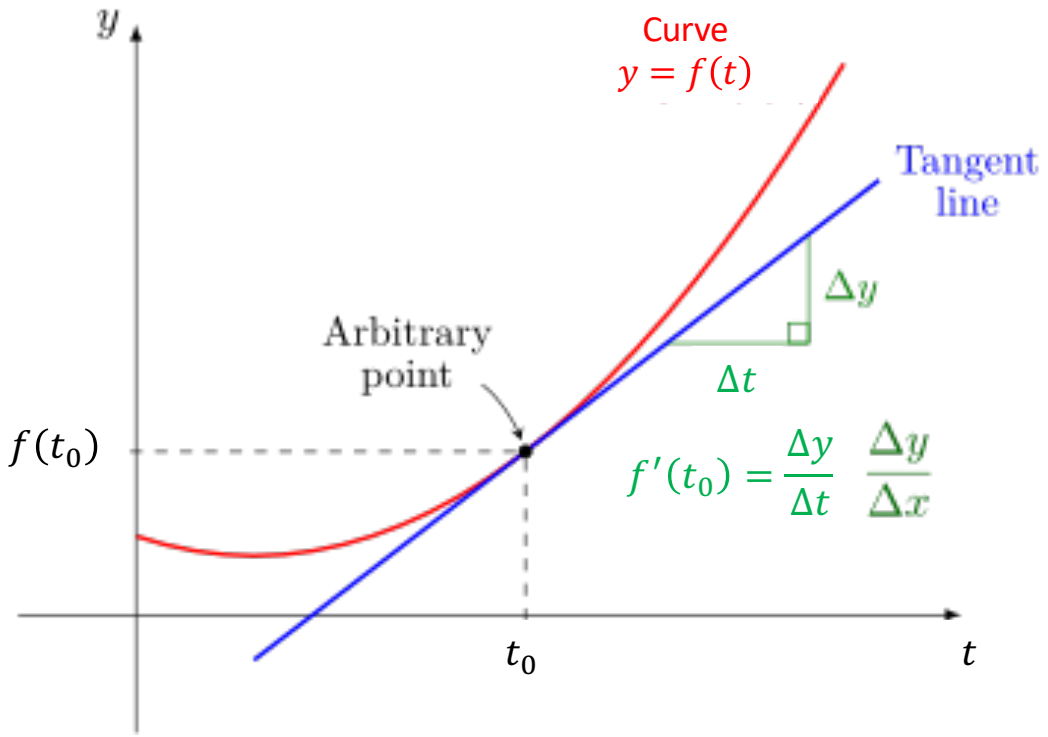
Recap: Derivative Table

- $f(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$
- $f(x) = ax \quad \rightarrow \quad \frac{df}{dx} = ax$
- $f(x) = x^n \quad \rightarrow \quad \frac{df}{dx} = nx^{n-1}$
- $f(x) = a^x \quad \rightarrow \quad \frac{df}{dx} = (\ln a)a^x$
- $f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$

- $f(x) = \log_a x \quad \rightarrow \quad \frac{df}{dx} = (\log_a e)\frac{1}{x}$
- $f(x) = \ln x \quad \rightarrow \quad \frac{df}{dx} = \frac{1}{x}, x \neq 0$
- $f(x) = \sin(x) \quad \rightarrow \quad \frac{df}{dx} = \cos(x)$
- $f(x) = \cos(x) \quad \rightarrow \quad \frac{df}{dx} = -\sin(x)$
- $f(x) = \tan(x) \quad \rightarrow \quad \frac{df}{dx} = -\sec^2(x)$

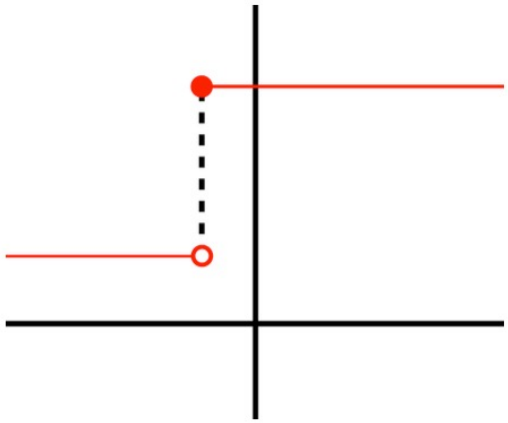
Geometric Interpretation

- The derivative at an arbitrary point (t_0) is equivalent to the **slope of the tangent line** at that point.

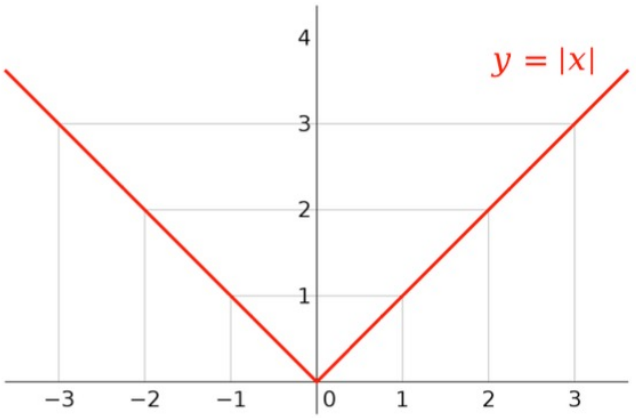


Differentiability

- A function is differentiable at a point if the limit defining the derivative exists.
- Functions that are not continuous or have sharp points (like $y=|x|$) are **not differentiable** at those points.



Not continuous



A sharp point

Multi-Variable Calculus: Partial Derivatives

- **Function of Multiple Variables:** Calculus extends to functions where the input consists of n variables, resulting in a scalar output ($f: \mathbb{R}^n \rightarrow \mathbb{R}$)
  - *Example:* $f(\mathbf{x}) = f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = f(x_1, x_2) = x_1^2 + 3x_1x_2 + x_2^2$
- **Partial Derivatives:** Since the input has multiple variables, we calculate partial derivatives ($\frac{\partial f}{\partial x_i}$). For examples
  - $\frac{\partial f}{\partial x_1} = 2x_1 + 3x_2$ and $\frac{\partial f}{\partial x_2} = 3x_2 + 2x_2$
- The partial derivative with respect to one variable is found by treating the other variables as constants.

