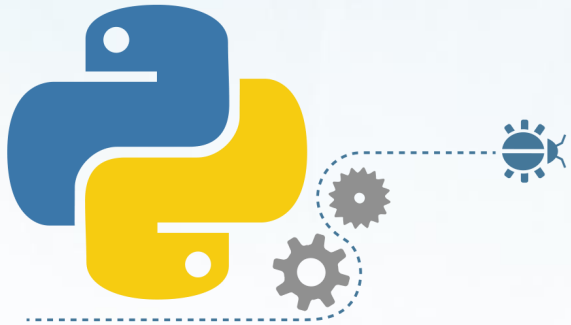
TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN HỆ THỐNG THÔNG TIN

NHẬP MÔN LẬP TRÌNH PYTHON (IPPA233277)

PHÂN TÍCH DỮ LIỆU
THƯ VIỆN NUMPY



GV. Trần Quang Khải

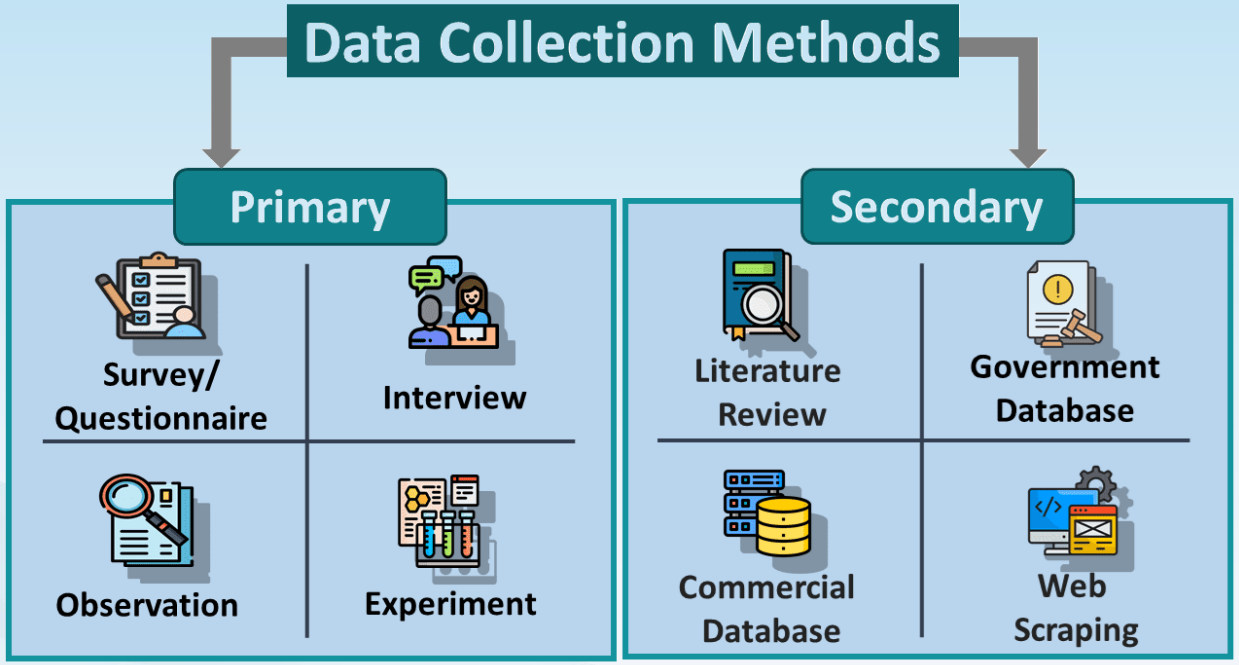
1. Nắm và hiểu được phân tích dữ liệu là gì?
2. Nắm và vận dụng được các hàm hỗ trợ thư viện Numpy

