

# Các phương pháp học máy

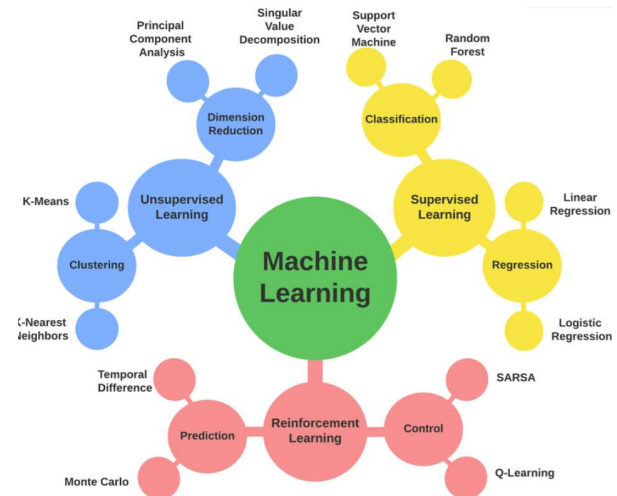
## Machine learning methods

4 TC: 2 LT – 2 TH

Giảng viên: **Tạ Hoàng Thắng**

[tahoangthang@gmail.com](mailto:tahoangthang@gmail.com)

0975399307

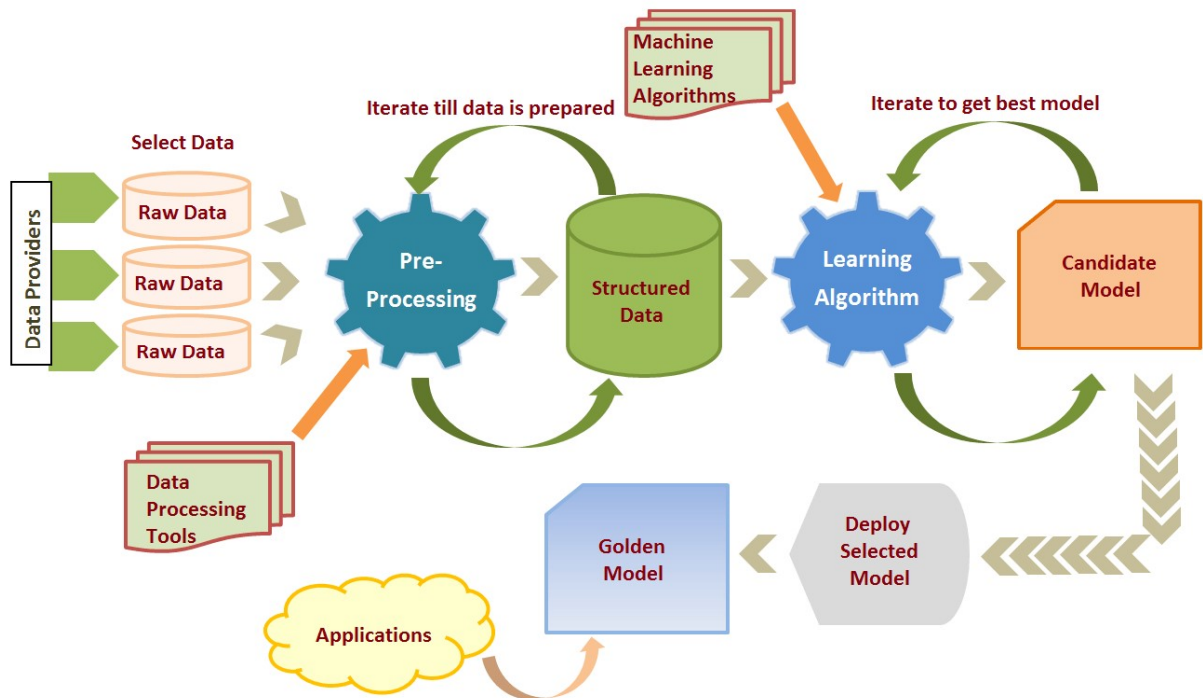


# Nhập nguồn dữ liệu

## Import data

There are several ways for importing data in machine learning using Python:

- Through URLs
- Integrated Libraries
- From Files



# Nhập nguồn dữ liệu

## Through URLs

- Use **requests** library (pip install requests)

```
import requests

# Define the URL
url = 'https://www.example.com'

# Send a GET request to the URL
response = requests.get(url)

# Check if the request was successful
if response.status_code == 200:
    # Print the content of the response
    print(response.text)
else:
    print(f"Failed to retrieve data. Status code: {response.status_code}")
```

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **sklearn.datasets**
  - <https://scikit-learn.org/stable/datasets.html>

4. Inspection	▼	<b><u>7.1. Toy datasets</u></b>
5. Visualizations		<a href="#"><u>7.1.1. Iris plants dataset</u></a>
6. Dataset transformations	▼	<a href="#"><u>7.1.2. Diabetes dataset</u></a>
7. Dataset loading utilities	^	<a href="#"><u>7.1.3. Optical recognition of handwritten digits dataset</u></a>
7.1. Toy datasets		<a href="#"><u>7.1.4. Linnerrud dataset</u></a>
7.2. Real world datasets		<a href="#"><u>7.1.5. Wine recognition dataset</u></a>
7.3. Generated datasets		<a href="#"><u>7.1.6. Breast cancer wisconsin (diagnostic) dataset</u></a>
7.4. Loading other datasets		
8. Computing with scikit-learn	▼	<b><u>7.2. Real world datasets</u></b>
9. Model persistence		<a href="#"><u>7.2.1. The Olivetti faces dataset</u></a>
10. Common pitfalls and recommended practices		<a href="#"><u>7.2.2. The 20 newsgroups text dataset</u></a>
11. Dispatching	▼	<a href="#"><u>7.2.3. The Labeled Faces in the Wild face recognition dataset</u></a>
12. Choosing the right estimator		<a href="#"><u>7.2.4. Forest covertypes</u></a>
		<a href="#"><u>7.2.5. RCV1 dataset</u></a>

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **sklearn.datasets**
  - Load dataset **iris**

**iris setosa**



petal      sepal

**iris versicolor**



petal      sepal

**iris virginica**



petal      sepal

# Nhập nguồn dữ liệu

**Integrated Libraries:** Use `sklearn.datasets`: Load dataset iris

```
1 from sklearn.datasets import load_iris
2 import pandas as pd
3
4 # Load the Iris dataset
5 iris = load_iris()
6
7 # Create a DataFrame with features and labels
8 iris_df = pd.DataFrame(iris.data, columns=iris.feature_names)
9 iris_df['species'] = iris.target
10
11 # Map numeric labels to species names
12 species_map = {i: species for i, species in enumerate(iris.target_names)}
13 iris_df['species'] = iris_df['species'].map(species_map)
14
15 # Display the DataFrame
16 print("Iris dataset with labels:")
17 print(iris_df.head())
18
19 #Iris dataset with labels:
20 #   sepal length (cm)  sepal width (cm)  ...  petal width (cm)  species
21 # 0                5.1                3.5  ...                0.2    setosa
22 # 1                4.9                3.0  ...                0.2    setosa
23 # 2                4.7                3.2  ...                0.2    setosa
24 # 3                4.6                3.1  ...                0.2    setosa
25 # 4                5.0                3.6  ...                0.2    setosa
26 # [5 rows x 5 columns]
```

