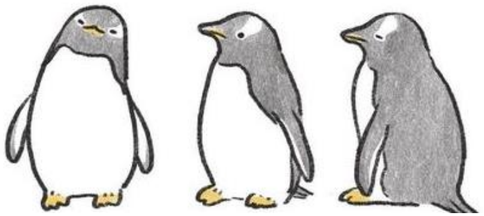


# 實驗課程研發工作坊與教師專業增能研習實施計畫

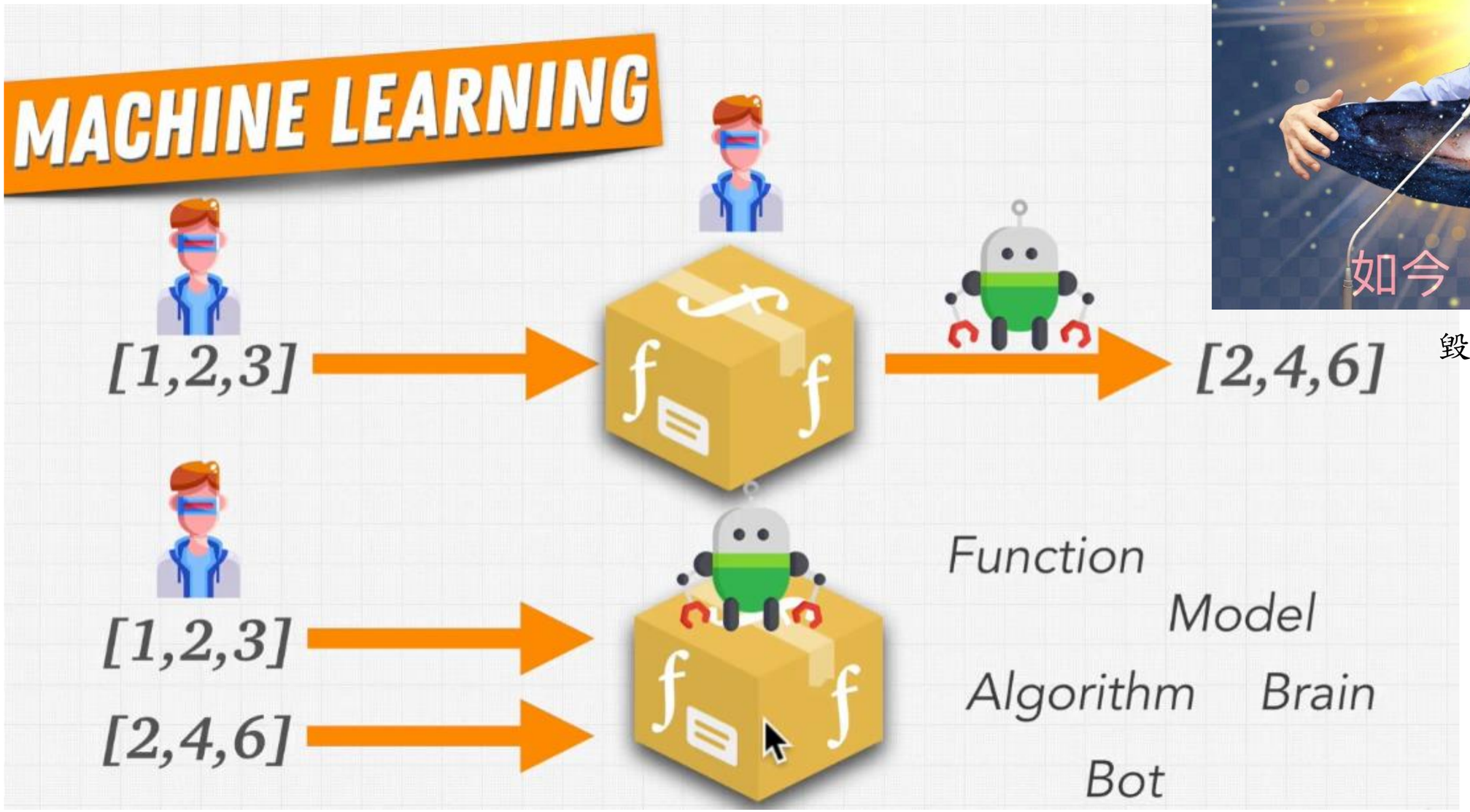
國立成功大學工程科學系

感知學習與人機合作實驗室


# Python究竟可以拿來幹嘛呢?

- 在前三堂的課程，都是以low level去看Python可以完成的事。
- 而Python本身是一個high level的程式語言，可以完成架網站、視覺化、爬蟲...甚至是當今最火熱的機器學習
- 因此，我們就來談【機器學習】這一個課題~
- 以下內容為一些基於python小小專案，採用分享的角度去論述，不會去描述過多的程式碼，會比較著重於【機器學習】裡面的一些知識點。

# 甚麼是機器學習



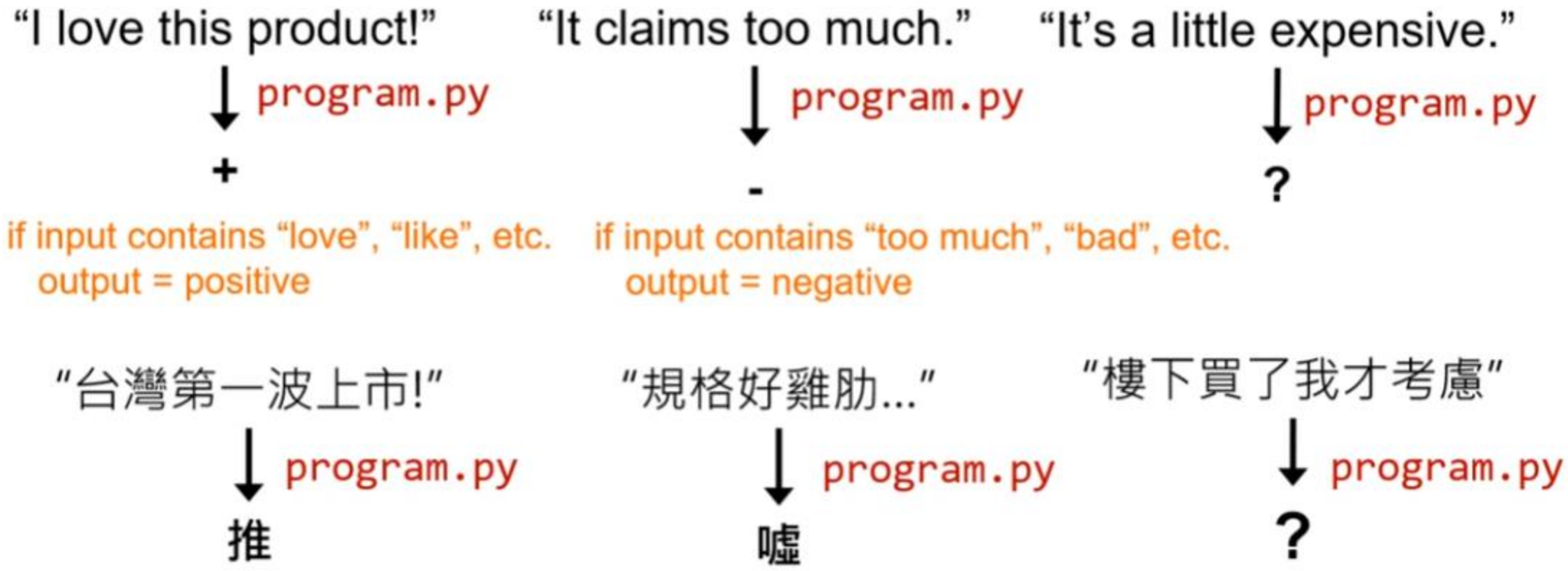


毀滅地球，征服宇宙(?

# 甚麼是機器學習

## 4 Program for Solving Tasks

- Task: predicting positive or negative given a product review

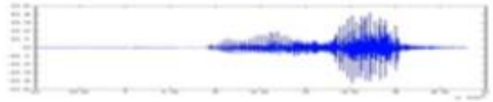



Some tasks are complex, and we don't know how to write a program to solve them.

來源：https://www.youtube.com/watch?v=Yd7R3JIFvnY

# 甚麼是機器學習

## 6 Learning ≈ Looking for a Function

- Speech Recognition $f($  $)=$ “你好”
- Handwritten Recognition $f($  $)=$ “2”
- Weather forecast $f($  Thursday $)=$ “  Saturday”
- Play video games $f($  $)=$ “move left”

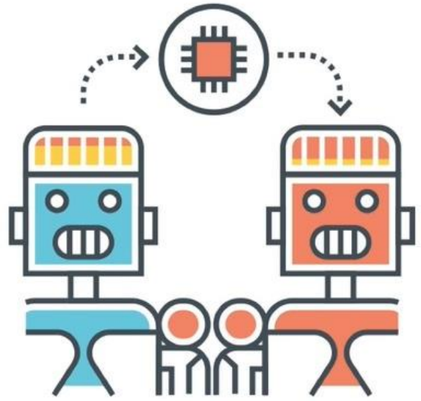
來源：https://www.youtube.com/watch?v=Yd7R3JIFvnY

# 甚麼是機器學習

## Main types of machine learning

Supervised Learning

Unsupervised Learning

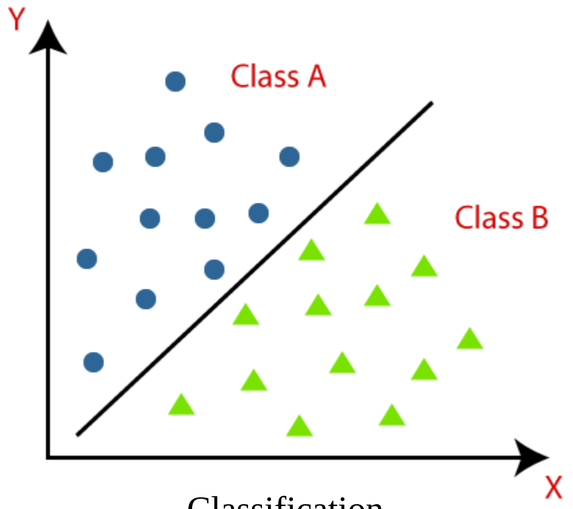


Transfer Learning


Reinforcement Learning

# 監督式學習（Supervised Learning）

- 監督式學習需要標註過的數據（有輸入和對應的輸出），模型學習數據中輸入和輸出之間的映射關係。
- 讓模型能夠根據新的輸入預測對應的輸出。
- 類型：
  - 分類（Classification）：預測類別，例如判斷圖片中的物體是貓還是狗。
  - 回歸（Regression）：預測連續值，例如房價、溫度等。
- 應用： 電郵垃圾郵件分類 - 房價預測 - 醫學影像診斷




Classification


Regressionin

# 監督式學習（Supervised Learning）

❖ 範例：透過結構方程模型評估小學生對於教學中生成式AI應用的態度

## 透過結構方程模型評估小學生對於教學中生成式 AI 應用的態度

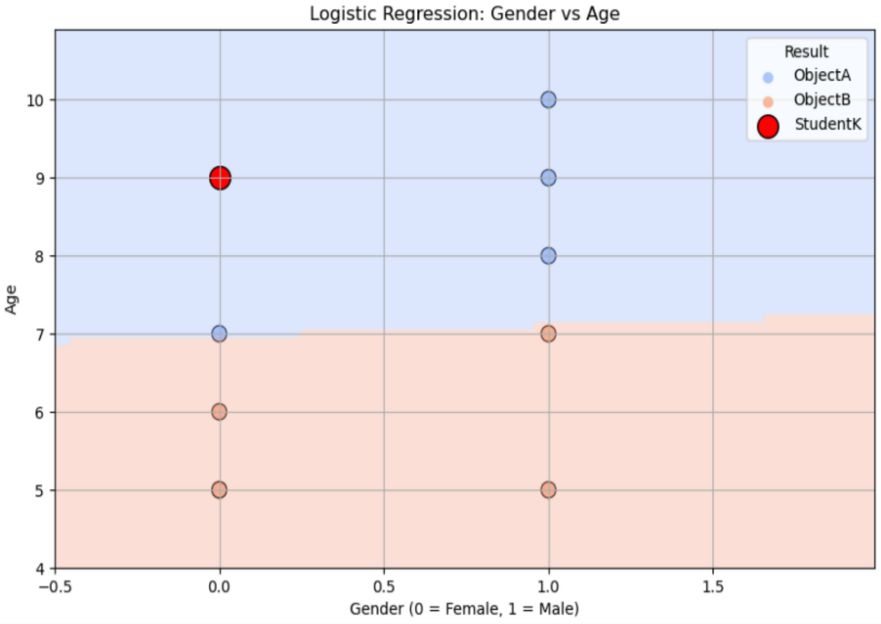
**Assessing educators' attitudes towards the use of generative AI in teaching through structural equation modeling**

### 概要


隨著 AI 迅速滲透到各行業，將 AI 功能整合進現代學習管理系統已成趨勢。這為優化和完善教育方法提供了良機，通過建立持續的反饋循環實現教育實踐的改進。在教育領域，GenAI 的強大語言生成和交互能力為教育者和學生提供個性化學習支持和即時反饋。然而，這些工具的成功實施取決於教育者的接受程度和使用體驗，這受到他們對 GenAI 的態度和 AI 素養的影響。教育環境在引入新技術時，通常會在樂觀支持與懷疑之間擺盪，主因是新技術在現實應用中的不確定性。

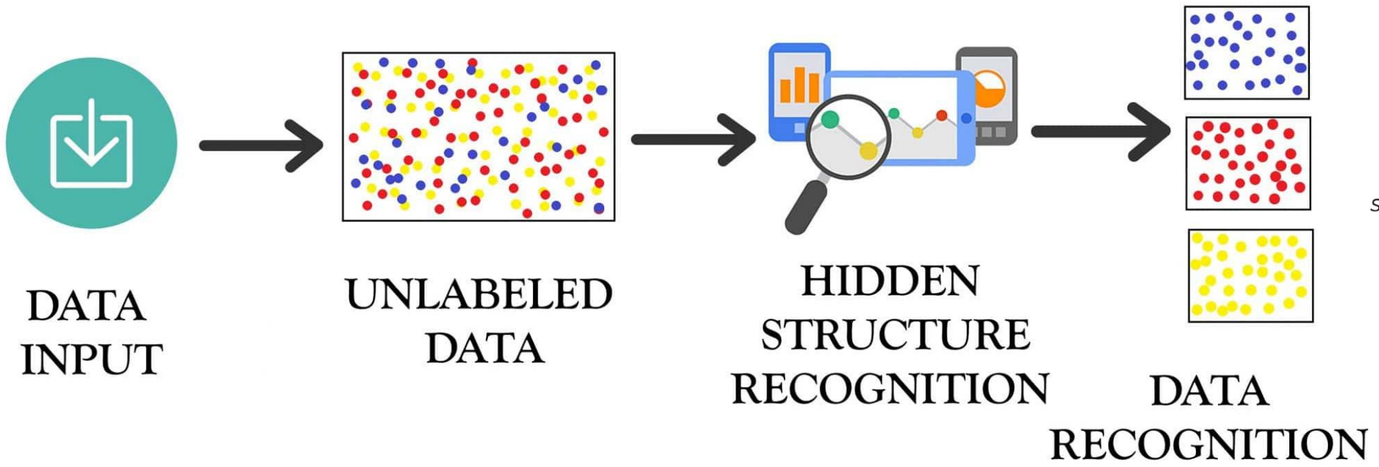
本研究旨在探討 GenAI 工具在教育領域的應用，特別關注小學生對於生成式 AI 工具的接受度、使用體驗和態度，並利用結構方程模型(SEM)進行數據分析。文獻回顧總結 AI 在教育中的應用和潛在益處及 AI 應用面臨倫理問題、教師培訓不足和技術基礎設施等挑戰。並建議教育機構和政策制定者在推進技術應用時，考慮教育者的態度和 AI 素養對其使用效果的影響。研究結果顯示，小學生對生成式 AI 的態度顯著影響其使用頻率和學習效果。本研究為未來教學設計和教育政策提供了重要參考。

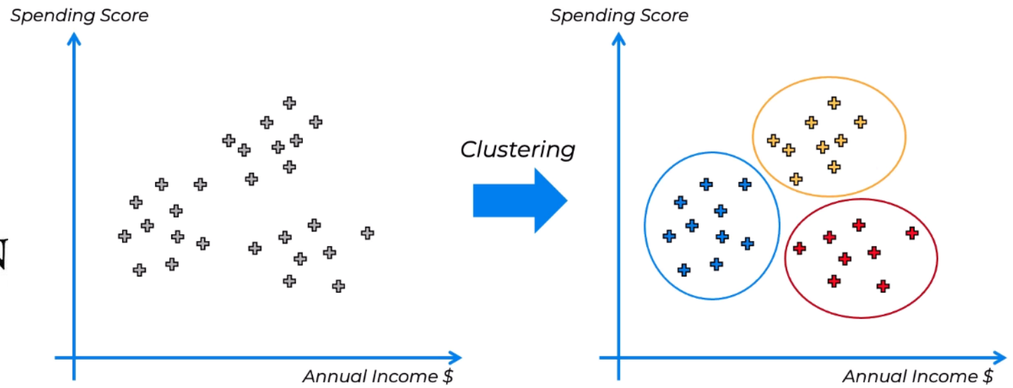

| *Sample* | *Gender* | *Age* | *Result* |
|---|---|---|---|
| *StudentA* | *Female* | *7* | *ObjectA* |
| *StudentB* | *Female* | *5* | *ObjectB* |
| *StudentC* | *Male* | *8* | *ObjectA* |
| *StudentD* | *Female* | *9* | *ObjectA* |
| *StudentE* | *Male* | *5* | *ObjectB* |
| *StudentF* | *Male* | *10* | *ObjectA* |
| *StudentG* | *Male* | *7* | *ObjectB* |
| *StudentH* | *Female* | *6* | *ObjectB* |
| *StudentI* | *Male* | *9* | *ObjectA* |
| *StudentJ* | *Female* | *5* | *ObjectB* |