Multi-Variable Calculus: The Gradient

- **The Gradient $\nabla f(\mathbf{x})$:** The first-order derivative of a vector-to-scalar function is called the **Gradient**. The gradient is typically represented as a **column vector**:

$$\nabla f(\mathbf{x}) = \nabla f\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}\right) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

For example: $f(\mathbf{x}) = f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = x_1^2 + x_2^2 \Rightarrow \nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial (x_1^2 + x_2^2)}{\partial x_1} \\ \frac{\partial (x_1^2 + x_2^2)}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix}$

Matrix Calculus: The Jacobian Matrix

- **Vector-to-Vector Functions:** The Jacobian Matrix is used when dealing with multiple functions of multiple variables ($f: \mathbb{R}^n \rightarrow \mathbb{R}^m$).
  - **Example:** $\mathbf{f}(\mathbf{x}) = \mathbf{f}\left(\begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix}\right) = \begin{bmatrix} x_1^2 + x_2 \\ x_1 + x_2^2 \end{bmatrix}$

- **Definition:** The **Jacobian Matrix** $J(\mathbf{x})$ is the derivative of a vector-to-vector function. For example:

$$J(\mathbf{x}) = \nabla \mathbf{f}(\mathbf{x}) = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 & 1 \\ 1 & 2x_2 \end{bmatrix}$$

The Jacobian Matrix

- $J(\mathbf{x})$ has dimensions $m \times n$ (where n is the number of inputs and m is the number of outputs).

$$J(\mathbf{x}) = \nabla \mathbf{f}(\mathbf{x}) = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \nabla f_1(\mathbf{x})^T \\ \nabla f_2(\mathbf{x})^T \\ \vdots \\ \nabla f_m(\mathbf{x})^T \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \frac{\partial f_1(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial f_2(\mathbf{x})}{\partial x_1} & \frac{\partial f_2(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m(\mathbf{x})}{\partial x_1} & \frac{\partial f_m(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_m(\mathbf{x})}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

  - Each row in the Jacobian is the gradient (∇) of one of the output functions.
  - The component in row i and column j is the partial derivative of the i-th output function (f_i) with respect to the j-th input component (x_j).

Application in AI: Differential Programming

- **Differential Programming** is the field relying on the application of calculus to automatically calculate derivatives.
  - The **key idea** behind training a neural network (identifying its weights) is **Gradient Descent**.
  - Training involves iteratively minimizing a **Cost Function** $\mathcal{L}(\theta_t)$ that measures the error of the neural network.
  - Parameters (θ_t) are updated based on the calculated gradient ($\nabla_\theta \mathcal{L}(\theta_t)$) and a learning rate ($\eta$):

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta \mathcal{L}(\theta_t)$$

The Gradient Descent Algorithm

Step 0: Randomly initialize weights and biases parameters: $\theta = \{\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(2)}, \ldots, \mathbf{W}^{(L)}, \mathbf{b}^{(L)}\}$

Step 1: Compute the cost function $\mathcal{L}(\theta)$, which measures how well the model is performing of the dataset.

Step 2: Find the gradients of the cost function with respect to each parameters $\nabla_\theta \mathcal{L}(\theta_t)$

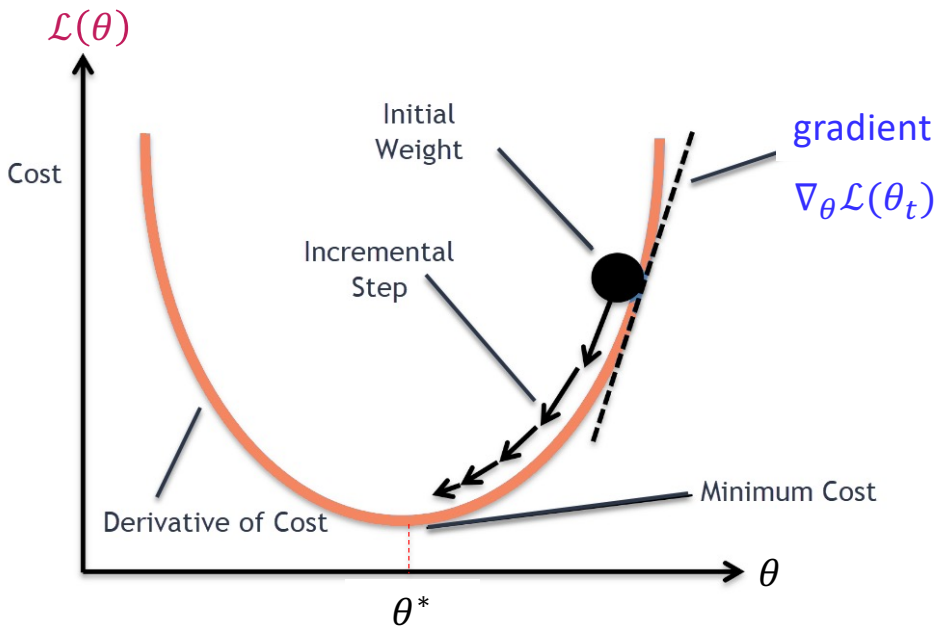
Step 3: Update the parameters by

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta \mathcal{L}(\theta_t)$$

where η is the learning rate that determines how big the updates should be in each iteration t.

- **Repeat** the above steps 1 to 3, unit the cost is low enough or convergence.

$$\theta^* = \min_\theta \mathcal{L}(\theta) = \min_\theta \frac{1}{N} \sum_{i=1}^{N} \ell\left(y^{(i)}, f_\theta(\mathbf{x}^{(i)})\right)$$



Chain Rule of Differential Calculus

- **The Chain Rule:** This fundamental rule expresses the derivative of a composite function (like $y = f\big(g(x)\big)$) in terms of the derivatives of its component functions (f and g).
- Single-variable example:
  - If $y = f(u)$ and $u = g(x)$ then we can use chain rule to find the derivative as

$$\frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx}$$

  - If $y = u^3$ and $u = 2x^2 + x$, then we have

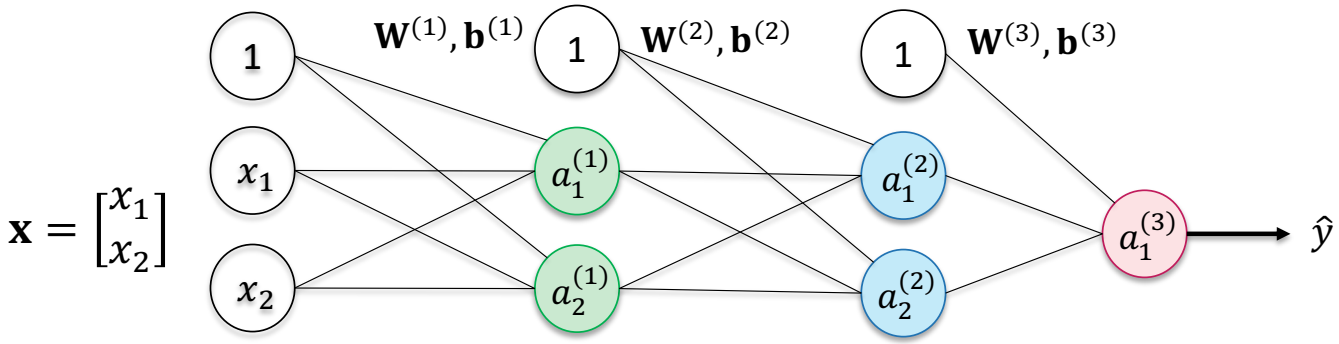
$$\frac{dy}{du} = 3u^2 \text{ and } \frac{du}{dx} = 4x + 1 \qquad \Rightarrow \frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx} = 3u^2 \cdot (4x + 1) = 3(2x^2 + x)^2(4x + 1)$$

Backpropagation

- **Backpropagation** is an **efficient algorithm** to compute gradients $\nabla_\theta \mathcal{L}(\theta_t)$ of a loss function with respect to all the parameters in a neural network by using the **chain rule** of calculus recursively.
- **Key Idea:** Instead of computing the gradient of the loss $\frac{\partial \mathcal{L}(\theta)}{\partial \theta}$ for every parameter θ separately (which would be very slow), backpropagation computes gradients layer-by-layer **backwards**, reusing intermediate results.