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **sklearn.datasets**
  - Load dataset **wine**

```
1 from sklearn.datasets import load_wine
2 import pandas as pd
3
4 # Load the Wine dataset
5 wine = load_wine()
6
7 # Create a DataFrame for the feature data
8 wine_df = pd.DataFrame(wine.data, columns=wine.feature_names)
9
10 # Add the target labels to the DataFrame
11 wine_df['class'] = wine.target
12
13 # Map target labels to class names
14 wine_df['class'] = wine_df['class'].map({i: name for i, name in enumerate(wine.target_names)})
15
16 # Display the first few rows of the DataFrame
17 print(wine_df.head())
18
19 # alcohol  malic_acid  ash  alkalinity_of_ash  ...  hue  od280/od315_of_diluted_wines  proline  class
20 #0      14.23      1.71  2.43                15.6  ...  1.04                3.92    1065.0  class_0
21 #1      13.20      1.78  2.14                11.2  ...  1.05                3.40    1050.0  class_0
22 #2      13.16      2.36  2.67                18.6  ...  1.03                3.17    1185.0  class_0
23 #3      14.37      1.95  2.50                16.8  ...  0.86                3.45    1480.0  class_0
24 #4      13.24      2.59  2.87                21.0  ...  1.04                2.93     735.0  class_0
```



# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **seaborn** (*pip install seaborn*)
  - Load **iris**

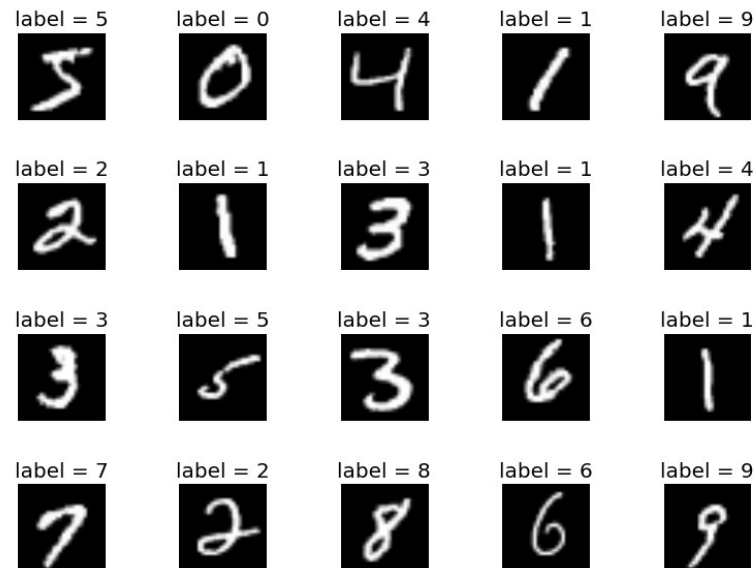
```
1 import seaborn as sns
2
3 # Load the 'iris' dataset
4 iris = sns.load_dataset('iris')
5 print(iris.head())
6
7 #    sepal_length  sepal_width  petal_length  petal_width  species
8 #0           5.1           3.5           1.4           0.2    setosa
9 #1           4.9           3.0           1.4           0.2    setosa
10 #2           4.7           3.2           1.3           0.2    setosa
11 #3           4.6           3.1           1.5           0.2    setosa
12 #4           5.0           3.6           1.4           0.2    setosa
```



# Nhập nguồn dữ liệu

## Integrated Libraries

- Use TensorFlow/PyTorch
  - Load **MNIST**



# Nhập nguồn dữ liệu

## Integrated Libraries

- Use TensorFlow/PyTorch (pip install torch, pip install tensorflow)
  - Load MNIST

```
1 import torch
2 from torchvision import datasets, transforms
3
4 # Define a transform to normalize the data
5 transform = transforms.Compose([
6     transforms.ToTensor(), # Convert image to tensor
7     transforms.Normalize((0.5,), (0.5,)) # Normalize with mean=0.5 and std=0.5
8 ])
9
10 # Download and load the training data
11 train_dataset = datasets.MNIST(root='data', train=True, download=True, transform=transform)
12 train_loader = torch.utils.data.DataLoader(train_dataset, batch_size=64, shuffle=True)
13
14 # Download and load the test data
15 test_dataset = datasets.MNIST(root='data', train=False, download=True, transform=transform)
16 test_loader = torch.utils.data.DataLoader(test_dataset, batch_size=64, shuffle=False)
```