# 非監督式學習（Unsupervised Learning）

- 非監督式學習處理未標註的數據，目的是讓模型發現數據中的潛在結構或模式。
- 將數據進行分組或降維，找出潛在規律。
- 類型：
  - 聚類（Clustering）：將相似的數據分成同一群組，例如顧客分群。
  - 關聯規則學習（Association Rule Learning）：發現數據中的關聯性，例如購物籃分析。
- 應用：- 市場顧客分群 - 異常檢測（如金融詐欺）- 購物推薦系統

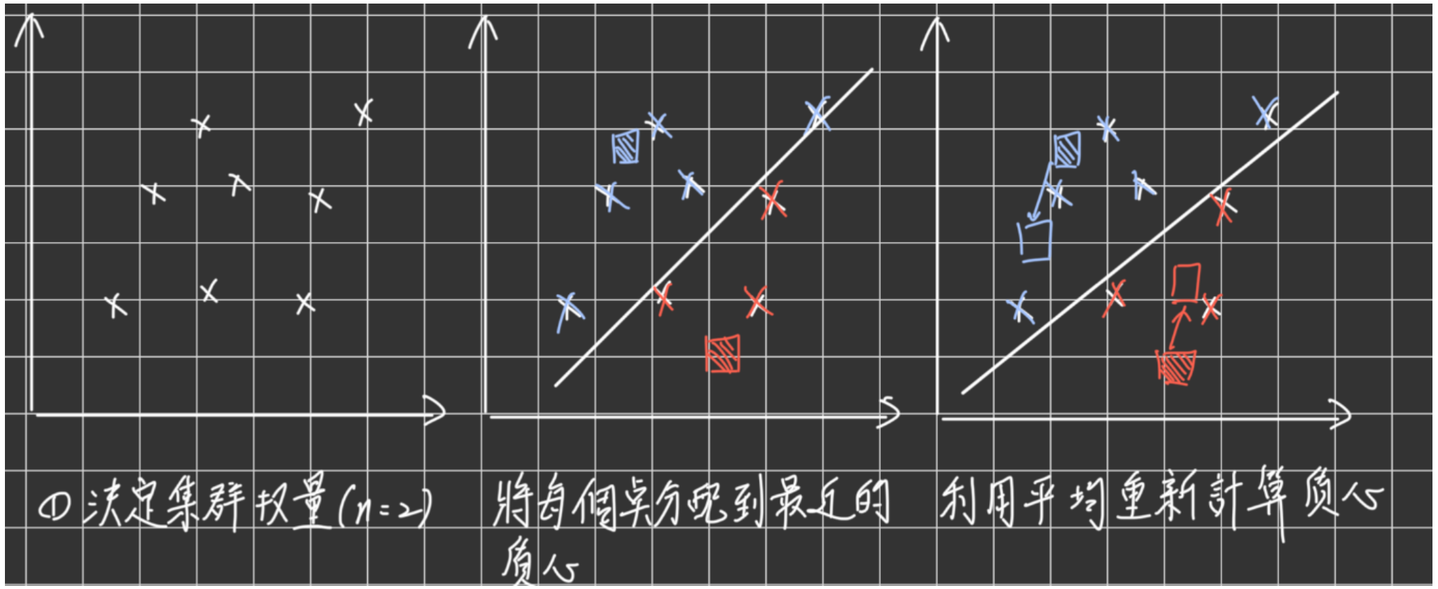


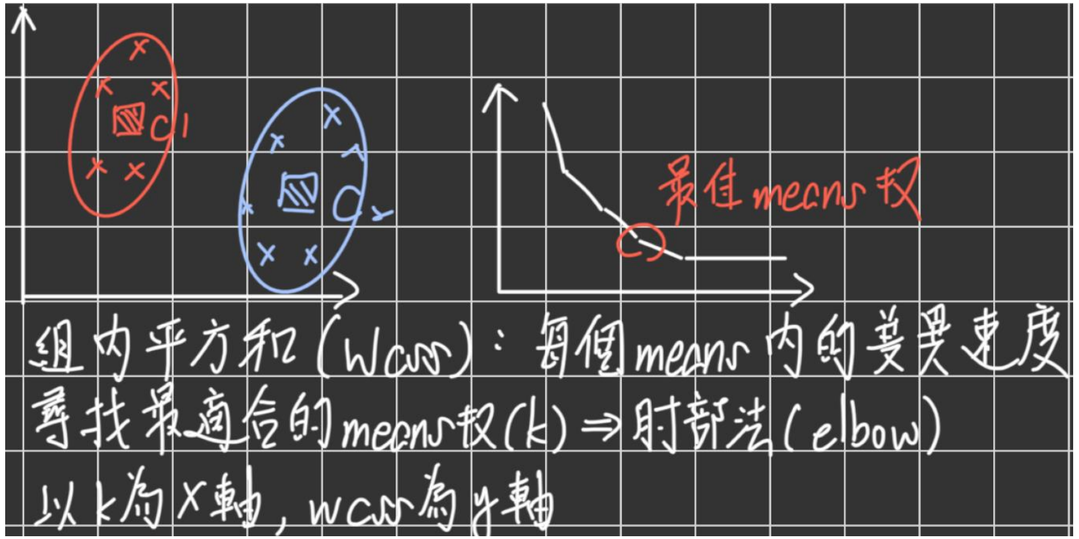



# 非監督式學習（Unsupervised Learning）

【K-means】

- 是一種無監督學習演算法，用於資料的聚類分析。
- 它會將資料分成 $k$ 個群集 (clusters)，每個群集都有自己的中心點 (centroid)。

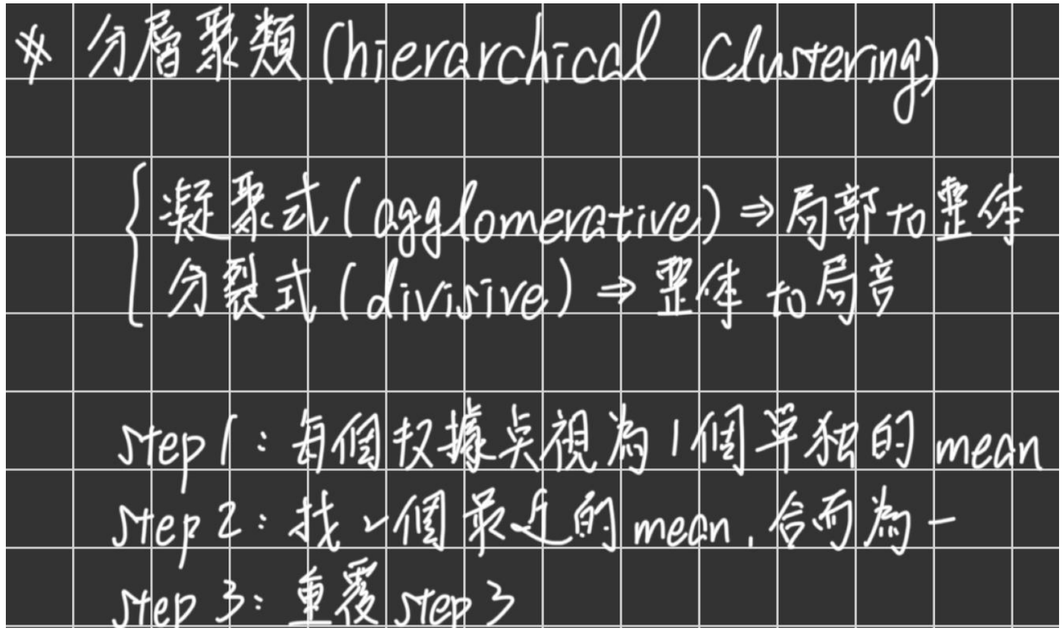


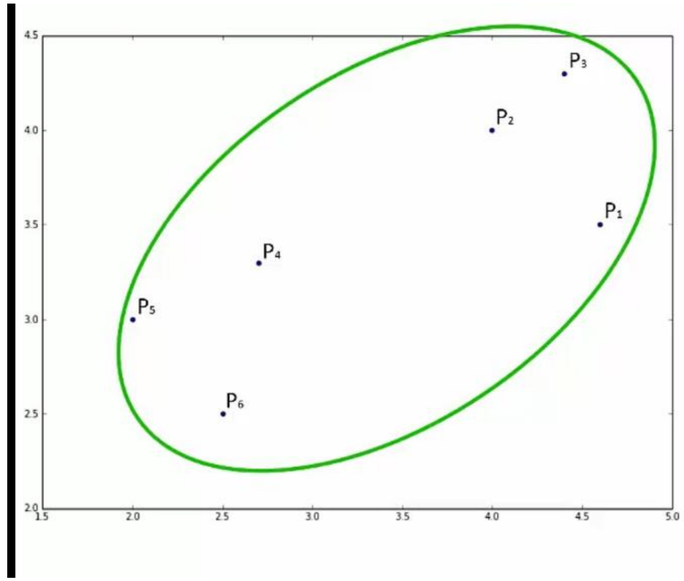



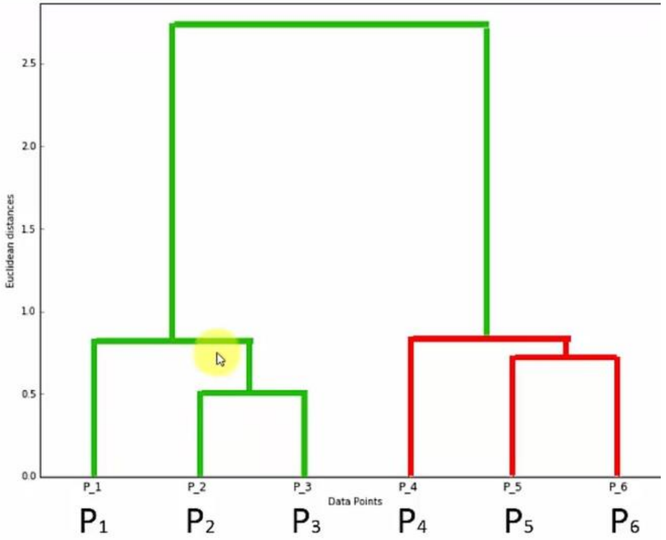
# 非監督式學習（Unsupervised Learning）

【層次聚類(Hierarchical Clustering)】

- Hierarchical Clustering (層次聚類) 是一種無監督學習演算法。
- 來將數據分成多個群集。
- 層次聚類的結果通常以樹狀圖（Dendrogram） 表示，顯示數據的合併過程和層次結構。



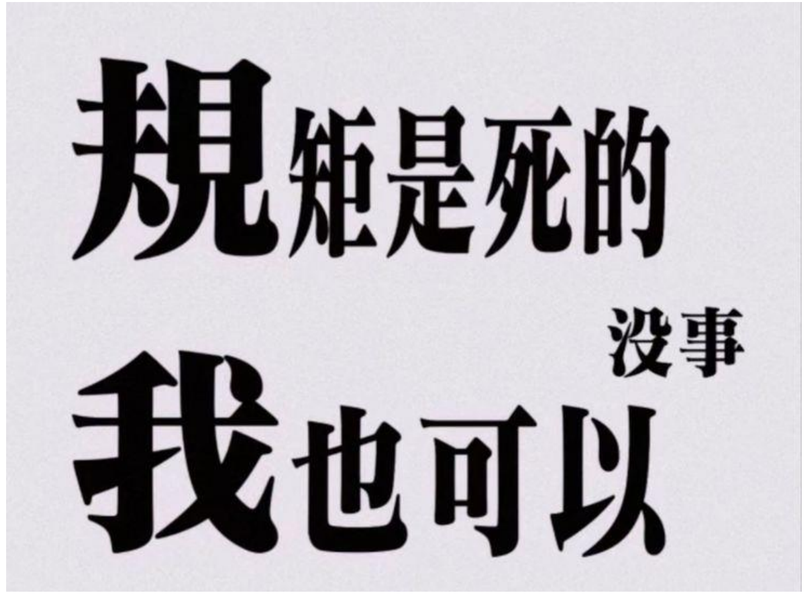



# 非監督式學習（Unsupervised Learning）

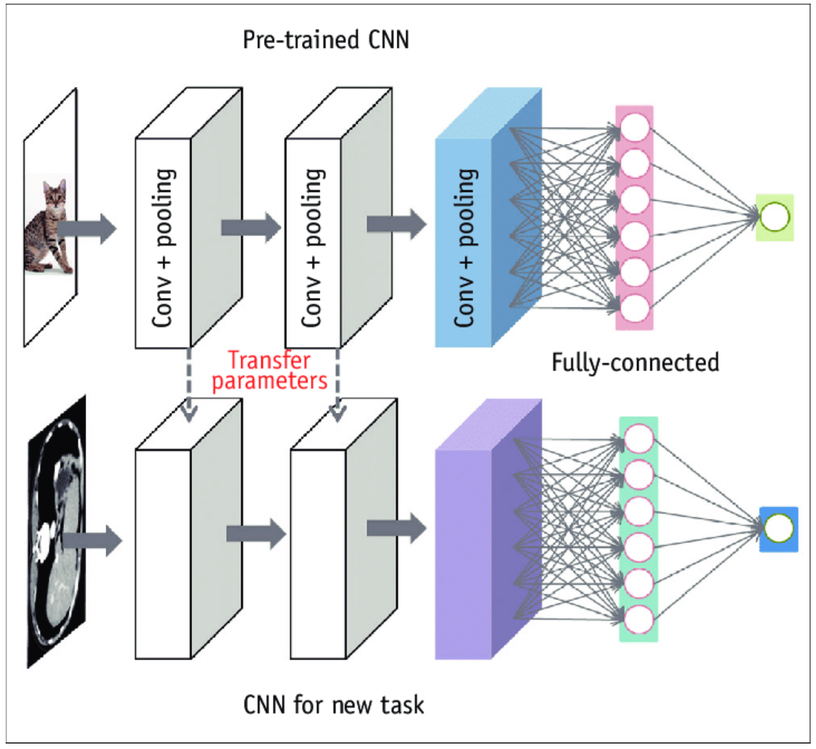
【關聯規則學習(Associated Rules Learning)】

- 啤酒跟尿布的故事
- 用於發現數據集中項目之間的關聯規則。
- 經常用於分析購物籃數據，找出哪些商品經常一起購買。’
- 演算法：Apriori、Eclat




# 遷移學習（Transfer Learning）

- 遷移學習是一種將在一個任務中學到的知識，應用到另一個相關任務上的方法。
- 減少訓練時間，讓模型在小數據集上達到良好效果。
- 應用：
  - 圖像識別：使用訓練好的 ResNet 或 VGG 模型進行新圖像分類。
  - 自然語言處理（NLP）：使用預訓練語言模型如 BERT 或 GPT 進行新任務微調。

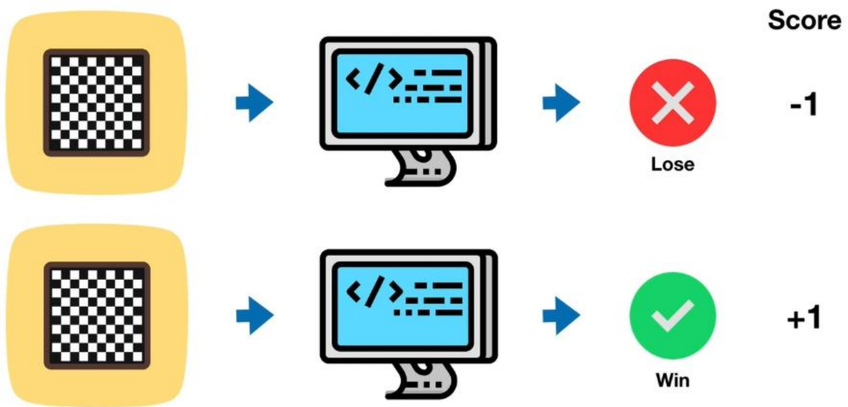


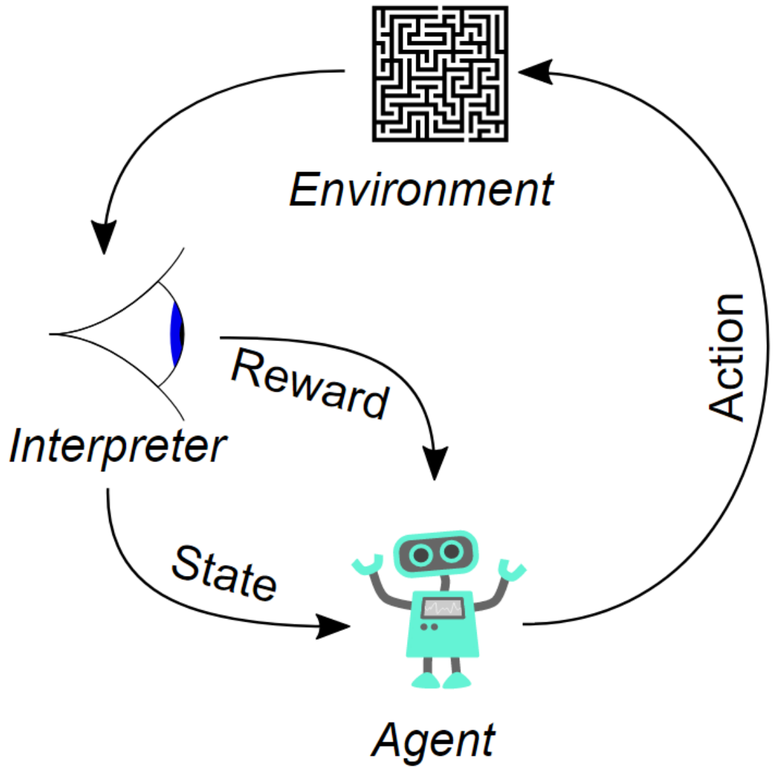





# 強化學習（Reinforcement Learning）

- 強化學習是一種基於試錯法的學習方式，智能體（Agent）與環境互動，透過獎勵和懲罰來學會最佳策略。
- 讓智能體學會如何在不同狀態下選擇動作，以最大化累積獎勵。
- 特點：
  - 學習過程沒有標準答案，而是根據獎勵信號來學習。
  - 適合多步驟決策問題。
- 應用：
  - 遊戲 AI：如 AlphaGo、Dota 2 AI 等。
  - 機器人控制：讓機器人學會走路或抓取物體。
  - 自動駕駛：學會在道路上決策。