Efficient Gradient Computation using the Chain Rule

- In a neural network, which is a composite function of many layers, the chain rule is utilized efficiently through **matrix multiplication** to compute the overall gradient



Step 1: $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} a_1^{(1)} \\ a_2^{(1)} \end{bmatrix}$

Step 2: $\begin{bmatrix} a_1^{(1)} \\ a_2^{(1)} \end{bmatrix} \rightarrow \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \end{bmatrix}$

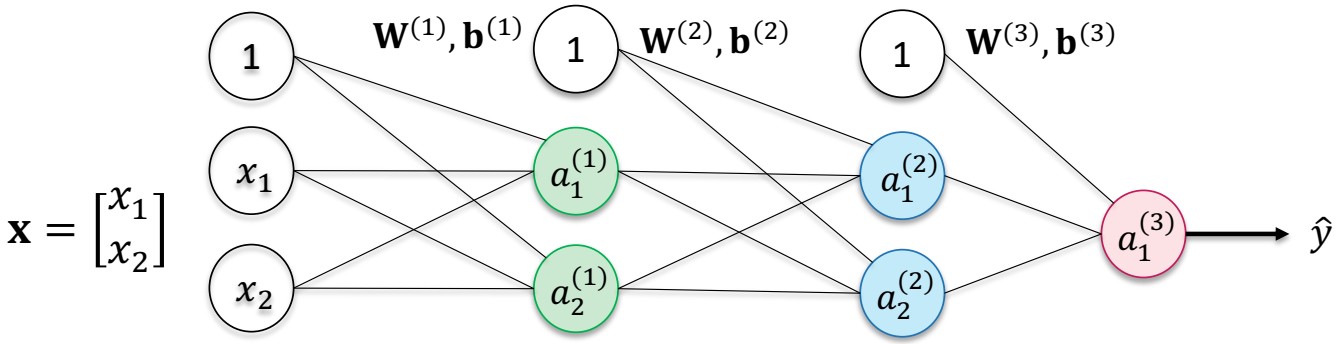
Step 3: $\begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \end{bmatrix} \rightarrow \hat{y}$

$$J_1 = \begin{bmatrix} \frac{\partial a_1^{(1)}}{\partial x_1} & \frac{\partial a_1^{(1)}}{\partial x_2} \\ \frac{\partial a_2^{(1)}}{\partial x_1} & \frac{\partial a_2^{(1)}}{\partial x_2} \end{bmatrix}$$

$$J_2 = \begin{bmatrix} \frac{\partial a_1^{(2)}}{\partial a_1^{(1)}} & \frac{\partial a_1^{(2)}}{\partial a_2^{(1)}} \\ \frac{\partial a_2^{(2)}}{\partial a_1^{(1)}} & \frac{\partial a_2^{(2)}}{\partial a_2^{(1)}} \end{bmatrix}$$

$$J_3 = \begin{bmatrix} \frac{\partial \hat{y}}{\partial a_1^{(2)}} \\ \frac{\partial \hat{y}}{\partial a_2^{(2)}} \end{bmatrix}$$

Backpropagation



- By representing the derivative of each step (or layer) in the neural network as a Jacobian matrix (J_1, J_2, J_3), the total gradient is calculated by multiplying these matrices.
- *Matrix Multiplication Example:*

$$\begin{bmatrix} \frac{\partial \hat{y}}{\partial x_1} \\ \frac{\partial \hat{y}}{\partial x_2} \end{bmatrix} = J_1 \cdot J_2 \cdot J_3$$

- This systematic calculation allows for efficient gradient computation for large networks.

Probability and Random Variables

Empirical Definition of Probability

- The probability of any event E can be defined as:

$$P(E) = \lim_{n \to \infty} \frac{\text{count}(E)}{N}$$

where $\text{count}(E)$ is the number of times that E occurred in N experiments

  - As you repeat an experiment many times, the ratio approaches the true probability
  - Also interpreted as the **chance** of E occurring

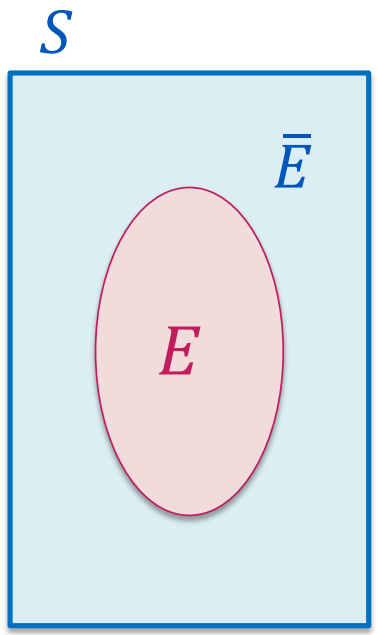
Probability of Equally Likely Outcomes

$$P(E) = \frac{number\ of\ outcomes\ in\ E}{number\ of\ outcomes\ in\ S} = \frac{|E|}{|S|}$$

- **Examples**
  - Coin flip: S = {Head, Tails}
  - Flipping two coins: S = {(H, H), (H, T), (T, H), (T, T)}
  - Roll of 6-sided die: S = {1, 2, 3, 4, 5, 6}

Core Probability Identities

- For an event E and a sample space S
  - $0 \leq P(E) \leq 1$
  - $P(S) = 1$
  - $P(E) = 1 - P(\bar{E})$