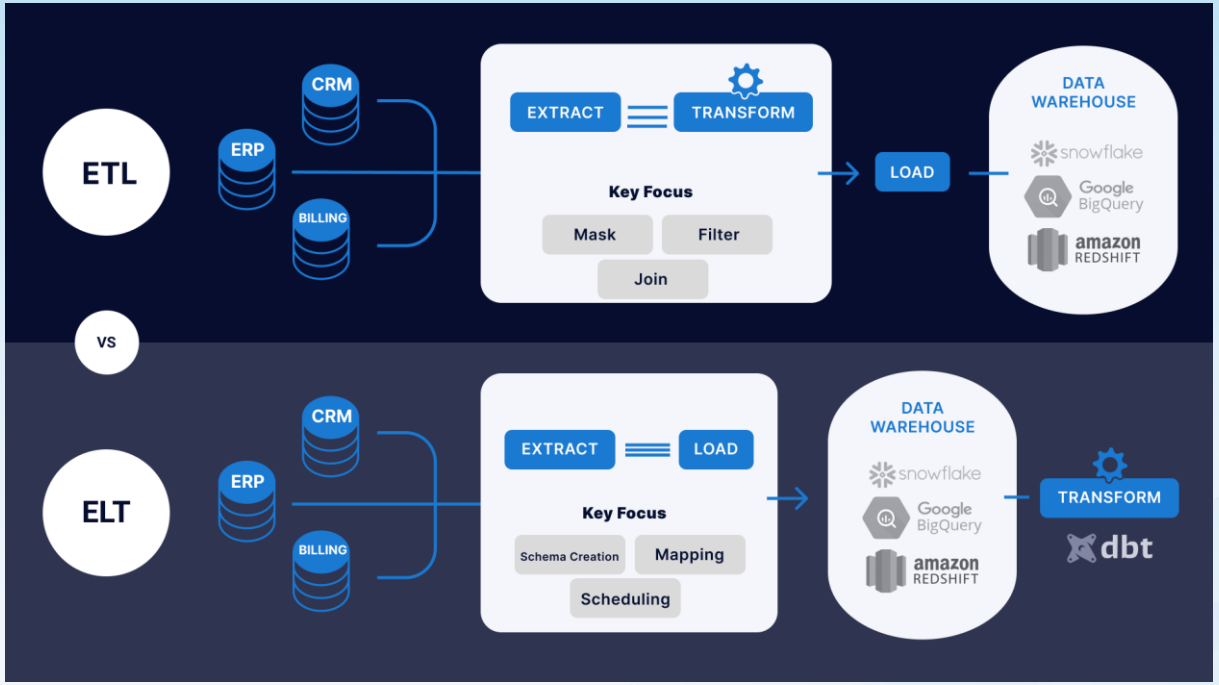
1. Giới thiệu về phân tích dữ liệu
2. Tiến trình phân tích dữ liệu
3. Phân loại phân tích dữ liệu
4. Các gói phân tích dữ liệu
5. Thư viện numpy

- Phân tích dữ liệu là quá trình quan trọng trong việc **đánh giá**, **làm sạch**, **biến đổi** và **mô hình hóa dữ liệu** nhằm mục đích tìm ra thông tin hữu ích, đưa ra kết luận và hỗ trợ ra quyết định.
- Mục đích chính của phân tích dữ liệu bao gồm việc hiểu rõ hơn về dữ liệu thông qua việc xác định các xu hướng, mẫu hình, và các đặc điểm khác giúp người phân tích có được cái nhìn sâu sắc hơn về dữ liệu, từ đó đưa ra những quyết định có cơ sở và thông minh hơn. Bên cạnh đó, phân tích dữ liệu còn có khả năng dự đoán và mô hình hóa, giúp dự báo các xu hướng và hành vi trong tương lai dựa trên dữ liệu hiện có.