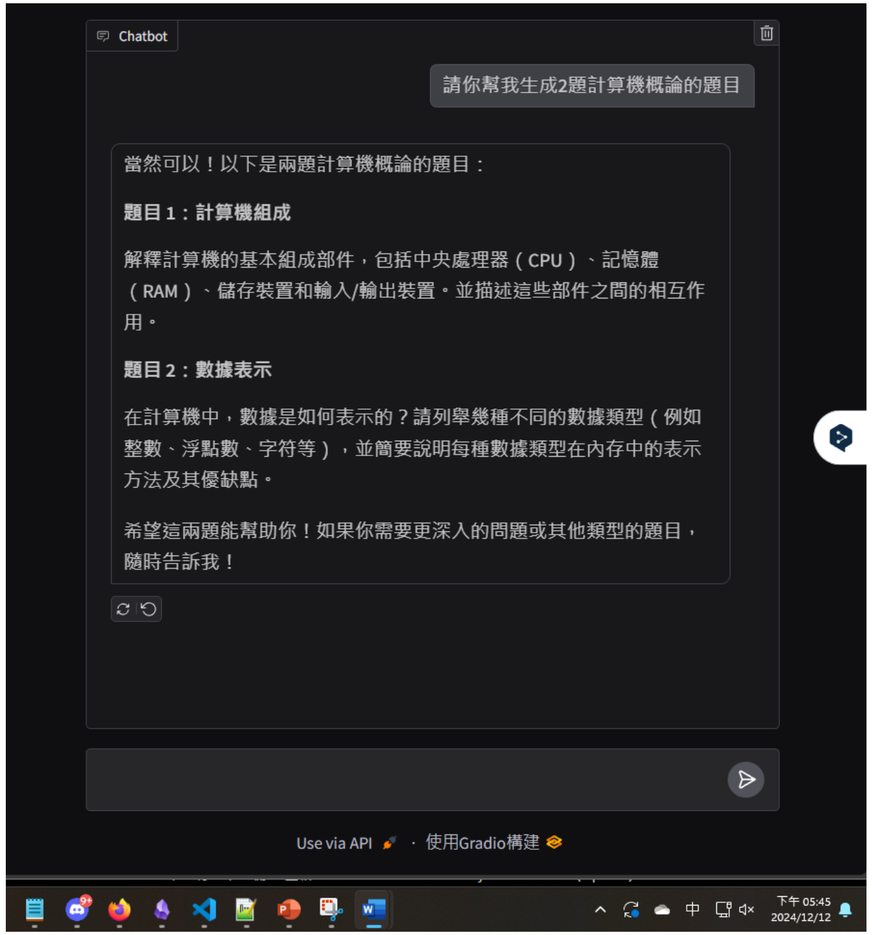


# AI Agent

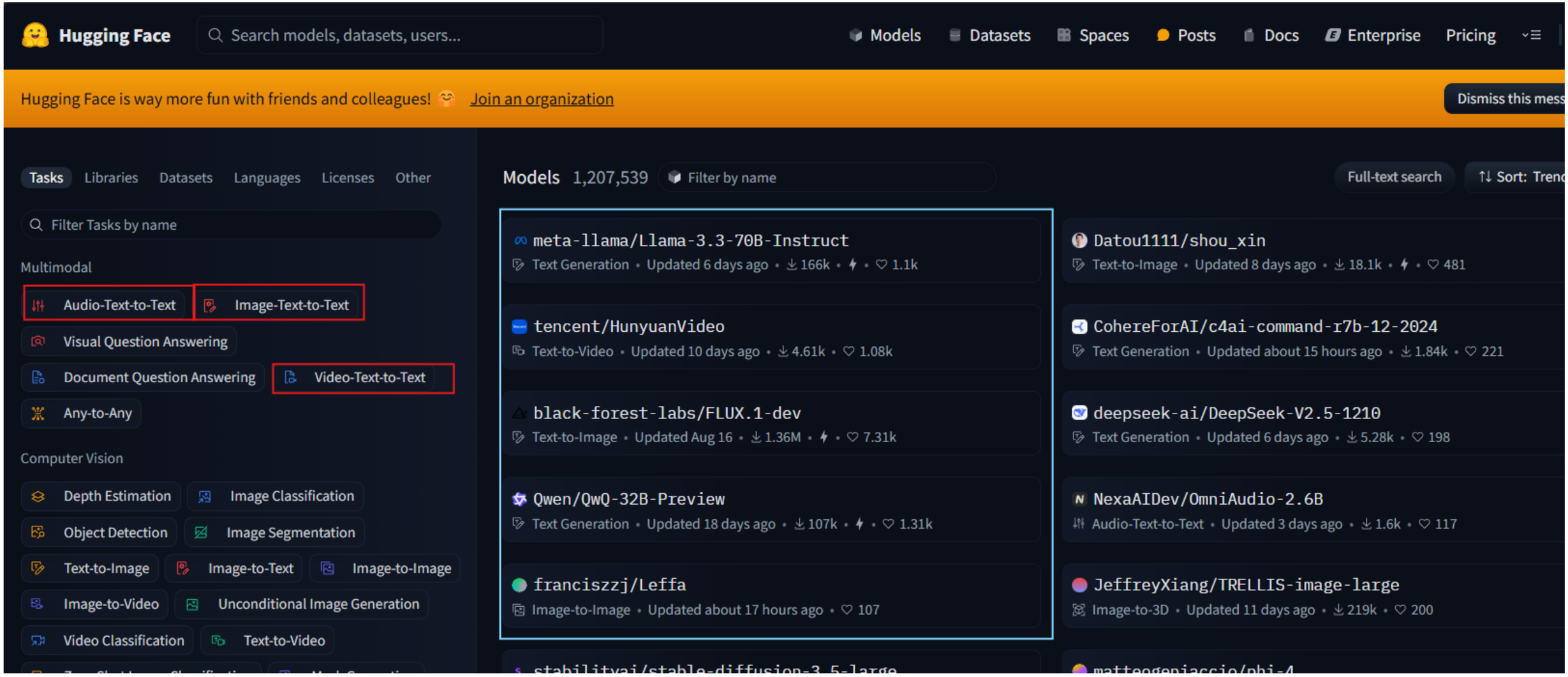
- 目前發展的model，都是基於某個目的而完成。
  - 像是：LLM(大型語言模型)(text to text)、text to image、Image to image、voice to text 、text to voice

dalle 3




Open api + graido

# AI Agent




Hugging face

# AI Agent

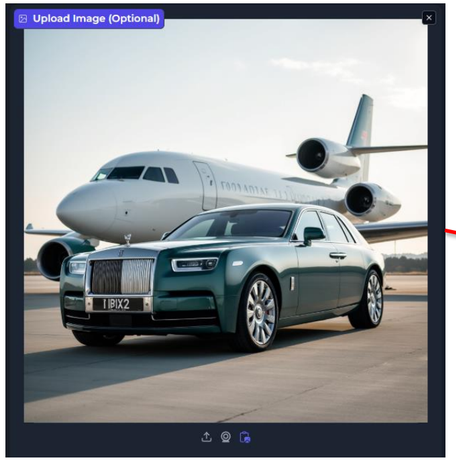
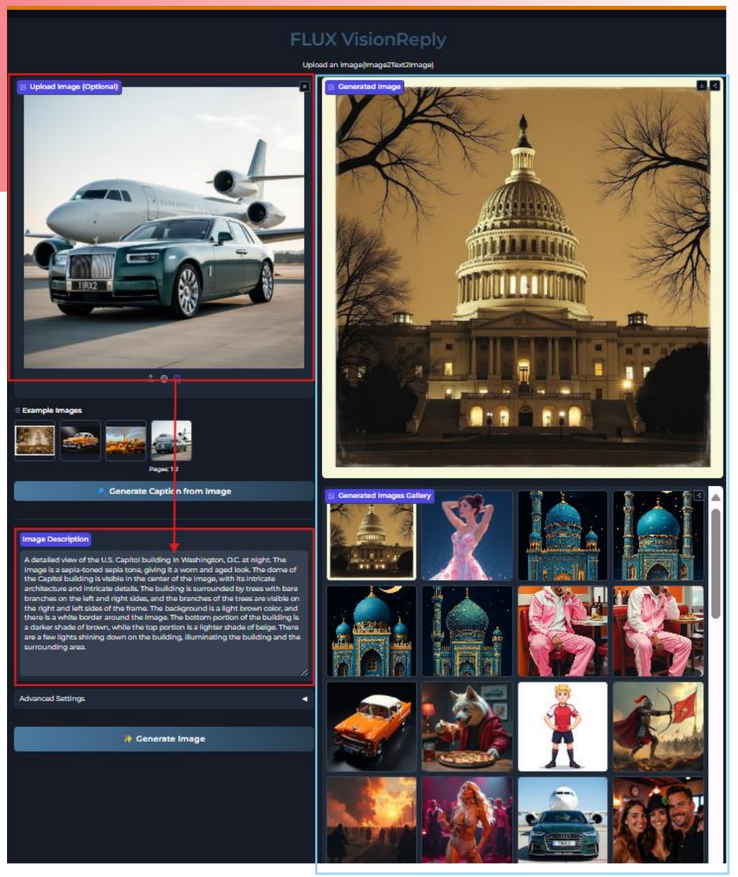



Image to text

**Image Description**

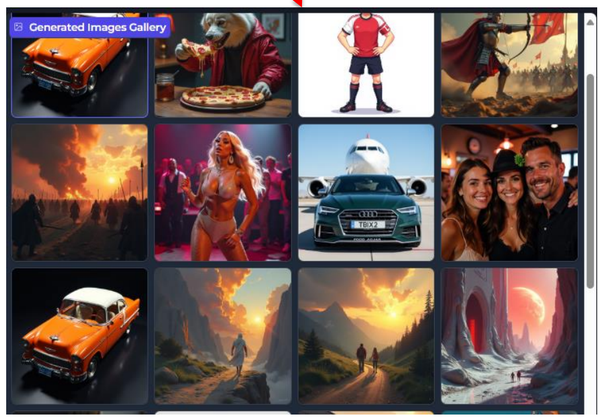
A detailed view of the U.S. Capitol building in Washington, D.C. at night. The image is a sepia-toned sepia tone, giving it a worn and aged look. The dome of the Capitol building is visible in the center of the image, with its intricate architecture and intricate details. The building is surrounded by trees with bare branches on the left and right sides, and the branches of the trees are visible on the right and left sides of the frame. The background is a light brown color, and there is a white border around the image. The bottom portion of the building is a darker shade of brown, while the top portion is a lighter shade of beige. There are a few lights shining down on the building, illuminating the building and the surrounding area.