Log Probabilities

$$\log P(E) \text{ (natural or base-10 log)}$$

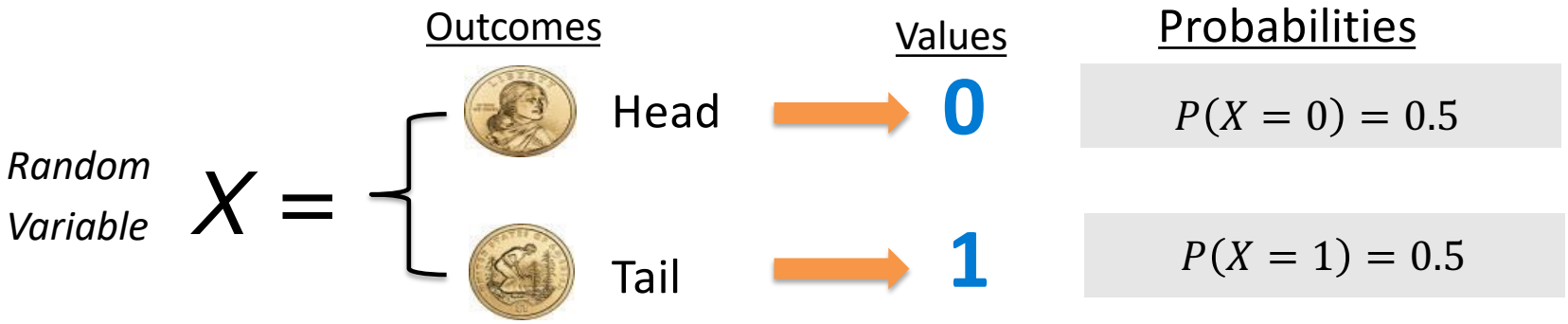
- Why use them?
  - Handles tiny probabilities (no underflow)
  - Turns products into sums (faster for computers)
- Example:

$$\log(P_1 \cdot P_2 \cdot \cdots \cdot P_n) = \log P_1 + \log P_2 + \cdots + \log P_n$$

Random Variables

A random variable X maps outcomes to numbers.

- **Discrete random variables**: Finite/countable values
  - Examples: **Die roll:** $X = 1,2,3,4,5,6$ and **coin flips :** $X = 0,1,$



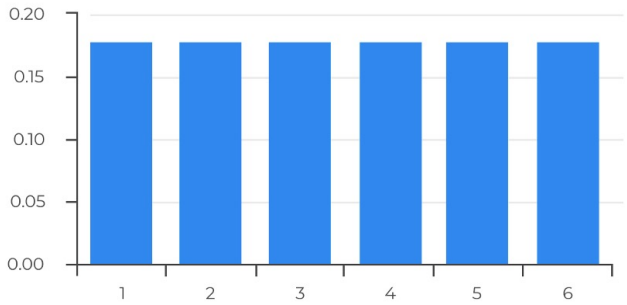
- **Continuous random variables**: Real-number values
  - Examples: Height, time, temperature

Probability Mass Function (PMF)

- A **PMF** is used to describe the **probability distribution** of **discrete random variables**.
- **Roll a fair die**: 6-sided dice with equal probability 1/6 for each of the 6 numbers
  - Uppercase letter X for a random variable.
  - Lowercase letter x for an observed value.

$$P(X = 1) = P(X = 2) = P(X = 3) = P(X = 4) = P(X = 5) = P(X = 6) = \frac{1}{6}$$

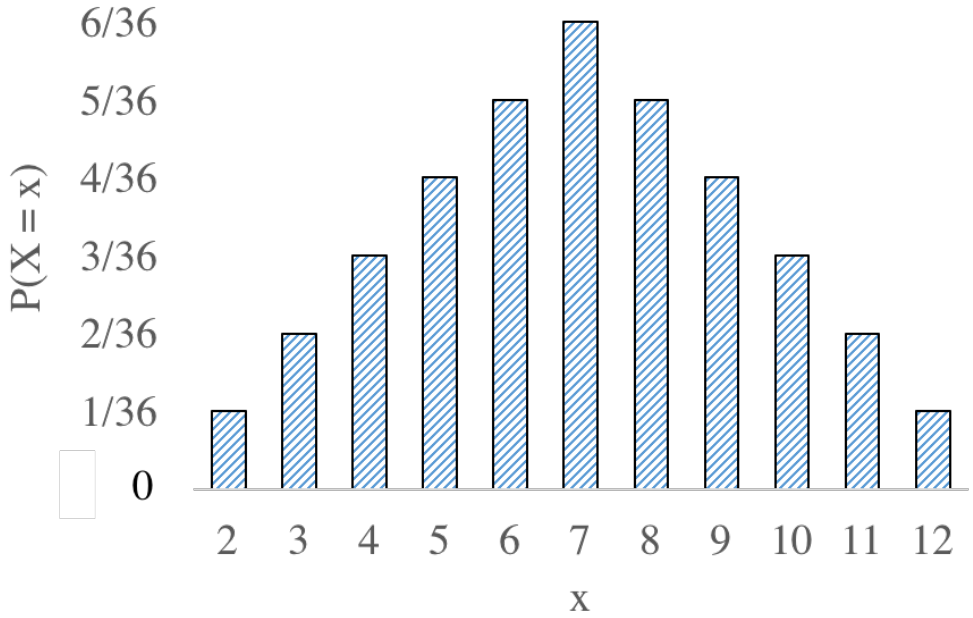
  - Short-hand notation: $P(x)$ for $P(X = x) \in \{1,2,3,4,5,6\}$



The PMF of Fair die discrete random variable is a Uniform Probability Distribution

The PMF of the Sum of Two Dice Rolls

- The sum of two dice example in the equally likely probability section. Again, the information that is provided in these graphs is the likelihood of a random variable taking on different values.

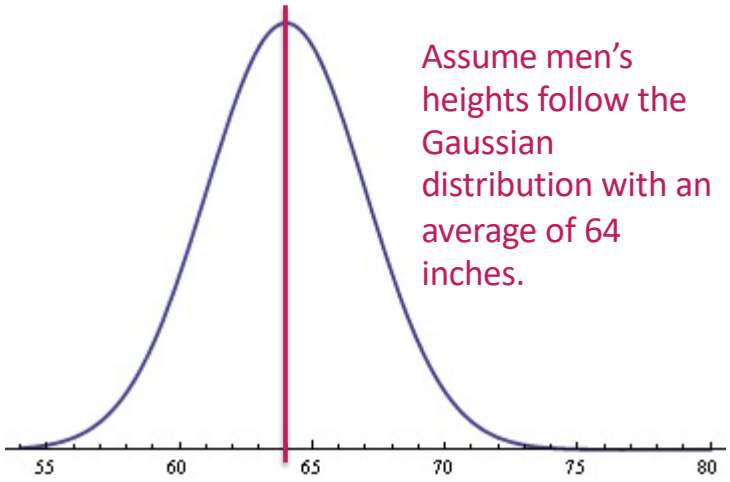


Probability Density Function (pdf)

- A **pdf** is used to describe the **probability distribution** of continuous random variables *x*.
  - Example: The PDF for adult male height might peak near 64 inches, showing higher likelihood near the mean and lower likelihood farther away.
  - **Gaussian (Normal) pdf**

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- $p(x)$ is **not** a probability; it's a density.
- The actual probability is the area under the curve:
  - $P(a \leq X \leq b) = \int_a^b p(x)dx$
  - $P(X = x) = 0$ for any exact point x (because it's continuous)

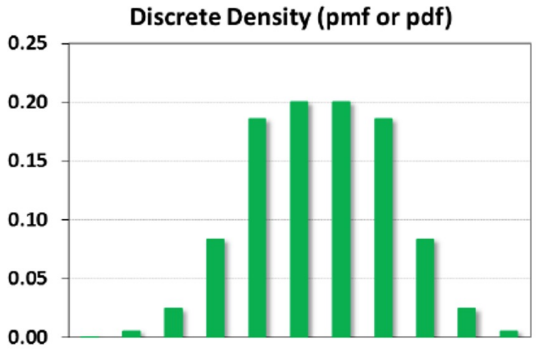


Assume men's heights follow the Gaussian distribution with an average of 64 inches.