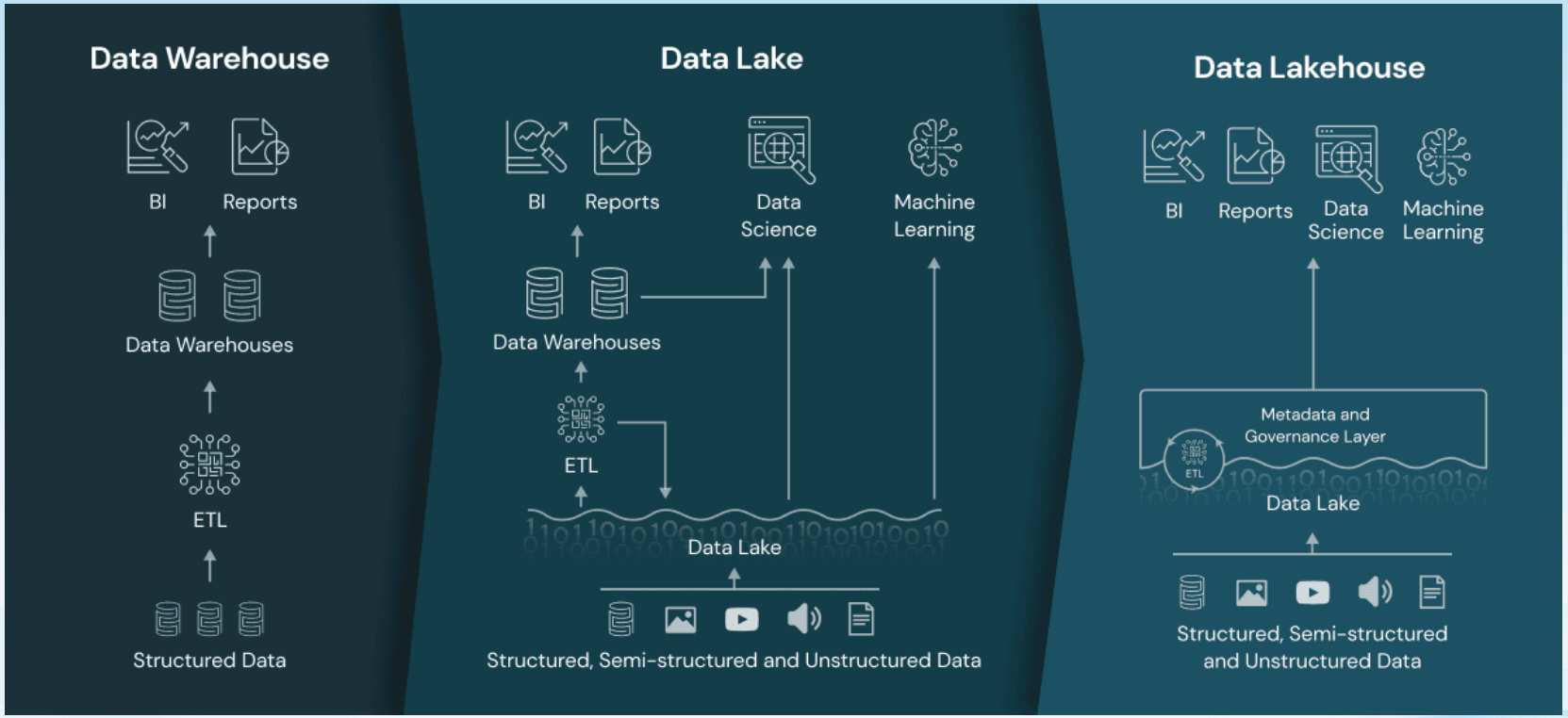
- Tiến trình phân tích dữ liệu có thể đơn giản hoặc phức tạp tùy theo mô hình đang được áp dụng vận hành gồm:
    1. Thu thập dữ liệu (Data Collection)
    2. Lưu trữ dữ liệu (Data Storage)
    3. Xử lý dữ liệu (Data Processing)
    4. Làm sạch dữ liệu (Data Cleansing)
    5. Phân tích dữ liệu (Data Analysis)
    6. Trực quan hóa dữ liệu (Data Visualization)

- Gồm các công việc xác định nguồn dữ liệu và thu thập dữ liệu từ các nguồn này
- Tuân thủ quá trình (**E**xtract, **T**ransform, **L**oad)
  - ✓ ETL: dữ liệu **trích xuất** được **chuyển đổi** thành định dạng tiêu chuẩn trước khi **tải** vào kho lưu trữ
  - ✓ ELT: dữ liệu **trích xuất** được **tải** vào kho lưu trữ, sau đó được **chuyển đổi** thành định dạng theo yêu cầu