來源：
https://huggingface.co/spaces/aiqcamp/FLUX-VisionReply

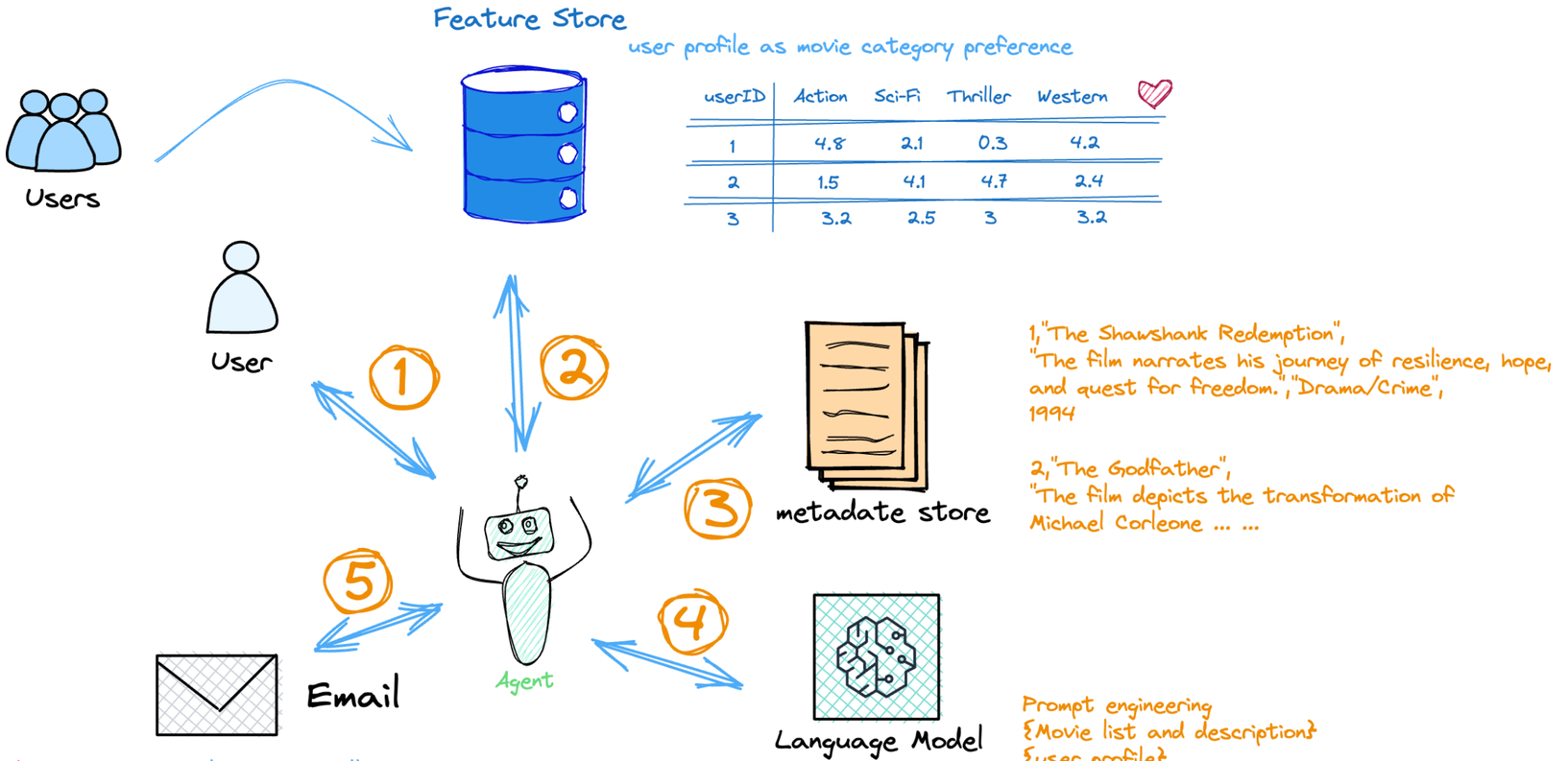

Text to image

other parameters



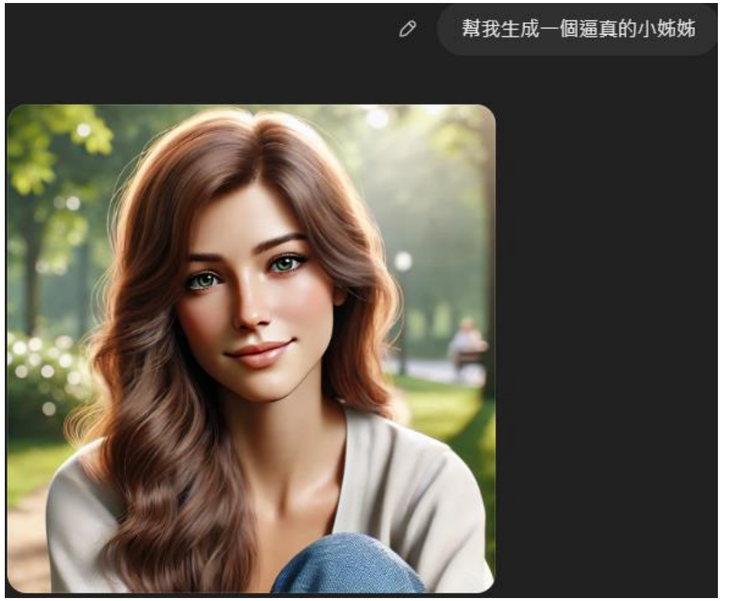
# AI Agent

❖ 那我們使用的chat gpt，她是如何那麼的【全面】呢





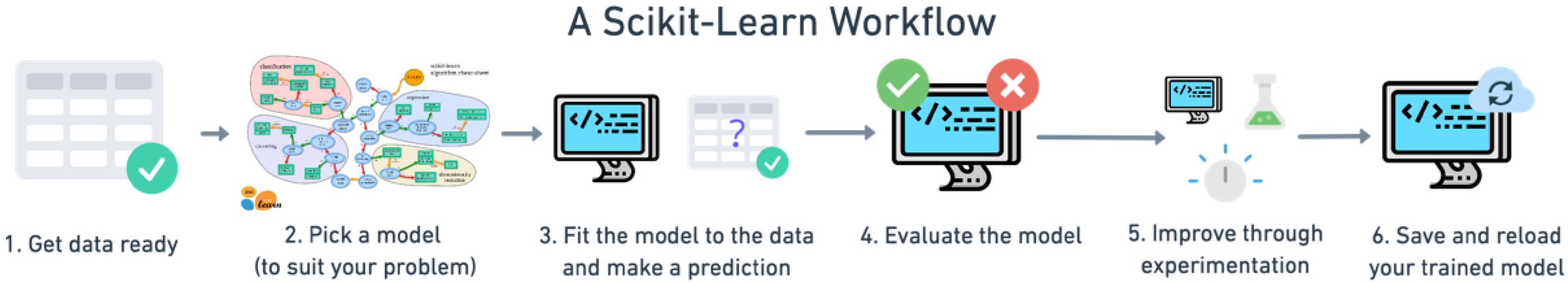
Ask for code




generate a picture

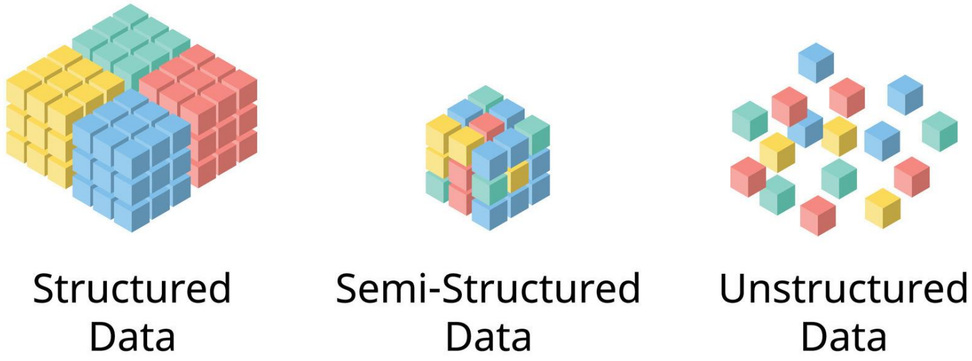
來源：https://aws.amazon.com/what-is/ai-agents/

# Model Workflow

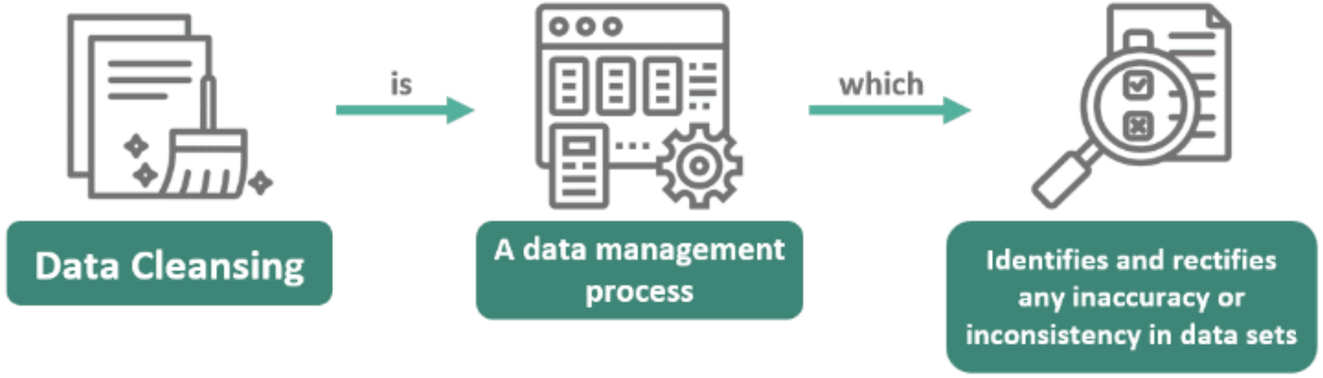


# Data

- 結構化數據（Structured Data）
  - 是高度組織化的資料。
  - 通常存放於關係型資料庫 (如 MySQL、PostgreSQL) 或電子表格 (如 Excel) 中。
  - 資料可以用行和列來清楚定義，並且每列代表一條記錄，每列的每一個欄位都有固定的資料類型 (例如整數、字串、日期等)。
- Semi-structured Data (半結構化資料)
  - 資料有一定的結構，但不如結構化資料那麼規範。
  - 不一定存儲在關係型資料庫中，而是以某種形式的標記語言或層次結構來組織，例如 XML、JSON、YAML 等。
- Unstructured Data (非結構化資料)
  - 通常為自由格式，資料的組織方式不規則。
  - PDF、image、mp3、mp4



# Data cleaning





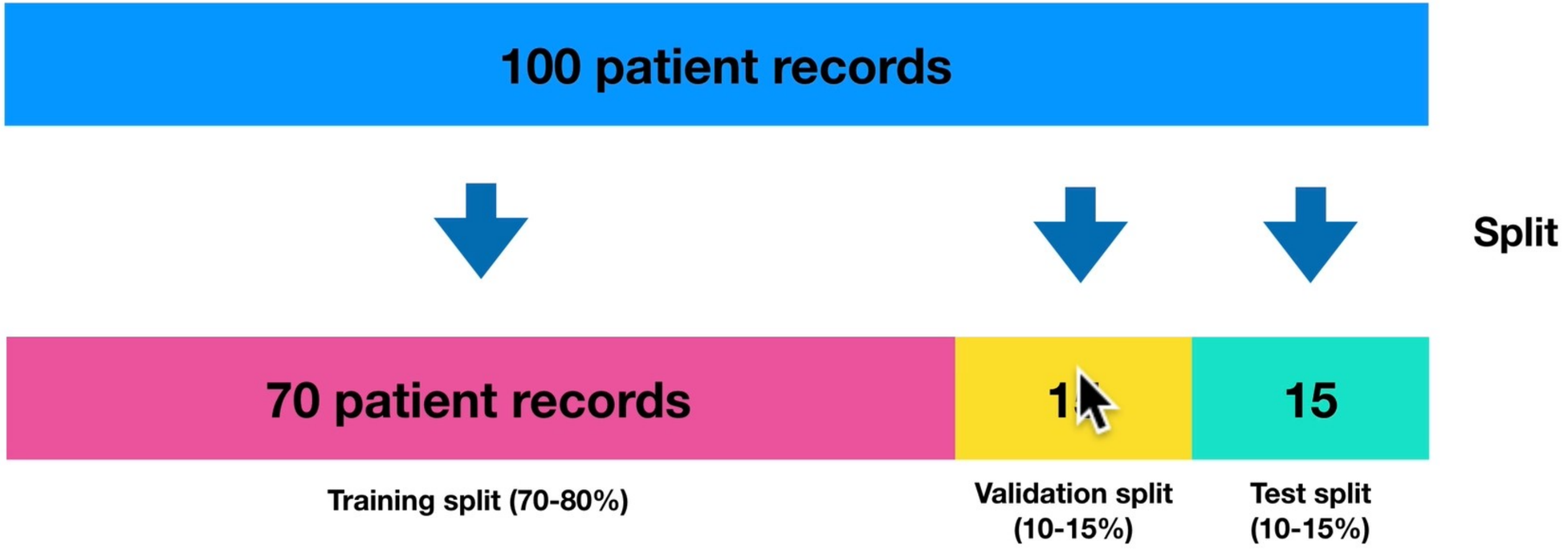
```
heart_disease = pd.read_csv("./data set/heart-disease.csv")
heart_disease.head() # 查看前5行

# 顯示的target，為該筆資料最終是否具有糖尿病。
```

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 63 | 1 | 3 | 145 | 233 | 1 | 0 | 150 | 0 | 2.3 | 0 | 0 | 1 | 1 |
| 1 | 37 | 1 | 2 | 130 | 250 | 0 | 1 | 187 | 0 | 3.5 | 0 | 0 | 2 | 1 |
| 2 | 41 | 0 | 1 | 130 | 204 | 0 | 0 | 172 | 0 | 1.4 | 2 | 0 | 2 | 1 |
| 3 | 56 | 1 | 1 | 120 | 236 | 0 | 1 | 178 | 0 | 0.8 | 2 | 0 | 2 | 1 |
| 4 | 57 | 0 | 0 | 120 | 354 | 0 | 1 | 163 | 1 | 0.6 | 2 | 0 | 2 | 1 |

```
heart_disease.info() # 查看檔案資料
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 3   trestbps  303 non-null    int64  
 4   chol      303 non-null    int64  
 5   fbs       303 non-null    int64  
 6   restecg   303 non-null    int64  
 7   thalach   303 non-null    int64  
 8   exang     303 non-null    int64  
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64  
 11  ca        303 non-null    int64  
 12  thal      303 non-null    int64  
 13  target    303 non-null    int64  
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

# Splitting Data

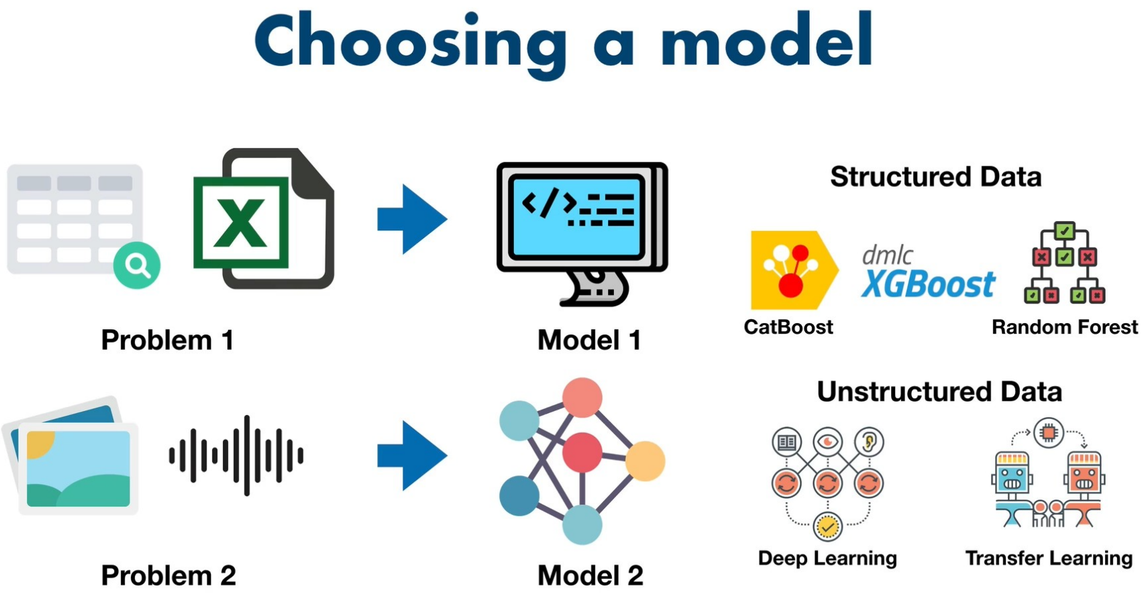


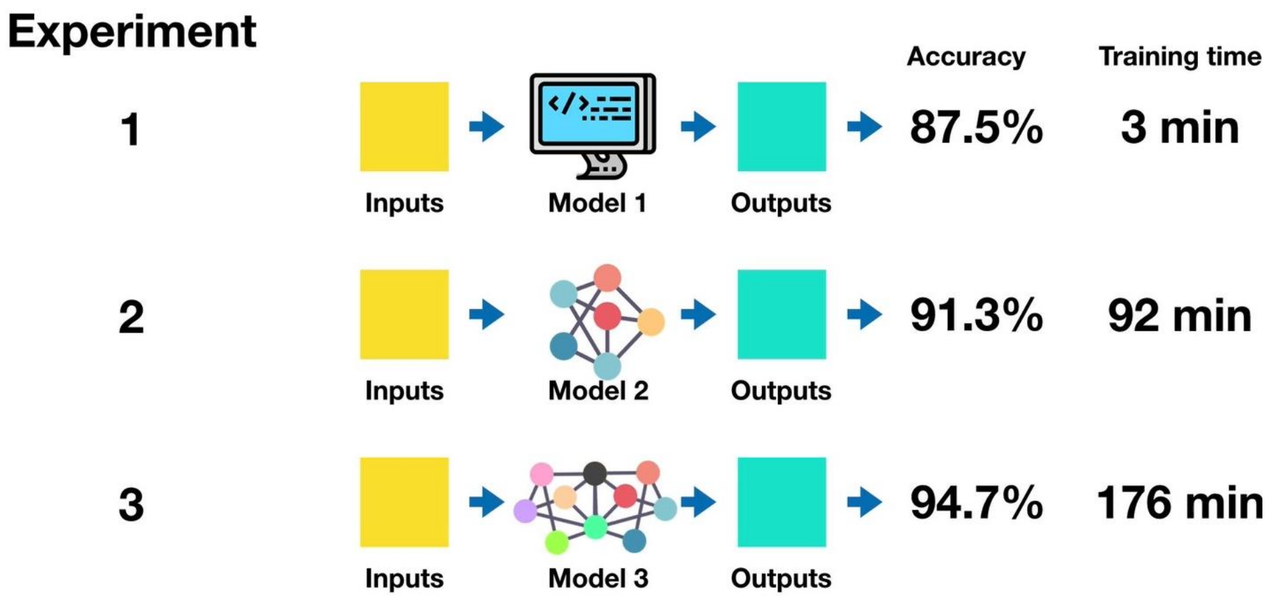

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8, test_size=0.2) # 80% 20%

# View the data shapes
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((242, 13), (61, 13), (242,), (61,))
```

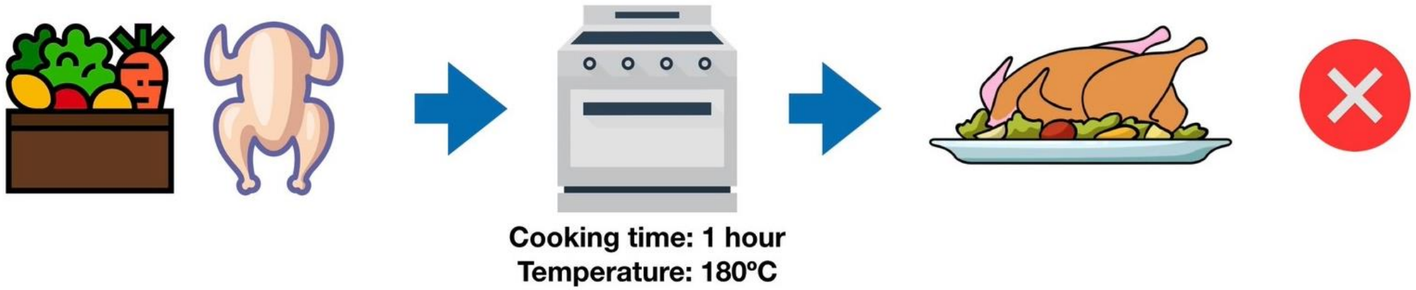
# Modelling

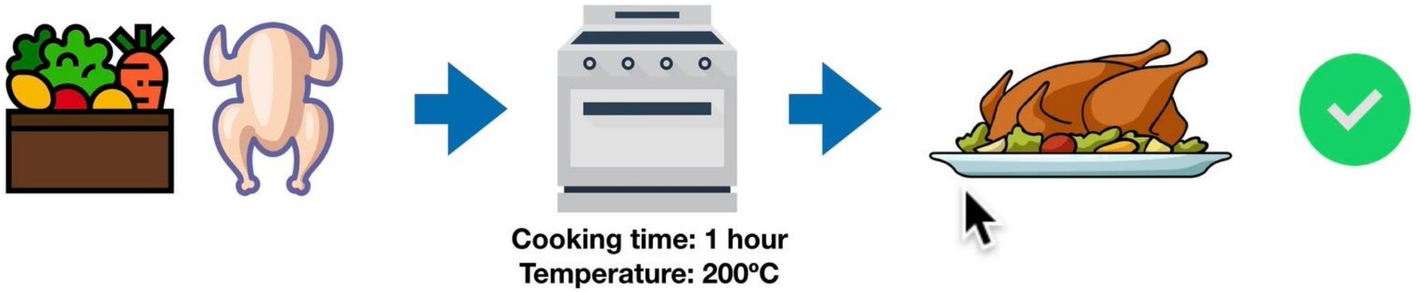
- 根據不同的資料，去選擇合適的模型。



# Model Tuning



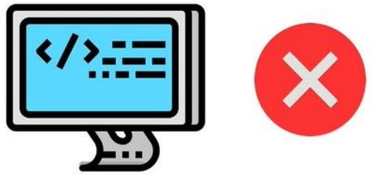

```python
# 手動調整第
np.random.seed(42)
for i in range(10, 100, 10):
    print(f"Trying model with {i} estimators...")
    model = RandomForestClassifier(n_estimators=i).fit(X_train, y_train)
    print(f"Model accruacy on test set: {model.score(X_test, y_test)}")
    print("")
```

```
Trying model with 10 estimators...
Model accruacy on test set: 0.7540983606557377

Trying model with 20 estimators...
Model accruacy on test set: 0.8360655737704918

Trying model with 30 estimators...
Model accruacy on test set: 0.7540983606557377

Trying model with 40 estimators...
Model accruacy on test set: 0.7704918032786885

Trying model with 50 estimators...
Model accruacy on test set: 0.7868852459016393

Trying model with 60 estimators...
Model accruacy on test set: 0.819672131147541

Trying model with 70 estimators...
Model accruacy on test set: 0.7540983606557377

Trying model with 80 estimators...
Model accruacy on test set: 0.8032786885245902

Trying model with 90 estimators...
Model accruacy on test set: 0.819672131147541
```