Properties of PDF/PMF

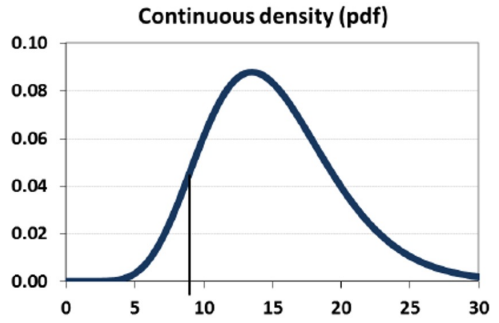
- Key Properties
  - Discrete:

$$\sum_x P(x) = 1$$



  - Continuous:

$$\int p(x)dx = 1$$



Expectation of a Random Variable

- The expectation of a random variable is a weighted average of the possible values of x can take; each value being weighted according to the probability of that event.
- For discrete random variables, the expectation is defined as

$$\mathbb{E}_x[P(x)] = \sum_x x\, P(x)$$

- For continuous random variables, the expectation is defined as

$$\mathbb{E}_x[p(x)] = \int x\, p(x) dx$$

Variance of a Random Variable

- The variance is a measure of the "spread" of a random variable around the mean. Variance for a random variable, x, with expected value $\mathbb{E}[x] = \mu$ is:

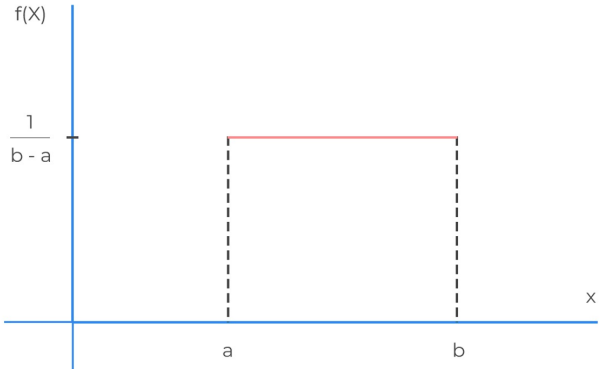
$$\text{Var}[x] = \sigma^2 = \mathbb{E}[(x - \mu)^2]$$

- Semantically, this is the average distance of a sample from the distribution to the mean. When computing the variance often we use a different (equivalent) form of the variance equation:

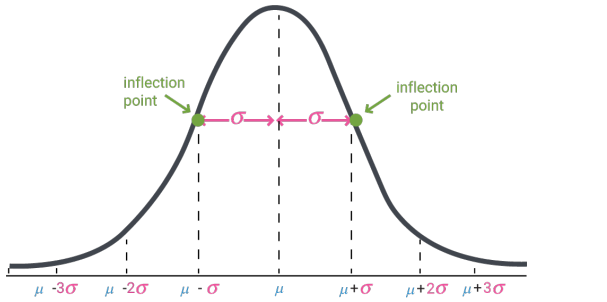
$$\text{Var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$$

Common Distributions

- **Uniform Distributions:** constant probability
  - $p(x) := \mathcal{U}(a, b)$
  - $p(x) = \begin{cases} \frac{1}{b-a} & a \leq x \leq b \\ 0 & otherwise \end{cases}$



- **Normal Distributions:** bell-shaped, defined by mean μ and variance σ^2
  - $p(x) := \mathcal{N}(\mu, \sigma^2)$
  - $p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$



Calcworkshop.com

Multivariate Normal Distribution

- A multivariate normal distribution (also called a **multivariate Gaussian distribution**) generalizes the one-dimensional normal distribution to multiple dimensions.
- It is defined by
  - **Mean vector** (expected values): $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_d]^T$,
  - **Covariance matrix $\boldsymbol{\Sigma}$:** A $d \times d$ symmetric, positive semi-definite matrix where

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{12} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{d1} & \sigma_{12} & \cdots & \sigma_d^2 \end{bmatrix}$$

Captures variances (on the diagonal) and pairwise covariance (off-diagonal).

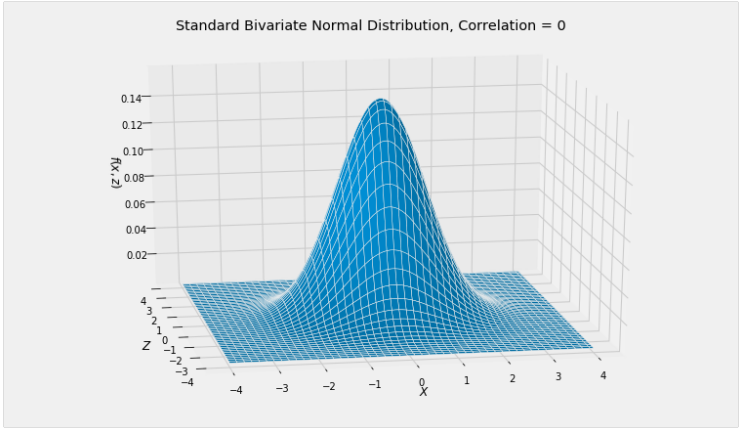
PDF of Multivariate Normal Distribution

- For a d-dimensional random vector $\mathbf{x} \in \mathbb{R}^d$, the PDF is:

$$p(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

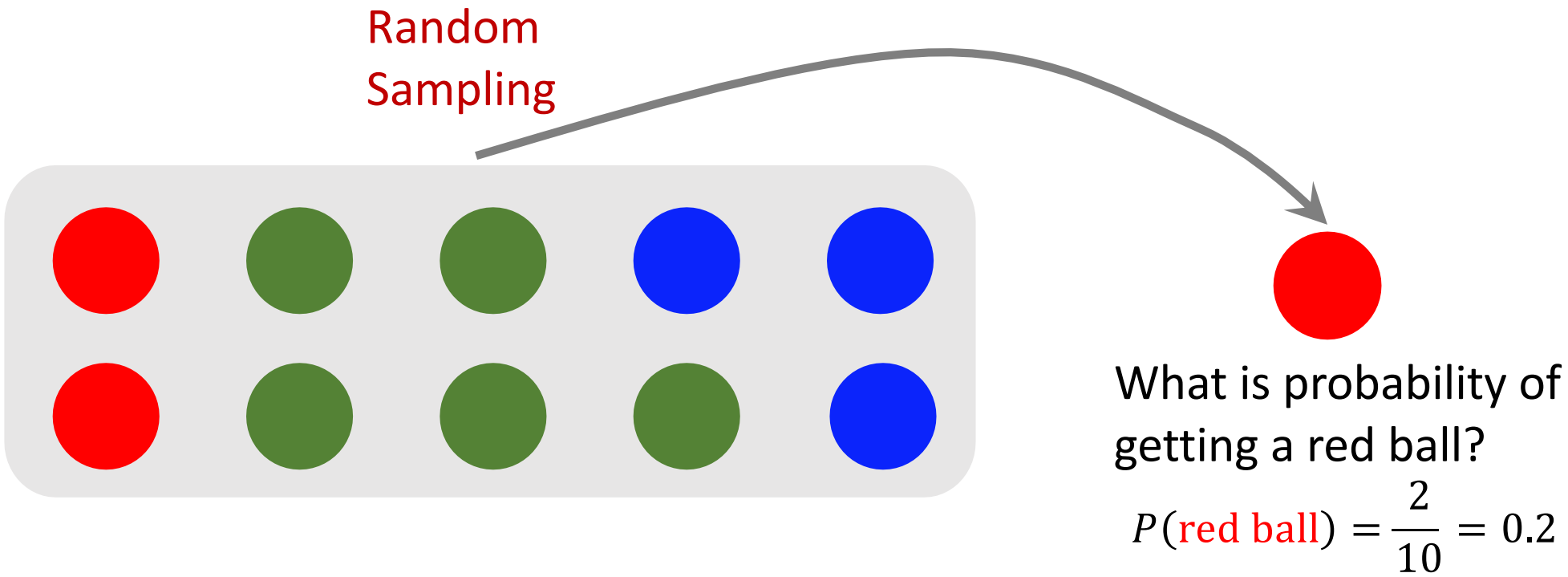
where $|\boldsymbol{\Sigma}|$ is the determinant of the covariance matrix.

- **Standard Bivariate Normal Distribution** $\mathcal{N}(\mathbf{0}, \mathbf{I})$
  - $d = 2$
  - $\boldsymbol{\mu} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$
  - $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$



Random Sampling

- There are 10 balls in the bin: 2 are red, 5 are green, and 3 are blue.



What is probability of getting a red ball?

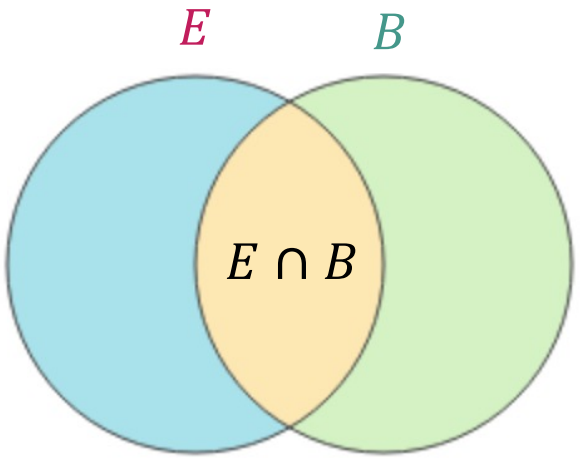
$$P(\text{red ball}) = \frac{2}{10} = 0.2$$

Conditional Probability

- The probability of E given that (aka conditioned on) event B already happened:

$$P(E|B) = \frac{P(E \cap B)}{P(B)}$$

Probability that E occurs given that B has already occurred



$P(E)$

$P(B)$

$P(E \cap B) = P(E \text{ and } B)$

Chain Rule of Probability

- The **chain rule of probability** (also called the **general product rule**) lets you express the joint probability of multiple events by breaking it down into a sequence of conditional probabilities.
- For any two events A and AB: $P(A \cap B) = P(A) \cdot P(B|A)$
- For any n events $A_1, A_2, \ldots, A_n$, the joint probability is:

$$P(A_1 \cap A_2 \cap \cdots \cap A_n) = P(A_1) \cdot P(A_2|A_1) \cdot P(A_1) \cdot P(A_3|A_1 \cap A_2) \cdot \cdots \cdot P(A_n|A_1 \cap A_2 \cap \cdots \cap A_{n-1})$$

or

$$P(A_1, \ldots, A_n) = \prod_{i=1}^{n} P(A_i|all\ previous\ events)$$

- It always holds, even if events are dependent or independent. It's how you multiply probabilities step-by-step using conditionals.

Bayes' Theorem

Thomas Bayes
(1701-1761)

Prior probability

Likelihood

$$P(H|E) = \frac{P(H) \cdot P(E|H)}{P(E)}$$

Posterior probability

Marginal likelihood
/model evidence

Theorem of Total Probability

- Let $y_1, y_2, \ldots, y_N$ be a set of mutually exclusive events (i.e. $y_i \cap y_j = 0$) and event X is the union of N mutually exclusive events, then

$$P(x) = \sum_{i=1}^{N} P(x|y_i)P(y_i)$$

- We can also represent $p(y|x)$ as

$$P(y|x) = \frac{P(x|y)\,P(y)}{\sum_{i=1}^{N} P(x|y_i)P(y_i)}$$

Join, Marginal and Conditional Probability

- **Joint Probability** (of X and Y)

$$P(x, y)$$

- **Conditional Probability** (of X conditioned on Y)

$$P(x|y) = \frac{P(x, y)}{P(y)}$$

- **Marginal Probability** (of Y)

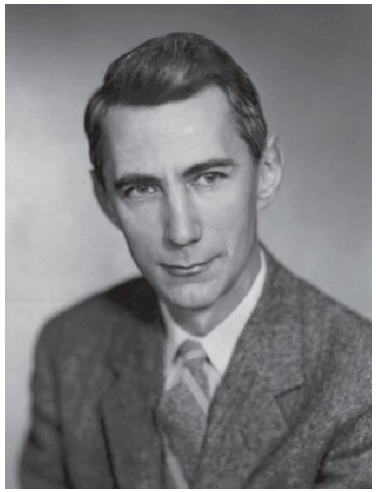
$$P(y) = \int_{x \in X} P(x, y)\, dx$$

Information Theory (1948): Self-Information

- **Observing unlikely events is more informative than likely ones.**
- **Example:** "Sun rose today" is uninformative; "solar eclipse today" is informative (surprising). Quantify as "surprise level."
- **Requirements:** Low info for likely/certain events; high for unlikely; additive for independents (e.g., two heads = twice one head).
- **Definition:** To satisfy all requirements, Claude Shannon define the **self-information** of an event x as

$$I(x) = \log\left(\frac{1}{P(x)}\right) = -\log P(x)$$

where log refers to the natural logarithm and $I(x) \geq 0$.