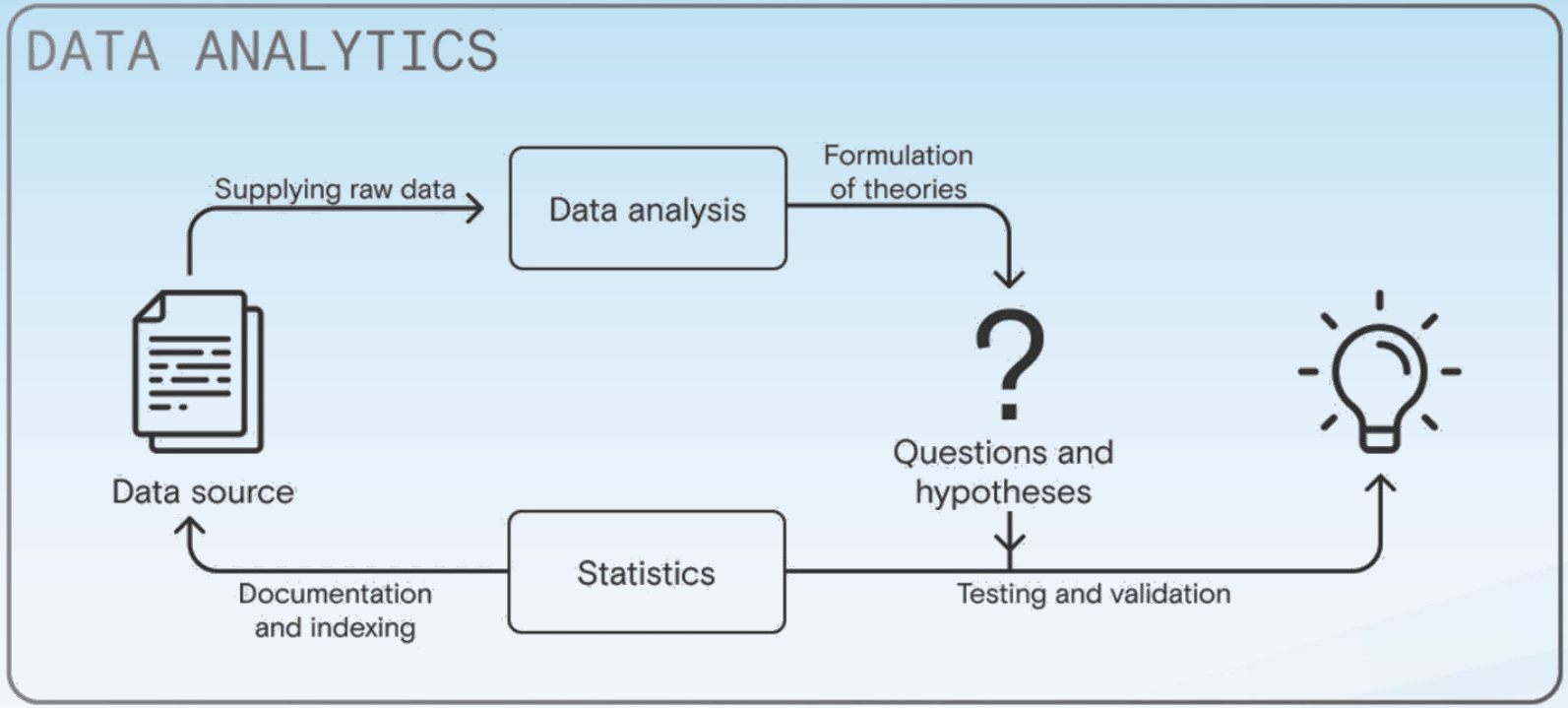
- Dựa trên mức độ phức tạp của dữ liệu, dữ liệu có thể được lưu trữ trong data warehouse hoặc data lake. **Data warehouse** là một cơ sở dữ liệu được tối ưu để phân tích dữ liệu quan hệ từ các hệ thống giao dịch và ứng dụng kinh doanh. **Data lake** có thể lưu trữ dữ liệu có cấu trúc và phi cấu trúc mà không cần xử lý thêm do đó, cho phép thu thập mọi dữ liệu quan trọng cần thiết.