# Model Comparison

- 欠擬合(❌)
- 模型在訓練集和測試集上的性能都很低。
- (1)模型太過簡單，無法捕捉數據中的特徵和模式。(2)可能是使用的模型過於基礎，或者訓練不夠充分。
- 解決方案：
  - 使用更複雜的模型（例如從線性模型升級到樹模型或神經網路）。
  - 增加特徵工程，改善數據品質。
  - 增加訓練時間或調整超參數。

| Data Set | Performance |
|---|---|
| Training | 98% |
| Test | 96% |



Underfitting (potential)

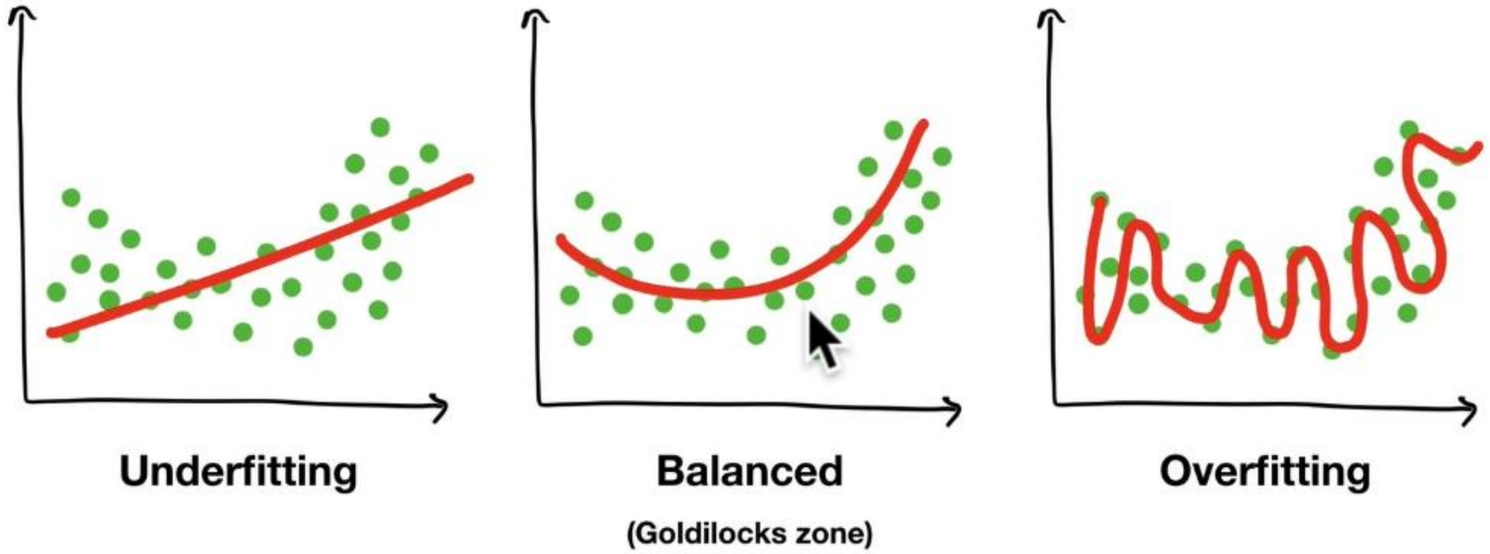
| Data Set | Performance |
|---|---|
| Training | 64% |
| Test | 47% |

Overfitting (potential)

| Data Set | Performance |
|---|---|
| Training | 93% |
| Test | 99% |

# Model Comparison

- 過擬合(✕)
- 現象：模型在訓練集上表現極佳，但在測試集上的性能下降或不自然地高。
- 模型過於複雜，記住了訓練集的細節和噪音，而無法泛化到新數據。
- 解決方案：
  - 使用正則化方法（如 L1、L2 正則化）來約束模型。
  - 簡化模型結構或使用較小的模型。
  - 增加訓練數據，讓模型學到更通用的模式。
  - 使用交叉驗證（Cross-Validation）
    來檢查模型泛化能力。



# Indicator

### 10.1.1. Accuracy（準確率）

- 模型預測正確的樣本數量佔總樣本數的比例
- 適用場景：當數據集的類別分布均衡時（例如 50% 是 A 類，50% 是 B 類）。
- 缺點：對於類別不平衡的數據集效果不好，容易偏向佔比多的類別。****
- $Accuracy = \frac{\text{正確預測的樣本數}}{\text{總樣本數}}$

### 10.1.2. Precision（精確率）

- 定義：模型預測正確的樣本數量佔總樣本數的比例。
- 適用場景：當 錯誤預測為正類的代價較高 時，例如醫療診斷中的誤報。
- $Precision = \frac{\text{真陽性 (TP)}}{\text{真陽性 (TP)} + \text{假陽性 (FP)}}$

### 10.2.2. Mean Squared Error (MSE)
### 10.2.2. 均方誤差 （MSE）

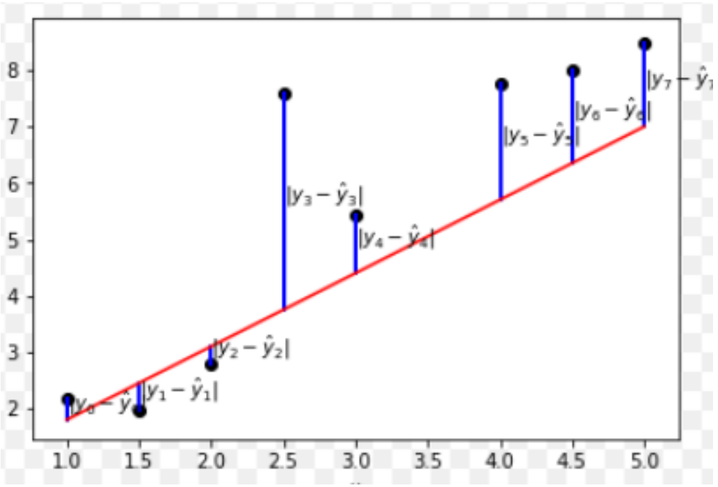
- 定義：預測值與真實值之間的平方誤差的平均值。
- 特點：誤差被平方，強調大誤差的影響。
- 缺點：單位變化（平方後的數值不直觀）。
- $MSE = 1n\sum i = 1n(yi - y^i)2MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

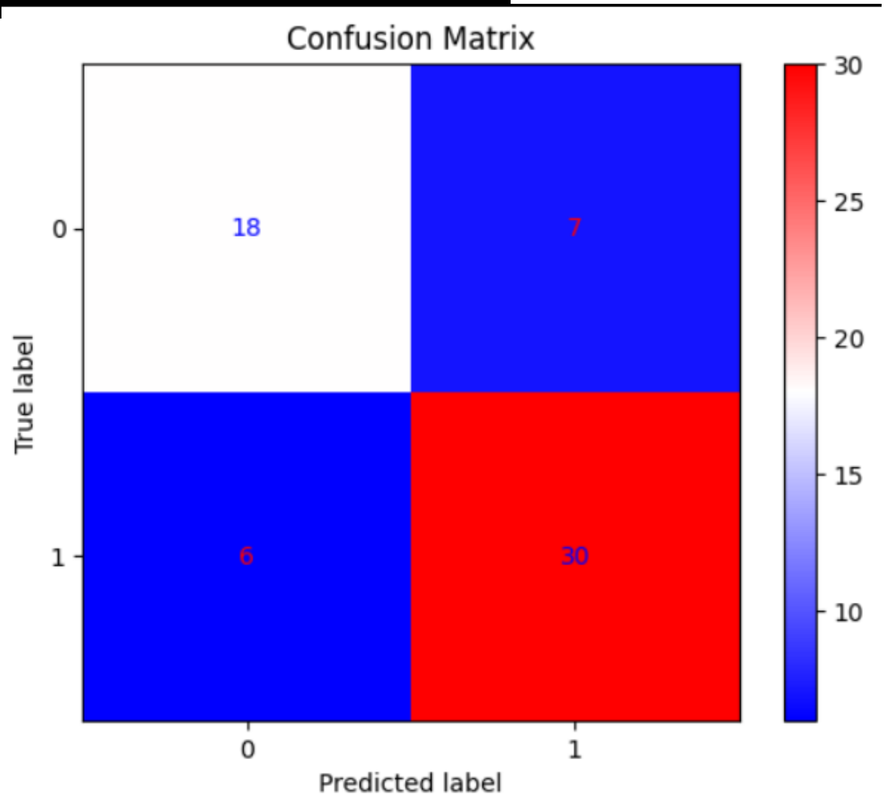
```
# 如果是以train set進行比較，一定會是1(就是用X_train, y_train來訓練的)
model.score(X_train, y_train)
```

```
1.0
```

```
model.score(X_test, y_test)
```

```
0.7868852459016393
```

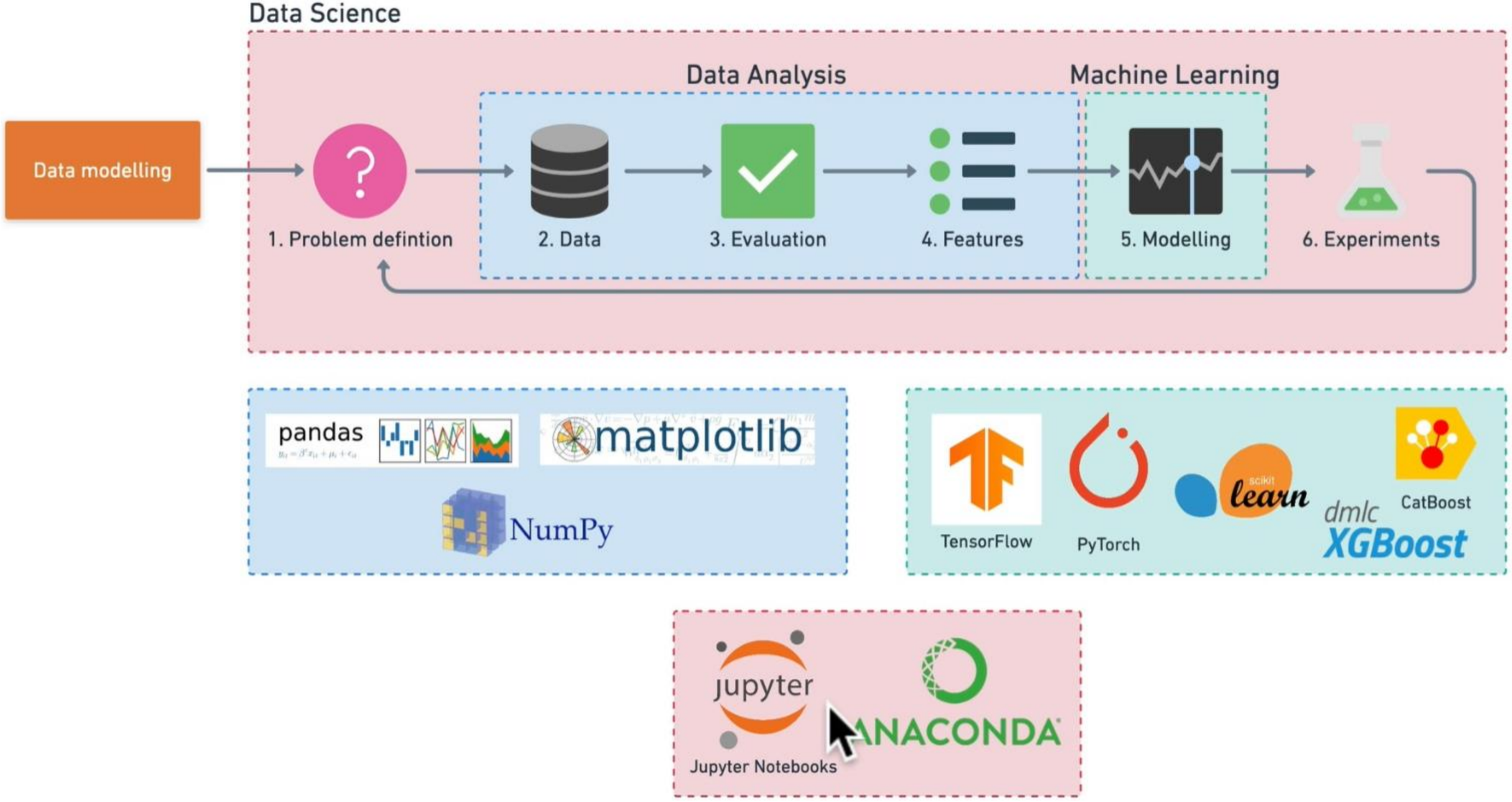


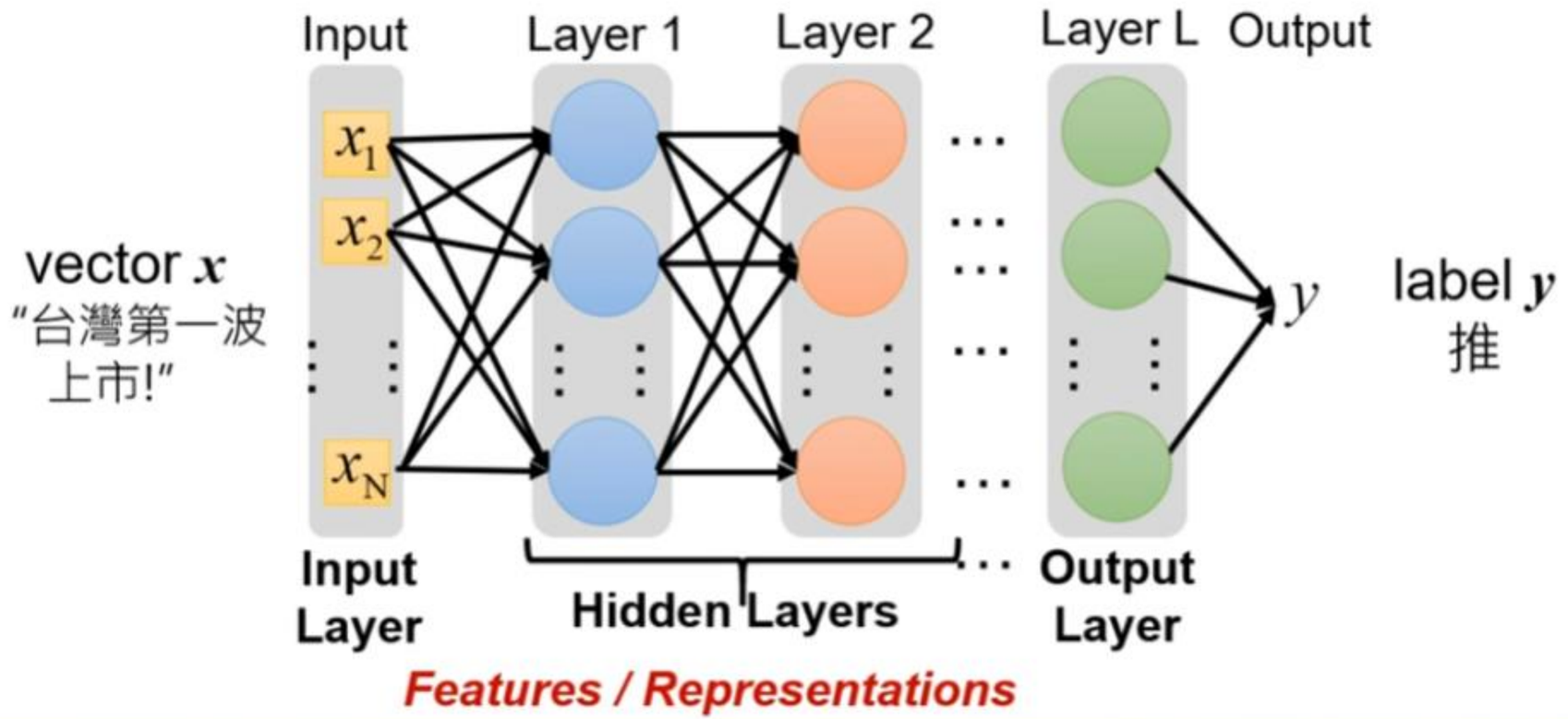

Heat map

# Tools We Will Use

- 複雜的工作流，是去如何實現的



# What is Deep Learning




Representation Learning attempts to learn good features/representations

Deep Learning attempts to learn (multiple levels of) representations and an output