Claude Shannon
(1916-2001)

Entropy: Average of Information

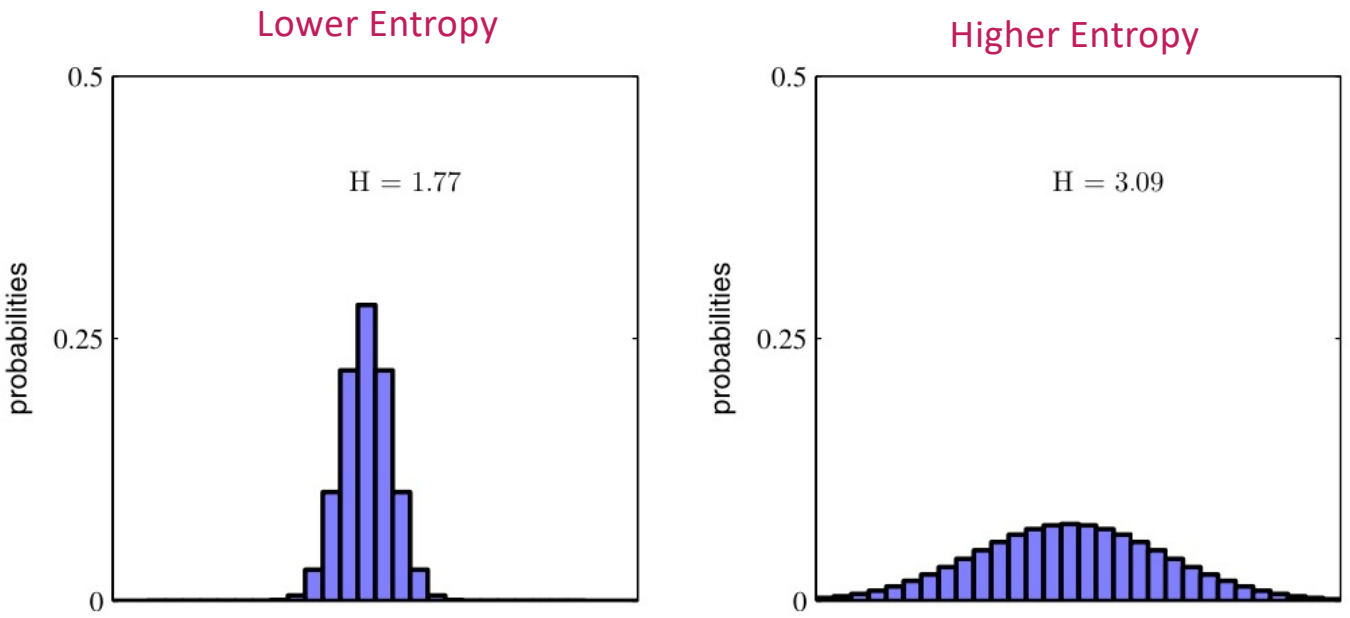
- **Entropy** $H(P)$ quantifies the uncertainty or average amount of information (self-information) in a probability distribution $P(x)$.

$$H(P) = -\mathbb{E}_{x\sim P}(\log P(x)), \qquad \text{where } H(P) \geq 0$$

  - **Interpretation:** Measures average bits (with base-2 log) needed to encode symbols from $P(x)$, setting the lower bound for data compression.
- **Discrete Case:** $H(P) = -\sum_x P(x) \log P(x)$
- **Continuous Case (Differential Entropy):** $H(P) = -\int_x P(x) \log P(x)\, dx$
- **Properties:** Zero for deterministic distributions (no uncertainty); maximum for uniform distributions (highest uncertainty).

Entropy Example

- Histograms of two discrete probability distributions over 30 bins
- The entropy of the broader distribution (on the right) is higher



Advanced Math for Deep Learning (Optional)

- Kullback-Leibler Divergence
- Maximum Likelihood Estimation (MLE)

Kullback-Leibler Divergence (1951)

KL-Divergence (D_{KL}) is a measure of how one probability distribution P is different from a **reference** probability distribution Q.

$$D_{KL}(P||Q) = -\mathbb{E}_{x\sim P}\left[\log\frac{P(x)}{Q(x)}\right] = -\mathbb{E}_{x\sim P}[\log Q(x)] + \mathbb{E}_{x\sim P}[\log P(x)]$$

This can be rewritten as:

$$D_{KL}(P||Q) = H(P,Q) - H(P)$$

Cross Entropy Entropy

Key properties:

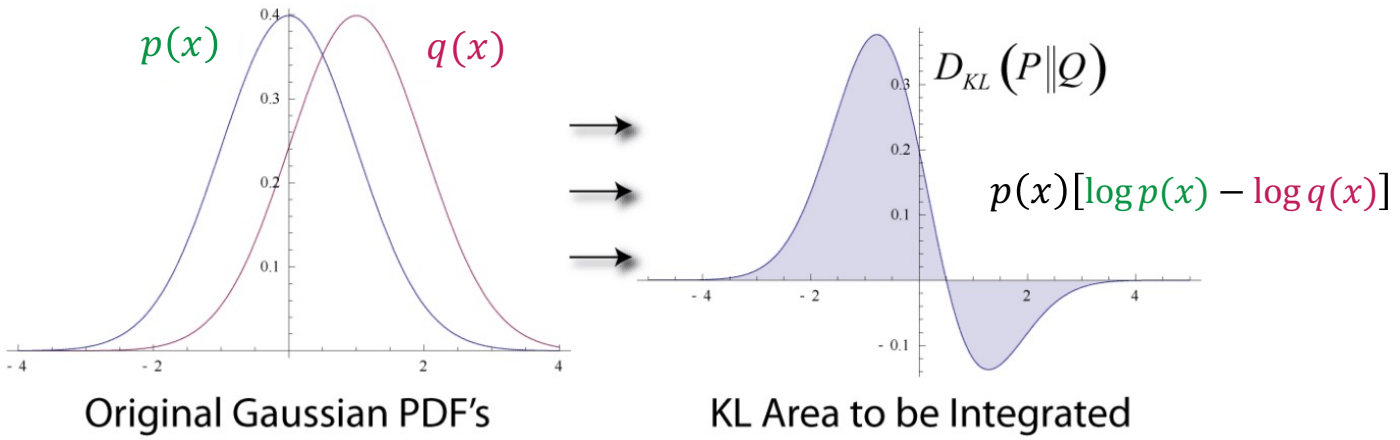
- Always non-negative: $D_{KL}(p||q) \geq 0$
- Not a true distance (asymmetric): $D_{KL}(P||Q) \neq D_{KL}(Q||P)$
- Smaller D_{KL} means more similar of two distributions

Dr. Solomon Kullback

Dr. Richard A. Leibler

KL-Divergence: Visual Example

$$D_{KL}(p||q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx = \int_{-\infty}^{\infty} p(x)[\log p(x) - \log q(x)]dx$$