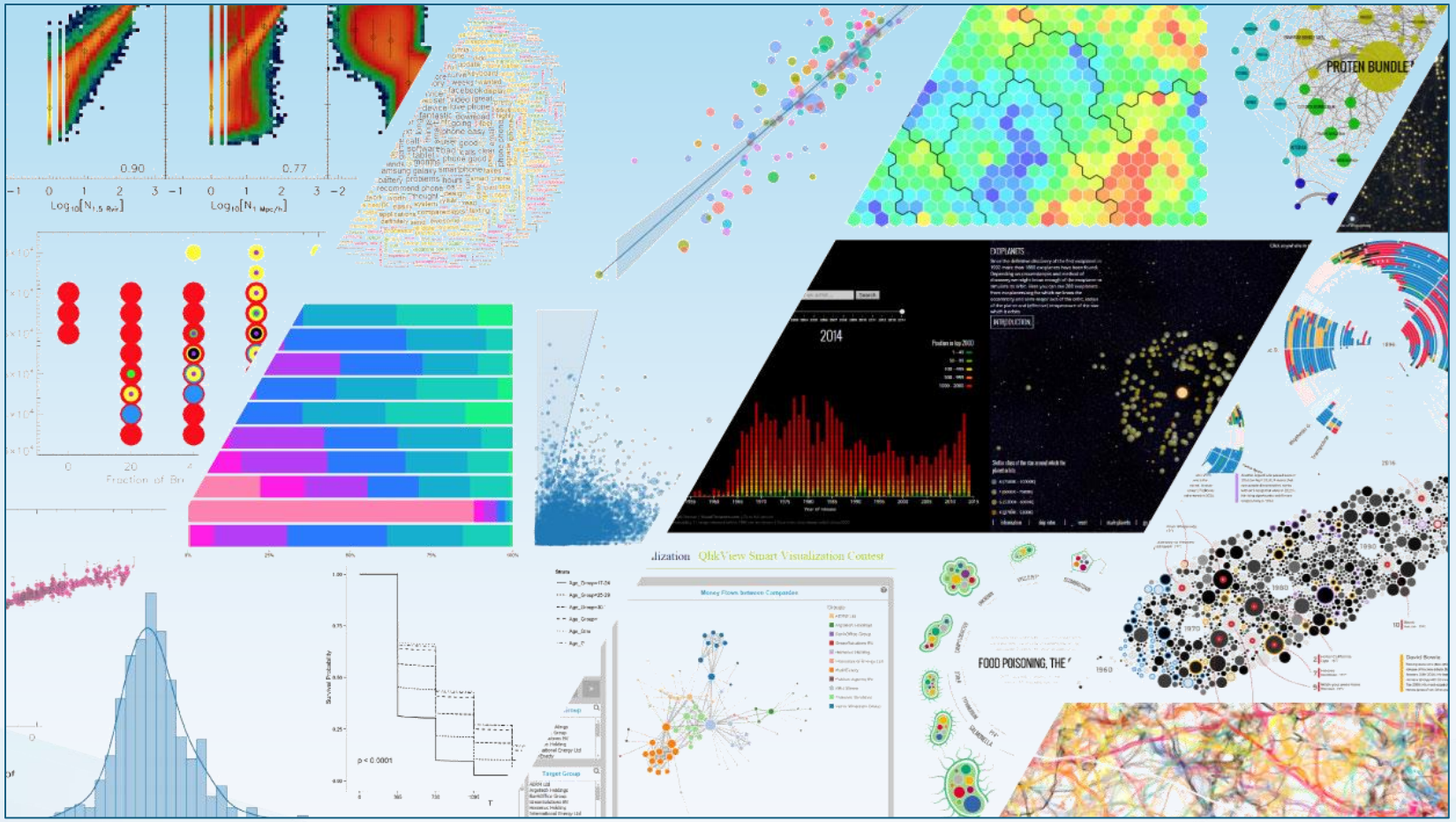
- Khi đã có dữ liệu, thì cần chuyển đổi, tổ chức để thu được kết quả chính xác từ các truy vấn phân tích
- Việc xử lý gồm một số cách thức sau:
  - ✓Xử lý tập trung: toàn bộ quá trình xử lý dữ liệu diễn ra trên một máy chủ trung tâm
  - ✓Xử lý phân tán: dữ liệu được phân tán và lưu trữ trên các máy chủ khác nhau
  - ✓Xử lý lô dữ liệu: các phần của dữ liệu tích lũy theo thời gian và được xử lý theo các lô (batch)
  - ✓Xử lý theo thời gian thực: dữ liệu được xử lý liên tục