來源：https://www.youtube.com/watch?v=F-HjOMQSQYE

# Large language Model

- 基於自然語言相關任務的深度學習模型


## GPT-3的训练数据

| Dataset | Quantity (tokens) | Weight in training mix | Epochs elapsed when training for 300B tokens |
|---|---|---|---|
| Common Crawl (filtered) | 410 billion | 60% | 0.44 |
| WebText2 | 19 billion | 22% | 2.9 |
| Books1 | 12 billion | 8% | 1.9 |
| Books2 | 55 billion | 8% | 0.43 |
| Wikipedia | 3 billion | 3% | 3.4 |

https://arxiv.org/abs/2005.14165


**Common Crawl** 网络爬虫公开数据集（海量互联网内容）

**WebText2** Reddit 论坛的网页文本

**Books1, Books2** 互联网书籍语料库

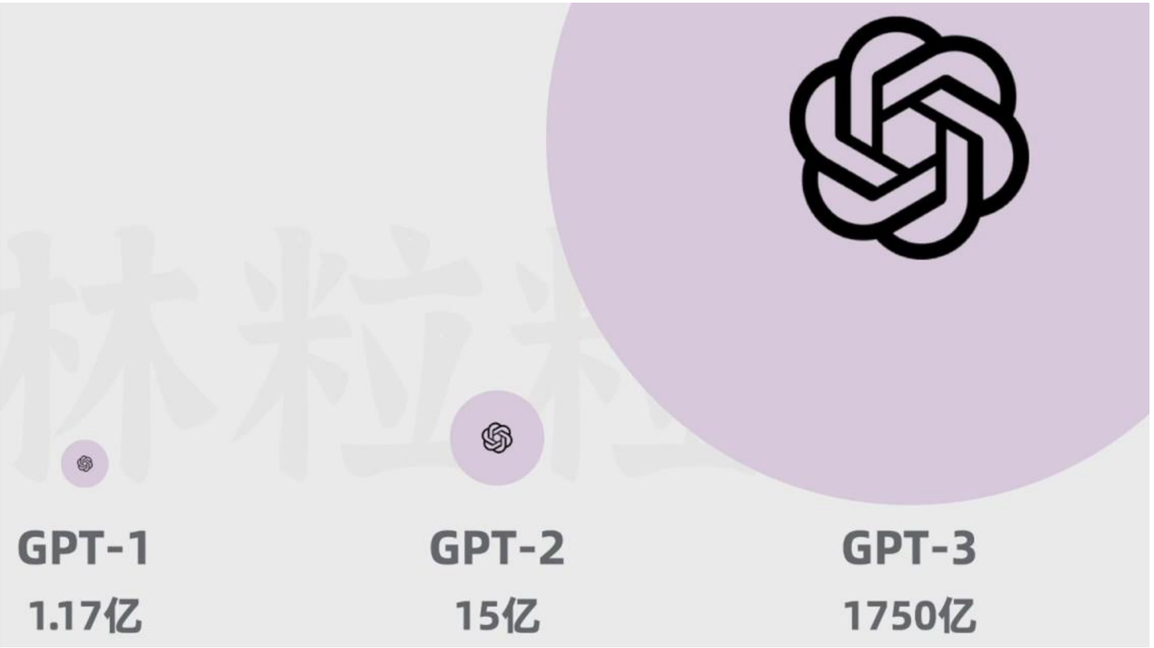
**Wikipedia** 整个英文维基百科知识库

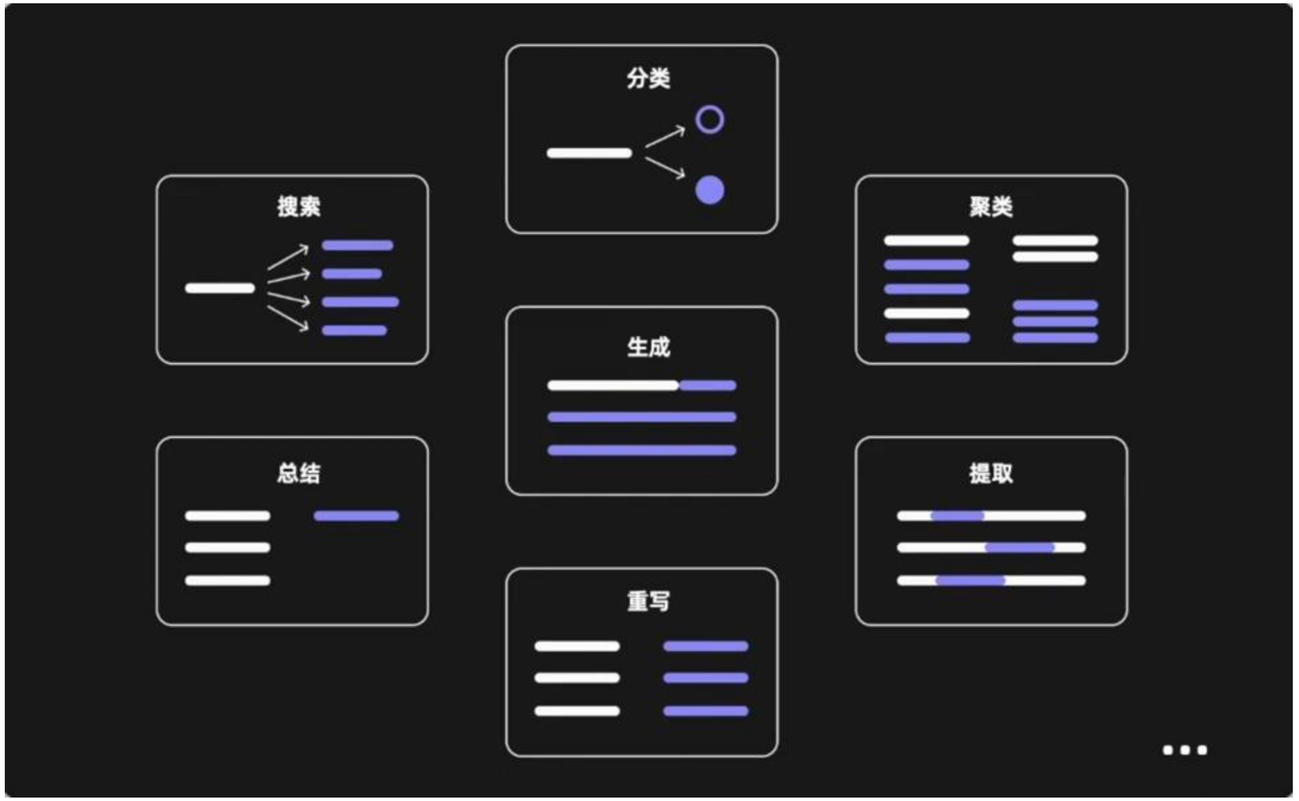
→ 3000亿文本单位

來源：https://youtu.be/la9nhavGG5M?si=vs3emH0k3GqBw_bF

# Large language Model

- GPT：Generative (生成) Pre-treaned(預訓練的) Transformer

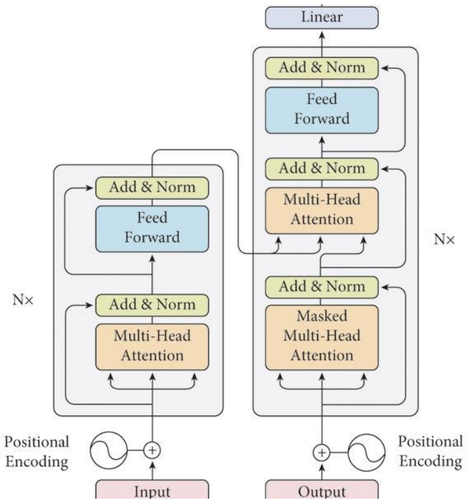



來源：https://youtu.be/la9nhavGG5M?si=vs3emH0k3GqBw_bF

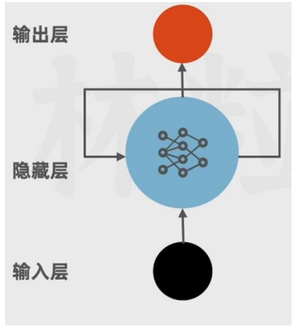
# Transformer

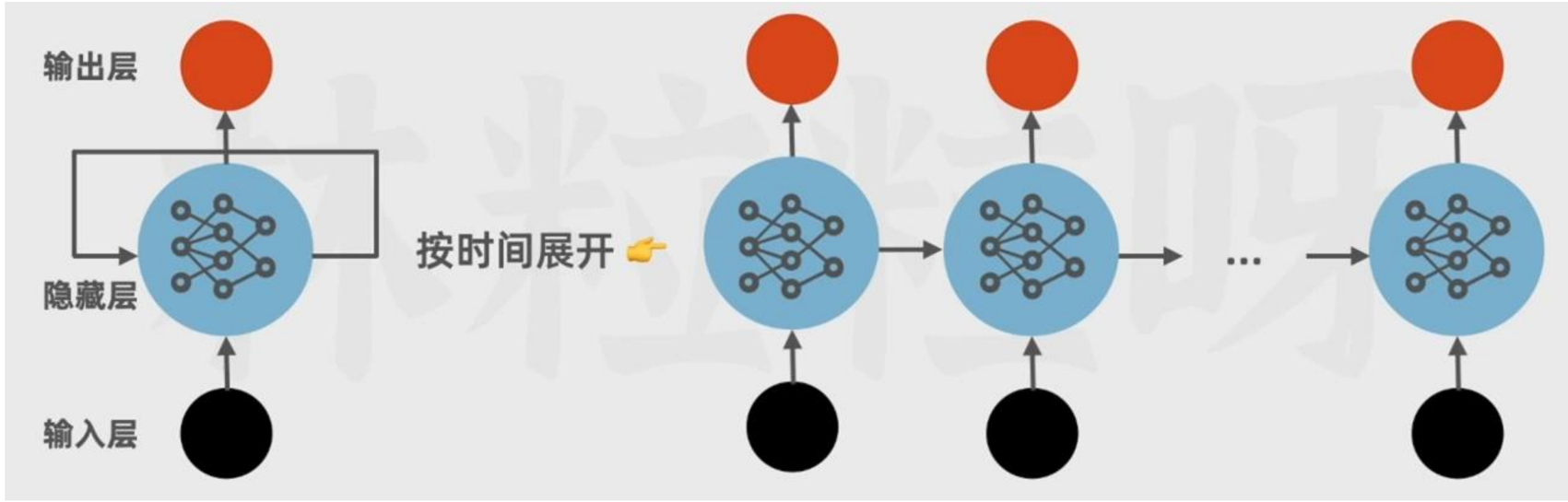
- GPT：Generative (生成) Pre-treaned(預訓練的) Transformer




Transformor structure

- 在transformor之前是基於RNN




- 我在高雄長大，在台南讀書。喜歡日本的吉伊卡哇，我的日常晚餐是OOOO…

來源：https://youtu.be/la9nhavGG5M?si=vs3emH0k3GqBw_bF

# Transformer

**Attention Is All You Need**


**Ashish Vaswani*** Google Brain avaswani@google.com

**Noam Shazeer*** Google Brain noam@google.com

**Niki Parmar*** Google Research nikip@google.com

**Jakob Uszkoreit*** Google Research usz@google.com

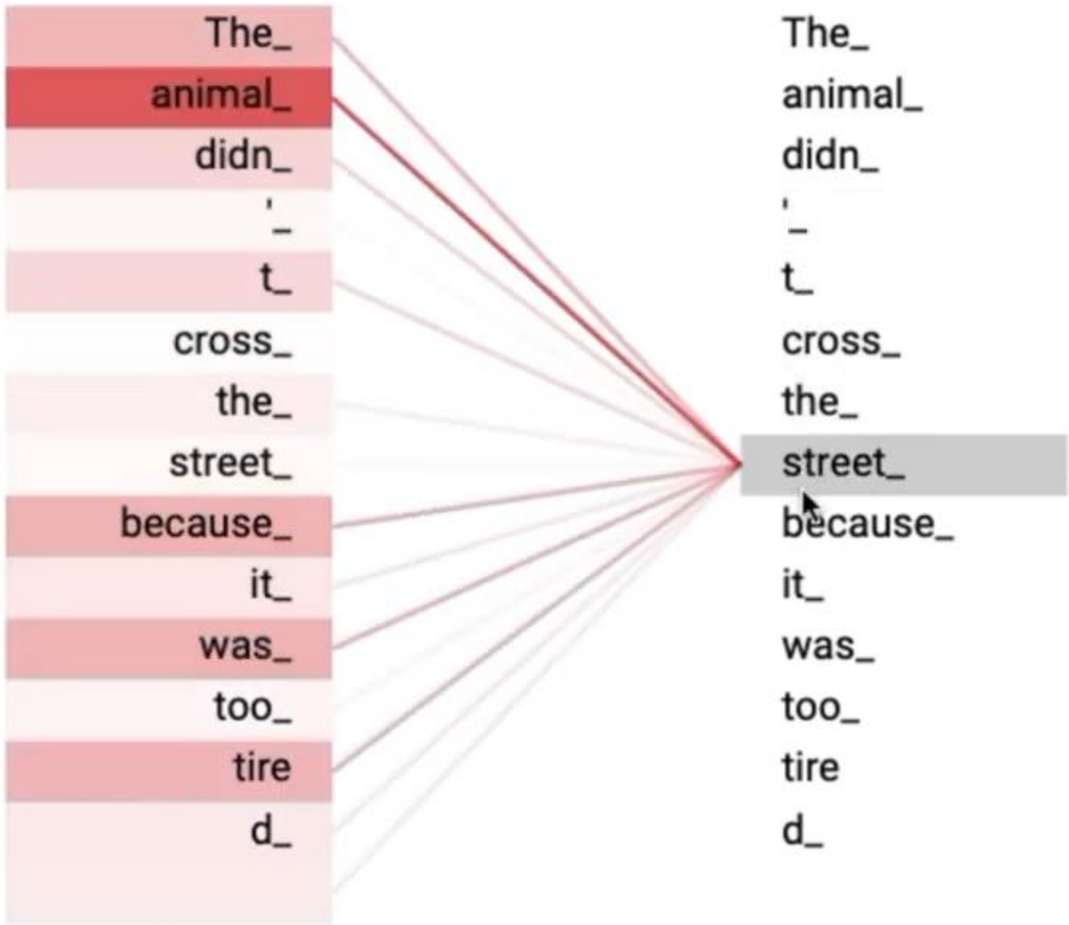
**Llion Jones*** Google Research llion@google.com

**Aidan N. Gomez*** † University of Toronto aidan@cs.toronto.edu

**Łukasz Kaiser*** Google Brain lukaszkaiser@google.com

**Illia Polosukhin*** ‡ illia.polosukhin@gmail.com



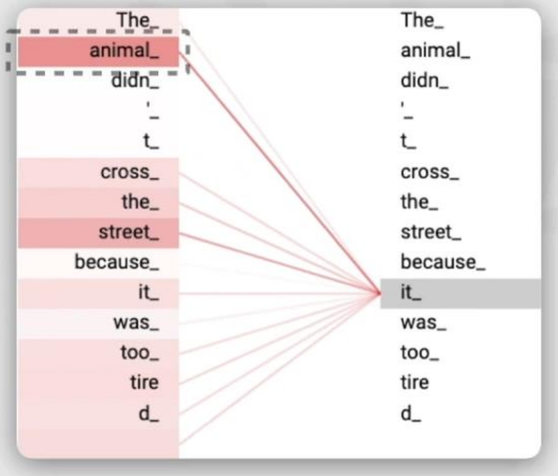



來源：https://youtu.be/la9nhavGG5M?si=vs3emH0k3GqBw_bF

# Transformer

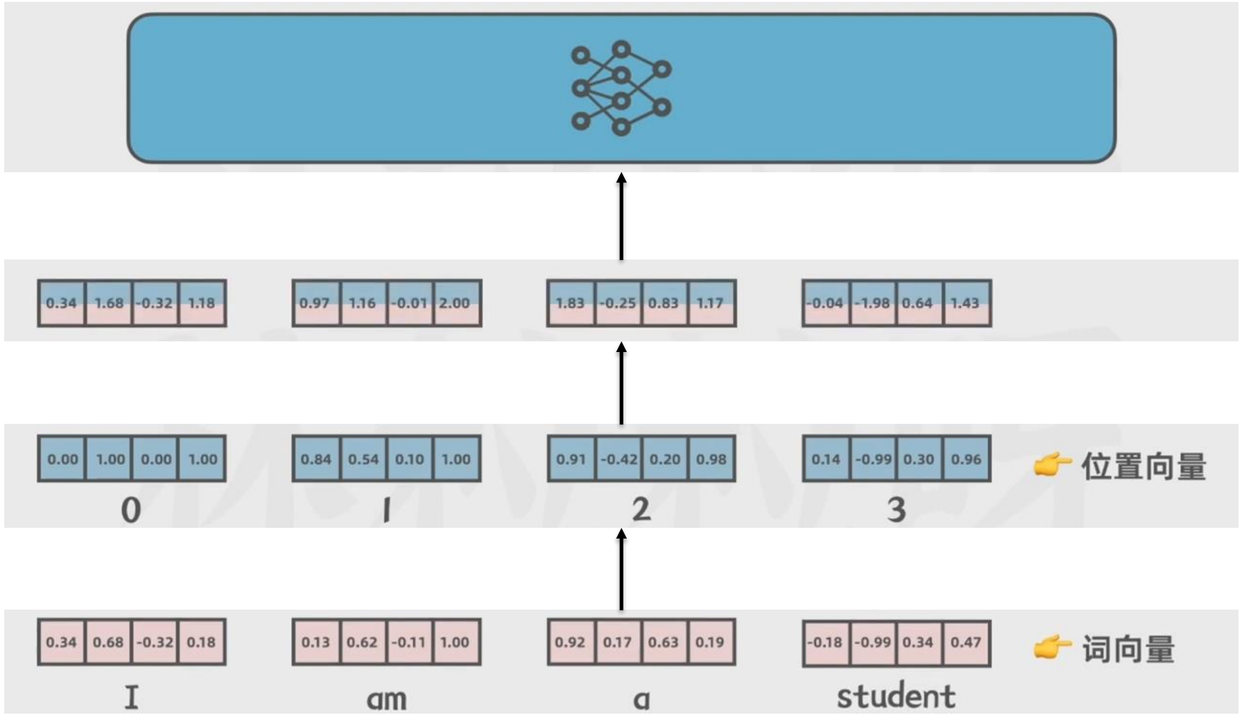
👀 下面的it，指的是什么？

The animal didn't cross the street because it was too tired




👉 可能是“animal”，也可能是“street”