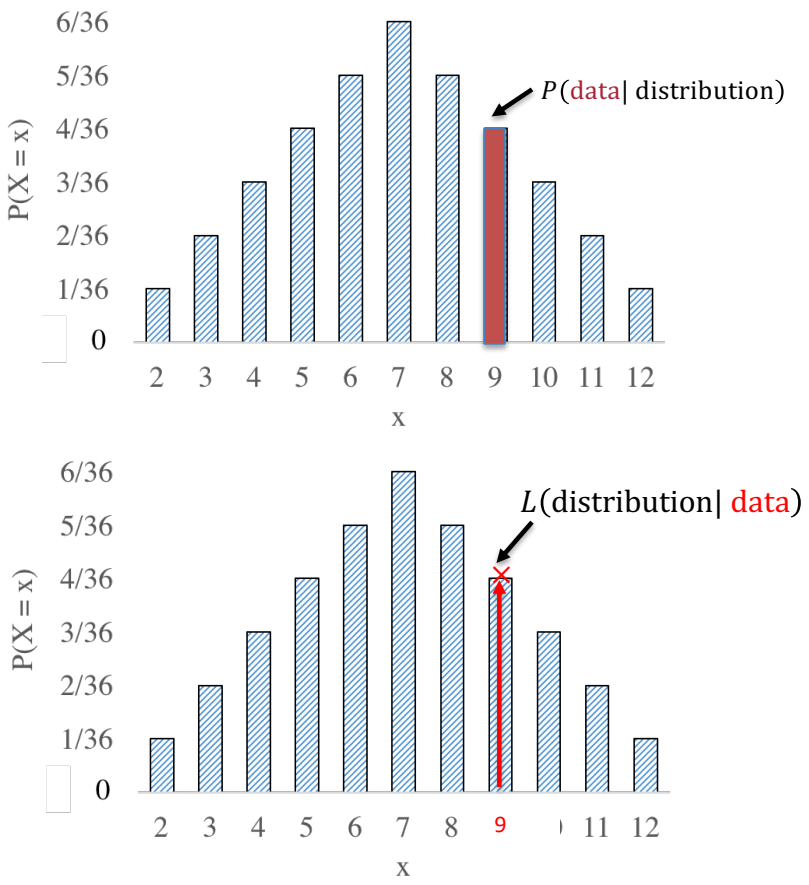


For each point $p(x)$ on the x-axis, we compute $[\log p(x) - \log q(x)]$ and multiply the result by $p(x)$. We then plot the resulting y-value in the right-hand plot. This is how we get to the curve given in the right-hand plot. The KL divergence is now defined as the *area under the graph*, which is shaded.

https://alpopkes.com/posts/machine_learning/kl_divergence/

Probability vs Likelihood

- **Probability**: Predicts data given fixed parameters.
  - Function of data; parameters held constant.
  - Normalized: sums/integrates to 1 over all possible data.
  - Example: $P(\text{head}|p = 0.5) = 0.5$.
- **Likelihood**: Evaluates parameters given fixed observed data.
  - Function of parameters; data held constant.
  - Not normalized—values don't sum to 1 over parameters.
  - Used in maximum likelihood estimation (MLE) to find best-fitting parameters.
  - Example: After observing 7 heads in 10 tosses, $L(p|data)$ peaks near $p = 0.7$.



Core idea: Same mathematical expression $P(\text{data}|\theta)$, but different perspectives—probability forecasts outcomes; likelihood assesses model fit.

The Argmin and Argmax Operators

- Let X denote a set. We define the arg min and arg max operators as follows:

$$\arg\min_{x \in X} f(x) = \min_{x \in X} f(x) = \left\{ x | f(x) = \min_{x' \in X} f(x') \right\}$$

$$\arg\max_{x \in X} f(x) = \max_{x \in X} f(x) = \left\{ x | f(x) = \max_{x' \in X} f(x') \right\}$$

- In words:
  - $\arg\min_{x \in X}$ = the set of points (argument(s)) that achieve the smallest value of f
  - $\arg\max_{x \in X}$ = the set of points that achieve the largest value of f
- Examples:
  - $\arg\min_{x \in \mathbb{R}} x^2 = 0$
  - $\arg\max_{x \in [0,4\pi]} \cos(x) = \{0, 2\pi, 4\pi\}$

Maximum Likelihood Estimation (MLE)

- **MLE** is a statistical method for **estimating parameters θ of a probability distribution** p_θ by maximizing the likelihood function, making the observed data most probable.
- For a given dataset $\{x^{(1)}, x^{(2)}, \dots, x^{(N)}\}$ from a real distribution $p_{data}(x)$
  - **Likelihood Function**: $L(\theta) = p_\theta(data) = \prod_i p_\theta(x^{(i)})$
  - **Log-Likelihood** (easier to maximize): $L(\theta) = \sum_i \log p_\theta(x^{(i)})$
  - **MLE** estimate θ:

$$\theta^* = \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x^{(i)})$$

Maximum Likelihood Estimation

- Data Samples: $\{x^{(1)}, x^{(2)}, \dots, x^{(N)}\}$ from $p_{data}(x)$

$$\theta^* = \arg\max_\theta \prod_{i=1}^{N} p_\theta(x^{(i)}) = \arg\max_\theta \log\left(\prod_{i=1}^{N} p_\theta(x^{(i)})\right) = \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x^{(i)})$$

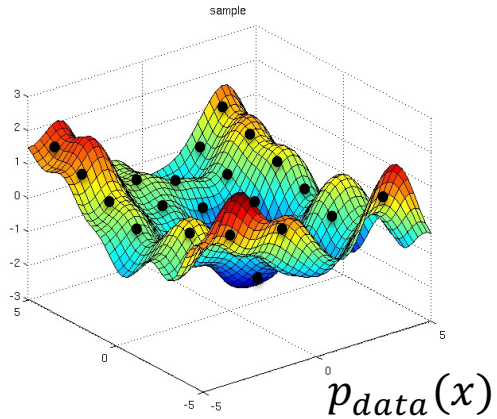
Maximum Likelihood **Maximum Log Likelihood**

$$\approx \arg\max_\theta \mathbb{E}_{x\sim p_{data}(x)}[\log p_\theta(x)] = \arg\max_\theta \int_x p_{data}(x) \log p_\theta(x)\, dx$$

$$= \arg\max_\theta \int_x p_{data}(x) \log p_\theta(x)\, dx - \int_x p_{data}(x) \log p_{data}(x)\, dx$$

(not related to θ)

$$= \arg\max_\theta \int_x p_{data}(x) \log \frac{p_\theta(x)}{p_{data}(x)}\, dx = \arg\min_\theta D_{KL}(p_{data}(x)||p_\theta(x))$$



Maximizing Likelihood (or Log Likelihood) = **Minimizing** KL Divergence

Homework

- Try to form your group project team with **5 members**
- Start to discuss the project direction
- Performing research on the selected topics
- Send your **group member list** to instructor at eelmpo@cityu.edu.hk **on or before the Friday of week 3 (Jan 31, 2026)**