- Gồm các công việc như xóa bất kỳ các vi phạm về mặt dữ liệu như trùng lặp, không nhất quán, dư thừa hoặc định dạng sai,… cũng như lọc các dữ liệu bất kỳ bài không cần thiết (không hữu ích) trong quá trình phân tích
- Bước này thường là bắt buộc cần thực hiện trước khi phân tích dữ liệu

- Là bước chuyển đổi dữ liệu thô thành thông tin chuyên sâu hữu ích
- Tập trung vào việc sử dụng các công cụ và kỹ thuật phân tích phù hợp để trích xuất thông tin quan trọng từ các dữ liệu đã được xử lý
- Kết quả phân tích có thể làm phát sinh các yêu cầu mới để làm rõ kết quả, đòi hỏi việc thu thập thêm dữ liệu hoặc tiến hành phân tích bổ sung



- Là bước chuyển đổi dữ liệu thành dạng thông tin trực quan sinh động dễ phân tích hơn
- Các đối tượng gồm biểu đồ, đồ thị, bản đồ, hình ảnh, bảng biểu,…



- **Phân tích mô tả (Descriptive analytics):** nhằm nắm bắt những gì đã hoặc đang xảy ra trong môi trường dữ liệu thông qua việc trực quan hóa dữ liệu
- **Phân tích chuẩn đoán (Descriptive analytics):** quá trình phân tích chuyên sâu hoặc chi tiết dữ liệu nhằm nhắm được nguyên nhân xảy ra. Kỹ thuật sử dụng như đào sâu, khám phá – khai phá dữ liệu và tương quan dữ liệu
- **Phân tích dự đoán (Descriptive analytics):** sử dụng dữ liệu trong quá khứ để dự báo chính xác về các xu hướng trong tương lai. Sử dụng các kỹ thuật như máy học, mô hình dự báo, so khớp mẫu,…
- **Phân tích đề xuất (Prescriptive analytics):** không chủ dự đoán mà còn đề xuất các giải pháp tối ưu thông qua việc phân tích các tác động tìm ẩn của các lựa chọn khác nhau và đề xuất xu hướng tốt nhất. Sử dụng phân tích đồ thị, mô phỏng, mạng neural, AI và các công cụ chuyên dụng

- Phân tích cụm (Cluster analysis)
- Phân tích theo nhóm (Cohort analysis)
- Phân tích hồi quy (Regression analysis)
- Mạng nơron (Neural network)
- Phân tích nhân tố (Factor analysis)
- Khai phá dữ liệu (Data mining)
- Phân tích văn bản (Text analysis)
- Phân tích chuỗi thời gian (Time series analysis)
- Cây quyết định (Decision Trees)
- Phân tích thuộc tính (Conjoint analysis)

- Một số công cụ phân tích dữ liệu
  - ✓ PowerBI: công cụ phân tích và trực quan hóa dữ liệu
  - ✓ Python: ngôn ngữ cung cấp các thư viện có khả năng làm sạch, chuyển đổi, phân tích và trực quan dữ liệu
  - ✓ Hadoop: lưu trữ và phân tích dữ liệu
  - ✓ MongoDB: sử dụng trên các tập dữ liệu thường xuyên thay đổi
  - ✓ Talend: tích hợp và quản lý dữ liệu
  - ✓ Cassandra: cơ sở dữ liệu phân tán được sử dụng để xử lý data chunk
  - ✓ Spark: xử lý thời gian thực và phân tích dữ liệu lớn
  - ✓ STORM: hệ thống tính toán thời gian thực mã nguồn mở
  - ✓ Kafka: nền tảng phân tán trực tuyến việc lưu trữ và có khả năng chịu lỗi cao