👉 这里自注意力机制更集中在“animal”上




來源：https://youtu.be/la9nhavGG5M?si=vs3emH0k3GqBw_bF

# Transformer-encoder

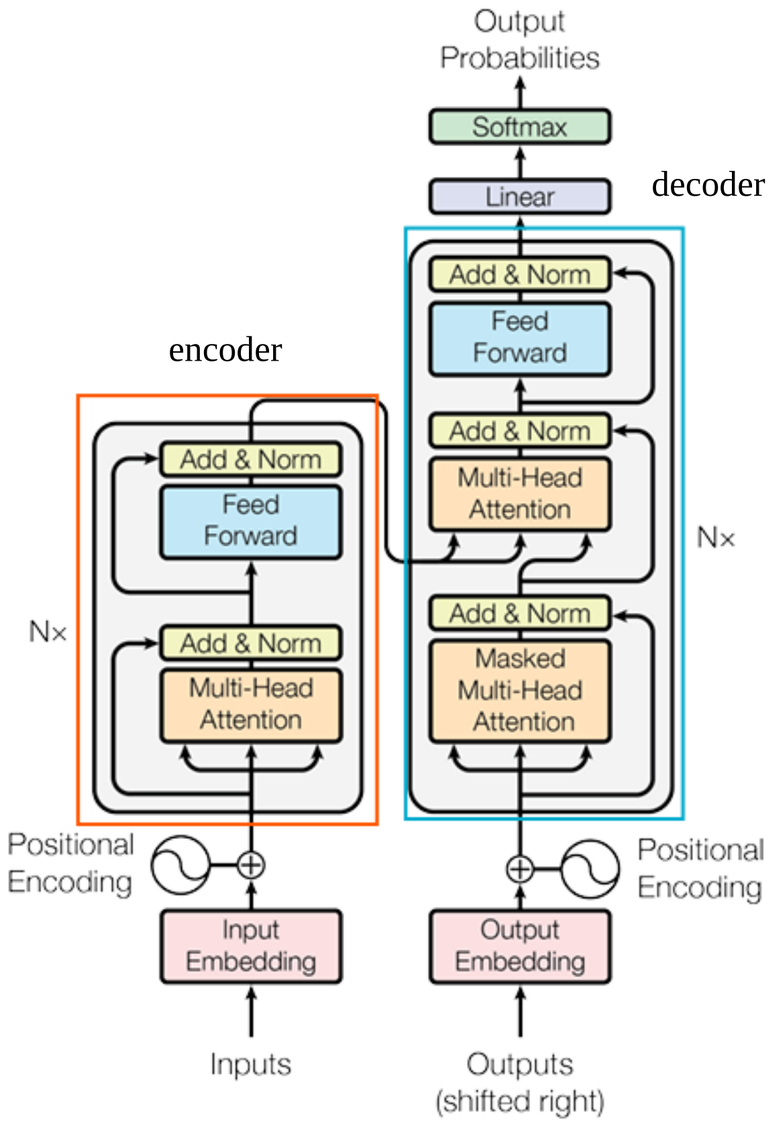
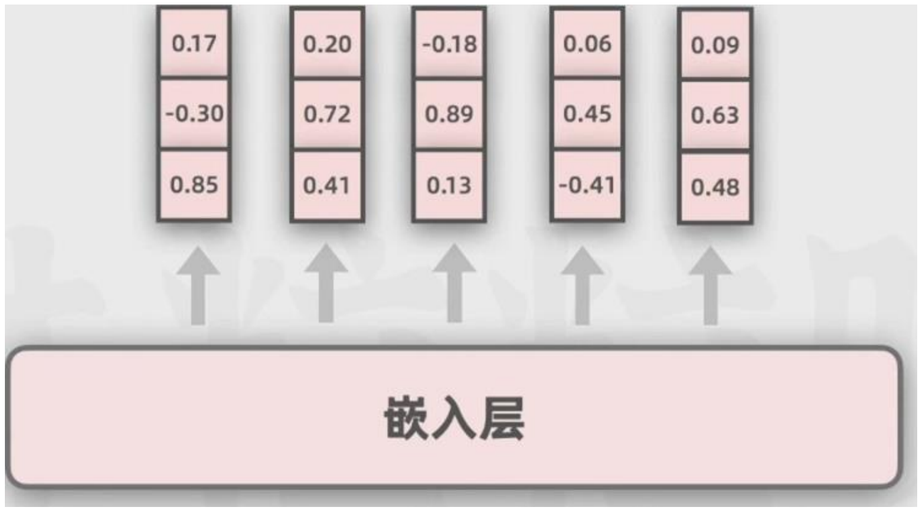
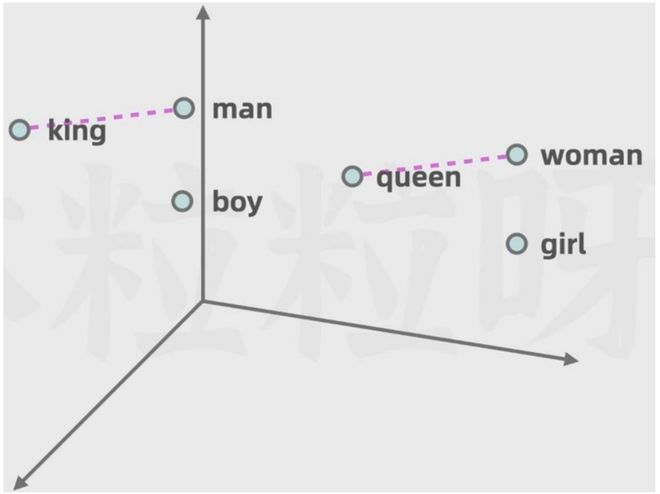
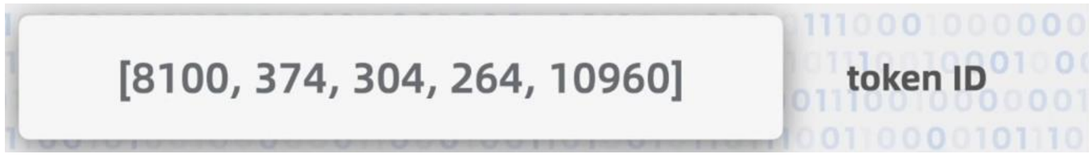


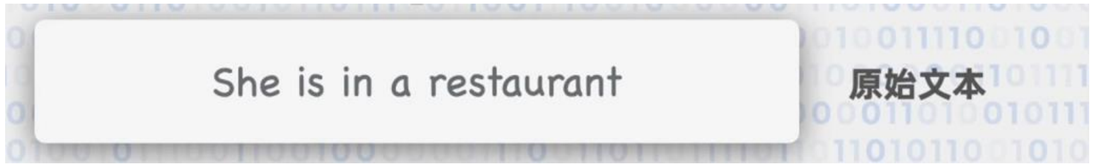

Figure 1: The Transformer - model architecture.








來源：https://youtu.be/la9nhavGG5M?si=vs3emH0k3GqBw_bF

# Transformer-decoder

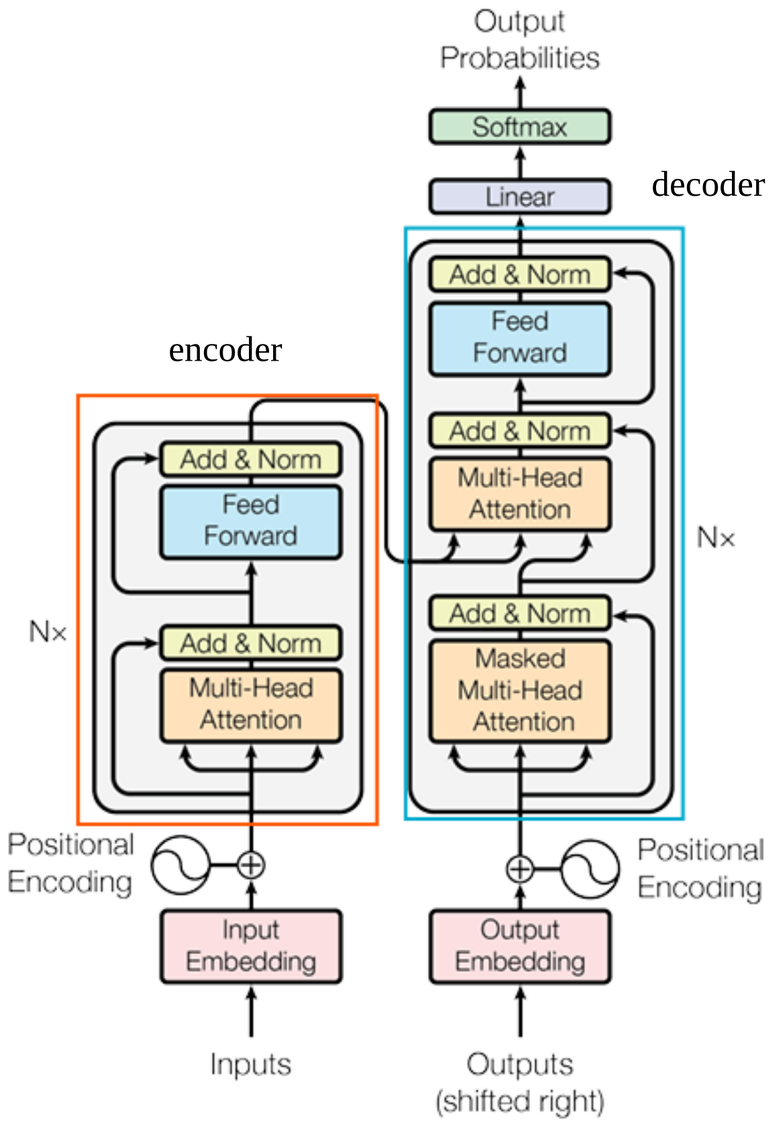
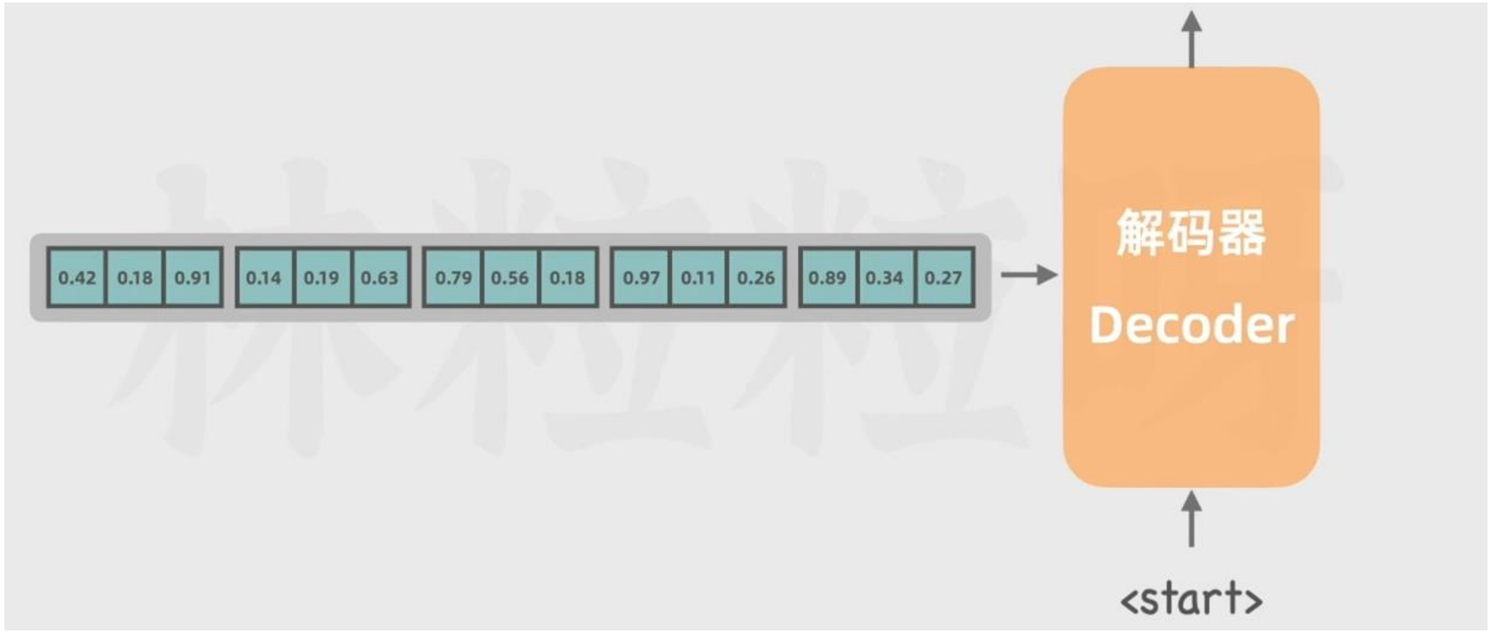



Figure 1: The Transformer - model architecture.

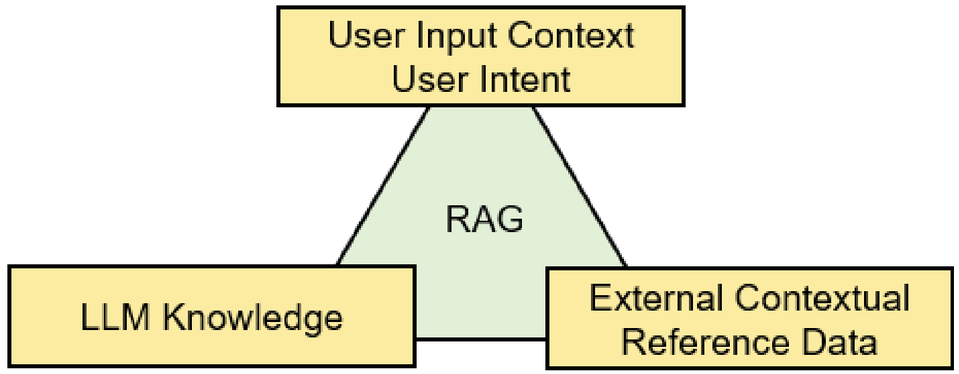




來源：https://youtu.be/la9nhavGG5M?si=vs3emH0k3GqBw_bF

# RAG

- 目前LLM的缺失：
  - 知識限制和時效性：傳統的語言模型(GPT-3 或 GPT-4)是基於預先訓練的數據進行訓練的，這些數據在特定時間點截斷，無法即時更新知識。
  - 提升回應準確性：純粹的生成模型可能因為「幻覺」問題（Hallucination）
  - 降低參數依賴和計算成本：要讓LLM儲存所有可能的知識，模型的參數量會急劇增加，導致訓練成本和推理成本居高不下。
  - 更高的靈活性和擴展性：傳統LLM的知識更新需要重新訓




**透過檢索增強生成技術提升學習成效：**

**應用於大學計算機概論課程的教育聊天機器人**

*Enhancing Learning Outcomes through Retrieval-Augmented Generation:*
*An Educational Chatbot for Computer Science Courses*