- Trong chương trước chúng ta đã tìm hiểu về list – một cấu trúc dữ liệu cho phép lưu trữ mảng. List đơn giản và có đầy đủ chức năng cơ bản của mảng. Tuy nhiên, khi chứa mảng số với nhiều phần tử, list có tốc độ thực thi và hỗ trợ tính toán không bằng numpy arrays – một cấu trúc dữ liệu dạng mảng của thư viện numpy.
- NumPy (phát âm là /ˈnʌmpaɪ/) viết tắt cụm từ **numerical python** là một thư viện cho Python, hỗ trợ làm việc với các mảng lớn, nhiều chiều, và cung cấp các hàm toán học cấp cao hoạt động trên các mảng này
- Thường được sử dụng tính toán đại số trên ma trận & vector

- Cài đặt numpy : **pip install numpy**
- Khai báo thư viện : **import numpy as np**

Cần cài đặt thư viện trước khi khai báo, nếu không khi thực thi lệnh import sẽ báo lỗi không tìm thấy thư viện

- **Tạo mảng**

arr = np.array([value1, value2,…], dtype = <datatype_name>)

Ví dụ:

Mảng 1 chiều: arr = np.array([1, 2, 3]) # [1, 2, 3]

Mảng 2 chiều: arr = np.array([(1, 2), (3, 4)]) # [[1, 2]

[3, 4]]

Mảng 3 chiều: arr = np.array(([[), ()], [(), ()], [(), ()]))

Lưu ý:

- **axis: 0|cột – 1|hàng**
- **shape: hàng x cột x …**

- **Hàm dựng sẵn**
  - ✓ np.zeros(shape) # ma trận toàn phần tử **0**
  - ✓ np.ones(shape) # ma trận toàn phần tử **1**
  - ✓ np.arange(start, stop, step) # ma trận $start \leq value < stop$
  - ✓ np.full(shape, fill_value) # ma trận toàn phần tử **fill_value**
  - ✓ np.random.seed(id_memory) # giữ nguyên kết quả random
  - ✓ np.random.randn(shape) # ma trận ngẫu nhiên
- **Ma trận vuông**
  - ✓ np.eye/identity(n) # ma trận đơn vị n
  - ✓ np.diag(arr) # ma trận đường chéo hoặc trả về đường chéo

- **Đọc và lưu numpy từ file**
  - ✓ np.save/load(file, arr) # lưu/đọc ma trận vào file .npy [.npz]
  - ✓ np.savez(file, arr1, arr2,…) # lưu nhiều ma trận vào file .npz
  - ✓ np.savetxt/loadtxt(file, arr) # lưu/đọc ma trận vào file text
- **Truy xuất ma trận | array[i][j] ~ array[i, j]**

Cú pháp:

indice_start : (indice_end + 1)

-num_indices: **#** lấy số lượng num_indices ở vị trí cuối cùng

:num_indices **#** lấy một số lượng num_indices ở vị trí đầu tiên

Cho đoạn code sau:

```
import numpy as np
X = np.array([[1, 2, 3, 4],
              [4, 5, 6, 2],
              [7, 8, 9, 1],
              [2, 5, 1, 5]])
print(X[1, 2])       # Truy cập vào phần tử thuộc dòng 1 và cột 2
print(X[1:3, 1:3])   # Truy cập vào các dòng từ 1 đến 2 và các cột từ 1 đến 2
print(X[:2, :2])     # Truy cập vào 2 dòng, 2 cột đầu tiên
print(X[-2:, -2:])   # Truy cập vào 2 dòng, 2 cột cuối cùng
print(X[[0, 2], :]) # Truy cập vào dòng 0 và 2. Cách lấy indices không liên tục
```


Cho đoạn code sau:

```
import numpy as np
X = np.array([[1, 2, 3, 4],
              [4, 5, 6, 2],
              [7, 8, 9, 1],
              [2, 5, 1, 5]])
```

Hãy dùng cách thức truy cập để lấy ra các giá trị sau:

a. **[[2, 3] [5, 6]]**

b. **[[1, 3] [4, 6] [7, 9]]**

c. **[1, 5, 9, 5]**

d. **[[1, 4] [4, 2]]**

e. **[[2, 3] [5, 6] [8, 9] [5, 1]]**

- **Chuyển vị các chiều**
  - ✓ np.transpose(array, (d_i, d_j, …, d_n))
- **Thay đổi shape của ma trận**
  - ✓ np.reshape(new_shape) hoặc arr.reshape(new_shape) với giá trị **-1** numpy sẽ tự tính chiều còn lại
- **Ghép ma trận**
  - ✓ np.concatenate((arr1, arr2, …, arrn), axis=position)
  - ✓ np.stack/vstack/hstack/dstack([arr1, arr2, …, arrn])
- **Tách ma trận**
  - ✓ arr1, arr2 = np.split/vsplit/hsplit/dsplitk(arr)
- **Mở rộng ma trận**
  - ✓ np.expand_dims(arr, axis=position)

Cho mảng B = np.array(np.arange(12))

Xem số phần tử của mỗi chiều ma trận

print("B.shape: ", B.shape)

Biến đổi số chiều của ma trận

B.reshape(2, 6) hoặc B.reshape(2, -1) hoặc np.reshape(B, (2, 6)) hoặc np.reshape(B, (2, -1))

a. **Phát sinh mảng A gồm 2352 phần tử ngẫu nhiên số nguyên không âm trong đoạn [1, 20]**
b. **Phát sinh mảng B gồm 2352 phần tử ngẫu nhiên số nguyên không dương trong đoạn [-20, -1]**
c. **Xem thông tin các chiều của mảng trên**
d. **Tạo ma trận C, D lần lượt từ mảng A, B về ma trận 3 chiều với (28, 28, 3)**
e. **Nối ma trận C, D theo dòng**
f. **Đặt 2 ma trận C, D lên nhau**
g. **Ghép 2 ma trận C, D theo chiều ngang, dọc, rộng**

- **Phép toán ma trận**
  - ✓ arr.T # phép chuyển vị m x n → n x m
  - ✓ np.linalg.pinv(arr) # ma trận nghịch đảo ma trận vuông
  - ✓ np.linalg.det(arr) # định thức của ma trận
  - ✓ np.trace(arr) # tổng giá trị trên đường chéo chính
  - ✓ np.linalg.norm(arr, ord='fro') # chuẩn Frobenious
  - ✓ np.dot(arr1, arr2) | arr1.dot(arr2) | A@B # nhân hai ma trận
  - ✓ arr * value # nhân ma trận với giá trị

- **Hàm trên numpy**
  - ✓ min/max/mean/sum([axis = position])
  - ✓ minimum/maximum(arr1, arr2, …, arrn)
  - ✓ argmax/argmin(arr, axis = position) # vị trí có giá trị max/min
  - ✓ argsort(+/-arr, axis = position) # vị trí có giá trị tăng/giảm dần
  - ✓ np.exp(arr) # tính số mũ cơ số tự nhiên e, $f(x) = e^x$
- **Giữ nguyên shape**
  - ✓ np.*functionX*(…, keepdims=True) # chiều tiêu giảm = 1

Cho 2 ma trận A = np.array([[1, 3, 6], [2, 7, 9]]) và B = np.array([[0, 4, 5], [1, 6, 10]])

a. Tìm phần tử nhỏ nhất trong A

b. Tìm phần tử nhỏ nhất của A theo từng dòng

c. Tìm phần tử nhỏ nhất của A theo từng cột

d. Tìm các phần tử maximum/minimum của A và B ở cùng vị trí indice

e. Tìm ra indice có giá trị lớn nhất của các dòng trong ma trận A

f. Tìm ra indice của các cột theo thứ tự tăng/giảm dần của ma trận A

Bài tập: Cho ma trận bên dưới là đầu ra có kích thước NxCtrong đó N là số quan sát và C là số classes. Mỗi dòng của ma trận là một phân phối xác suất của một quan sát. Thống kê kết quả nhãn thuộc về mỗi loại.

```
B = np.array([[0.1, 0.2, 0.7],
              [0.6, 0.4, 0.0],
              [0.1, 0.5, 0.4],
              [0.3, 0.3, 0.4],
              [0.1, 0.8, 0.1]])
```


- ✓ Họ tên : Trần Quang Khải
- ✓ Email : khaitq@hcmute.edu.vn

- ✓ Zalo (mã Qr)