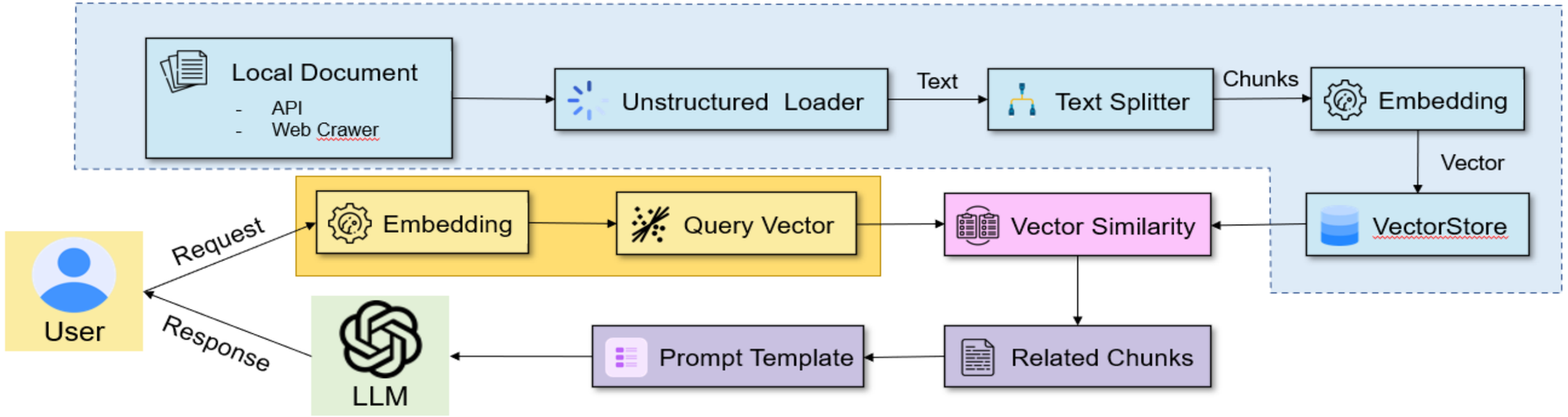
**ABSTRSCT**

本研究旨在設計並開發一套基於 RAG 的教育聊天機器人，應用於大學計算機概論課程，嵌入歷屆考古題以提供更為個性化與高效的學習輔助功能。

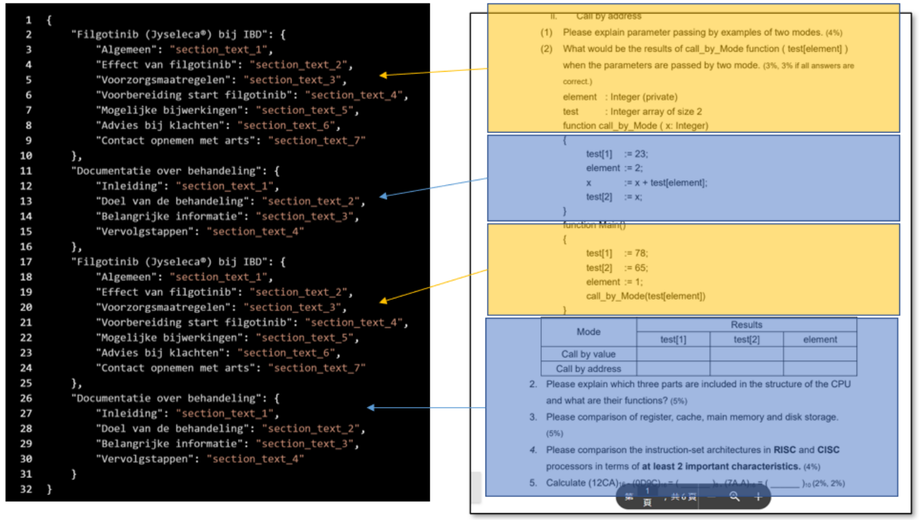
**INDRODUCTION**

聊天機器人技術在過去六十年間實現了顯著進步，這得益於自然語言處理(NLP)和機器學習(ML)的快速發展，以及大型語言模型(LLMs)的廣泛應用。聊天機器人是一種基於文本的數位對話代理，通過自然語言處理技術與用戶進行互動(Wirtz et al., 2018)。相比於傳統資訊系統，聊天機器人在互動性和智慧化方面具有優勢(Maedche et al., 2019)。這些特性使其成為企

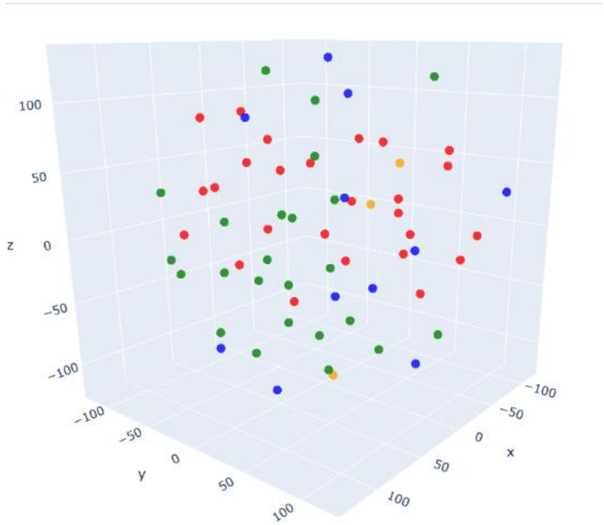
# RAG




RAG架構




文本分割




文字轉向量存儲(Vector Store)

# RAG

```
"""
##### NOTE: 加載 Markdown 文件 #####
從 "knowledge-base" 資料夾中加載所有 Markdown 文件，並將它們存儲為內存中的文檔列表。
"""
text_loader_kwargs = {'encoding': 'utf-8'}
folders = glob.glob("./誤刪knowledge-base/*")  # 獲取資料夾清單
documents = []  # 用於存儲文檔
for folder in folders:
    doc_type = os.path.basename(folder)  # 獲取文件類型（資料夾名稱）
    loader = DirectoryLoader(
        folder,
        glob="**/*.md",  # 遍歷資料夾中所有 Markdown 文件
        loader_cls=TextLoader,
        loader_kwargs=text_loader_kwargs
    )
    folder_docs = loader.load()  # 加載文檔
    # 添加 metadata（例如文件類型）
    for doc in folder_docs:
        doc.metadata["doc_type"] = doc_type
        documents.append(doc)
print(f"加載文件數量：{len(documents)}")  # 打印加載的文檔數量
```

# RAG

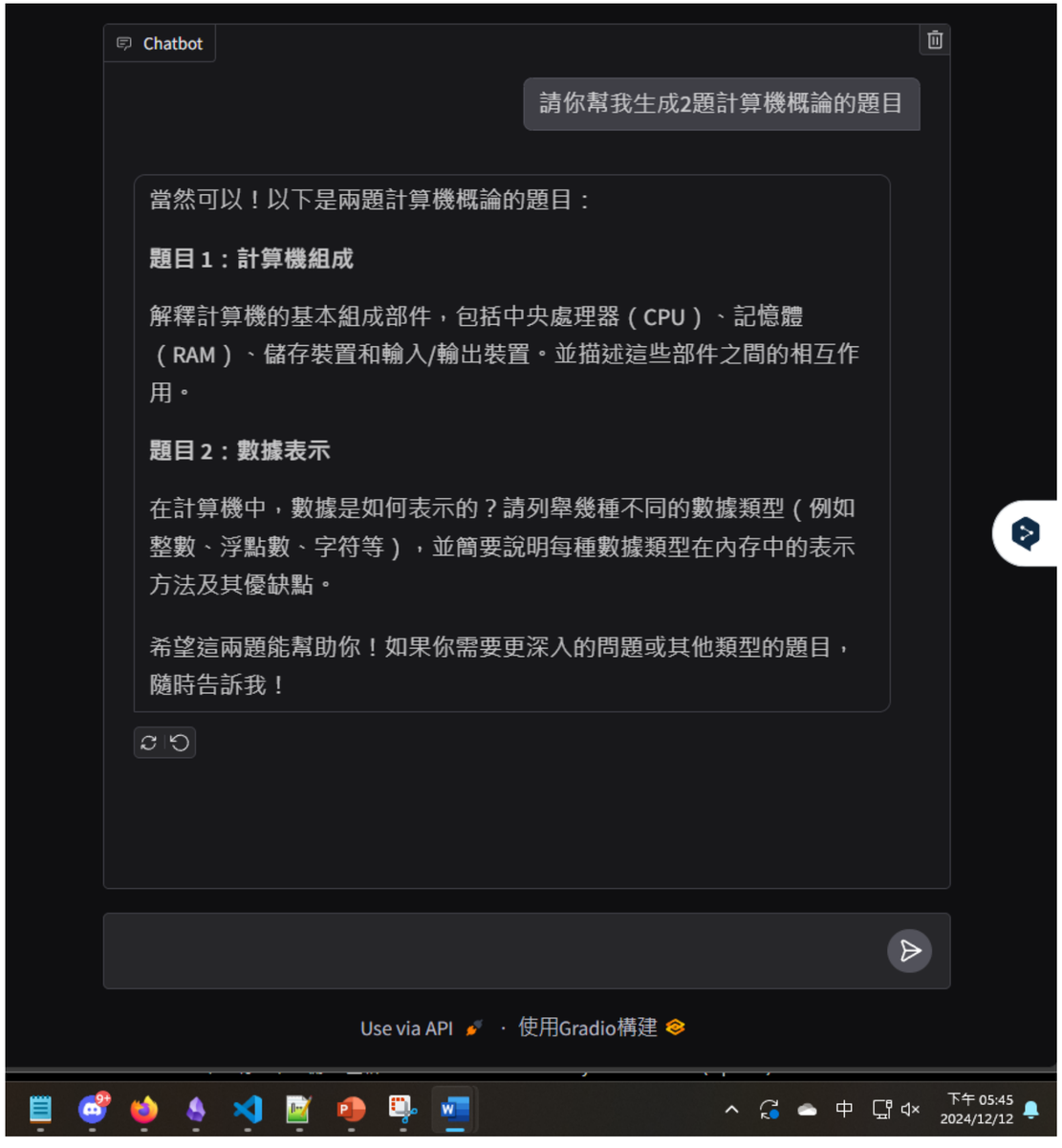
```
54  """
55  ##### NOTE: 分割文件為文本塊 #####
56  使用 CharacterTextSplitter 將每個文檔分割為較小的文本塊。
57  每塊的大小為 2000 字符，並有 400 字符的重疊部分，確保上下文連貫。
58  """
59  text_splitter = CharacterTextSplitter(chunk_size=2000, chunk_overlap=400)  # 分割參數
60  chunks = text_splitter.split_documents(documents)  # 分割文檔
61  print(f"生成小塊數量：{len(chunks)}")  # 打印生成的小塊數量
```

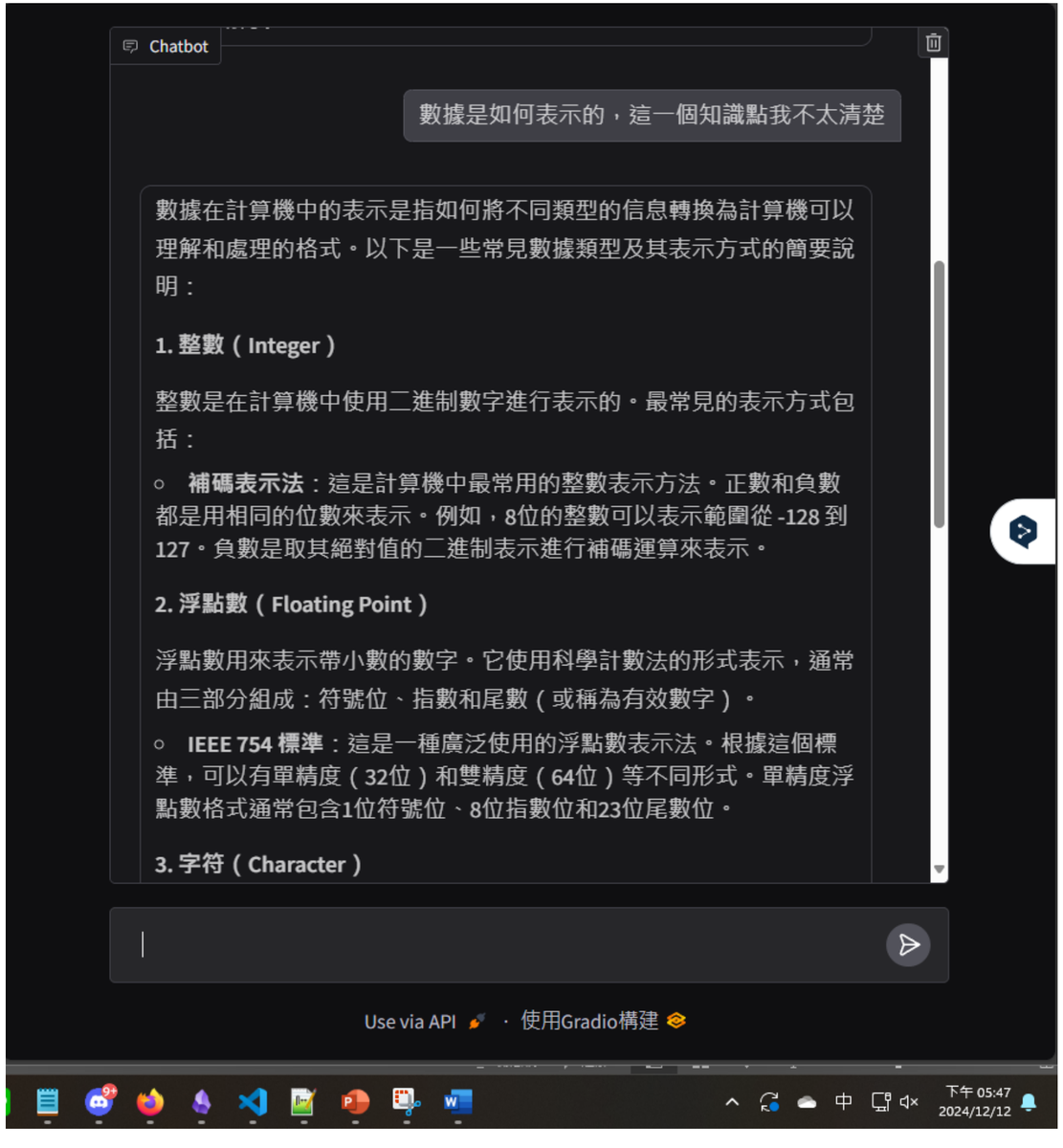
```
66  """
67  ##### NOTE: 建立向量存儲 #####
68  1. 使用 OpenAI 的嵌入模型將文本塊轉換為高維向量。
69  2. 將生成的嵌入向量存儲到 Chroma 數據庫中。
70  """
71  embeddings = OpenAIEmbeddings()  # 初始化嵌入模型
72  if os.path.exists(CFG.db_name):  # 如果向量數據庫已存在，刪除它
73      Chroma(persist_directory=CFG.db_name, embedding_function=embeddings).delete_collection()
74
75  vectorstore = Chroma.from_documents(
76      documents=chunks,  # 文本塊
77      embedding=embeddings,  # 嵌入模型
78      persist_directory=CFG.db_name  # 持久化數據庫位置
79  )
80  print(f"Vectorstore created with {vectorstore._collection.count()} documents")  # 打印存儲的文檔數量
```

# RAG

```
83    """
84    ##### NOTE: 啟動聊天機器人 #####
85    1. 使用 ConversationalRetrievalChain 建立檢索型對話鏈。
86    2. 使用 Gradio 提供一個簡單的聊天界面。
87    """
88    llm = ChatOpenAI(temperature=0.7, model_name=CFG.MODEL)  # 初始化 GPT 模型
89    retriever = vectorstore.as_retriever()  # 從向量存儲中檢索相關文檔
90    memory = ConversationBufferMemory(memory_key='chat_history', return_messages=True)  # 記錄對話歷史
91    conversation_chain = ConversationalRetrievalChain.from_llm(llm=llm, retriever=retriever, memory=memory)
92
93    # 聊天邏輯定義
94    def chat(message, history):
95        """
96        接收用戶輸入的問題與歷史對話，返回基於文檔的回答。
97        """
98        result = conversation_chain.invoke({"question": message})
99        return result["answer"]
100   view = gr.ChatInterface(chat, type="messages").launch(inbrowser=True)  # 啟動 Gradio Chat 介面
```

# RAG


Chatbot
請你幫我生成2題計算機概論的題目
當然可以！以下是兩題計算機概論的題目：
題目 1：計算機組成
解釋計算機的基本組成部件，包括中央處理器（CPU）、記憶體（RAM）、儲存裝置和輸入/輸出裝置。並描述這些部件之間的相互作用。
題目 2：數據表示
在計算機中，數據是如何表示的？請列舉幾種不同的數據類型（例如整數、浮點數、字符等），並簡要說明每種數據類型在內存中的表示方法及其優缺點。
希望這兩題能幫助你！如果你需要更深入的問題或其他類型的題目，隨時告訴我！
Use via API · 使用Gradio構建
下午 05:45
2024/12/12


Chatbot
數據是如何表示的，這一個知識點我不太清楚
數據在計算機中的表示是指如何將不同類型的信息轉換為計算機可以理解和處理的格式。以下是一些常見數據類型及其表示方式的簡要說明：
1. 整數（Integer）
整數是在計算機中使用二進制數字進行表示的。最常見的表示方式包括：
補碼表示法：這是計算機中最常用的整數表示方法。正數和負數都是用相同的位數來表示。例如，8位的整數可以表示範圍從 -128 到 127。負數是取其絕對值的二進制表示進行補碼運算來表示。
2. 浮點數（Floating Point）
浮點數用來表示帶小數的數字。它使用科學計數法的形式表示，通常由三部分組成：符號位、指數和尾數（或稱為有效數字）。
IEEE 754 標準：這是一種廣泛使用的浮點數表示法。根據這個標準，可以有單精度（32位）和雙精度（64位）等不同形式。單精度浮點數格式通常包含1位符號位、8位指數位和23位尾數位。
3. 字符（Character）
Use via API · 使用Gradio構建
下午 05:47
2024/12/12

# 超連結

【助教最後的小叮嚀】

- 下周一不上課，請大家帶樹梅派來歸還(或是自行找時間拿來實驗室)。
- 助教最近蠻有感的一段話【曾經的風雨會化為歲月的霞光，願你抱持溫柔凝向遠方】
- 祝大家 2025 新年快樂(未來如果各位大富大貴，記得內推一下助教)。



# 超連結

- 設定超連結顏色：link-underline-顏色
- 設定超連結透明度

(Currying)