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use TensorFlow/PyTorch
  - Load MNIST

```
17
18 # Check the size of the dataset
19 print(f'Training dataset size: {len(train_dataset)}')
20 print(f'Test dataset size: {len(test_dataset)}')
21
22 # Example: Accessing a single batch of data
23 data_iter = iter(train_loader)
24 images, labels = next(data_iter)
25
26 print(f'Batch of images shape: {images.shape}')
27 print(f'Batch of labels shape: {labels.shape}')
28 print(f'Example label: {labels[0].item()}')
29
```

```
Training dataset size: 60000
Test dataset size: 10000
Batch of images shape: torch.Size([64, 1, 28, 28])
Batch of labels shape: torch.Size([64])
Example label: 1
```

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **datasets** (*pip install datasets*, <https://huggingface.co/datasets>)

The screenshot displays the Hugging Face Datasets page. The top navigation bar includes the Hugging Face logo, a search bar, and links for Models, Datasets, Spaces, Posts, Docs, Solutions, Pricing, Log In, and Sign Up. The left sidebar contains a 'Main' tab and various filters for Modalities (3D, Audio, Geospatial, Image, Tabular, Text, Time-series, Video), Size (rows) (a slider from <1K to >1T), and Format (json, csv, parquet, imagefolder, soundfolder, webdataset, text, arrow). The main content area shows a list of datasets with 195,121 total results. The list includes datasets like 'hails/mmlu\_no\_train', 'lighteval/mmlu', 'argilla/databricks-dolly-15k-curated-en', 'lavita/medical-qa-shared-task-v1-toy', 'ceval/ceval-exam', 'SaylorTwift/bbh', 'lukaemon/bbh', 'cais/mmlu', 'HAERAE-HUB/KMMLU', and 'EleutherAI/hendrycks\_math'. Each dataset entry shows its name, update date, size, and download count.

Dataset Name	Updated	Size	Downloads
hails/mmlu_no_train	Jan 23	23.8M	13
lighteval/mmlu	Jun 9, 2023	5.82M	16.9M
argilla/databricks-dolly-15k-curated-en	Oct 2, 2023	15k	4.1M
lavita/medical-qa-shared-task-v1-toy	Jul 20, 2023	64	1.56M
ceval/ceval-exam	Aug 31, 2023	13.9k	1.29M
SaylorTwift/bbh	Jun 16	6.76k	896k
lukaemon/bbh	Feb 2, 2023	6.51k	895k
cais/mmlu	Mar 9	231k	675k
HAERAE-HUB/KMMLU	Mar 5	244k	662k
EleutherAI/hendrycks_math	Nov 2, 2023	639k	6

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **datasets** (*pip install datasets*)
  - <https://huggingface.co/datasets>
- Tìm hiểu **CIFAR10** và **dog-food** bằng <https://huggingface.co/datasets>
  - Số lượng dữ liệu ở tập train/test/validation?
  - Số nhãn (labels)?
  - Mô tả bí dụ?
  - Kích thước của 1 input?



# Hugging Face

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **datasets**
  - Load **IDBM**

```
1 from datasets import load_dataset
2
3 # Load the IMDB dataset
4 dataset = load_dataset('imdb')
5
6 # Display the dataset structure
7 print(dataset)
8
9 # Access the training and test splits
10 train_dataset = dataset['train']
11 test_dataset = dataset['test']
12
13 # Display the first few examples from the training set
14 print(train_dataset[0])
15
16 # Display the first few examples from the test set
17 print(test_dataset[0])
18
```

# Nhập nguồn dữ liệu

## Integrated Libraries

- Use **datasets**
  - Load **IDBM**

```
DatasetDict({
  train: Dataset({
    features: ['text', 'label'],
    num_rows: 25000
  })
  test: Dataset({
    features: ['text', 'label'],
    num_rows: 25000
  })
  unsupervised: Dataset({
    features: ['text', 'label'],
    num_rows: 50000
  })
})
```

```
{'text': 'I rented I AM CURIOUS-YELLOW  
from my video store because of all the  
controversy that surrounded it when it  
was first released in 1967. I also heard  
that at first it was seized by U.S.  
customs if it ever tried to enter this  
country, therefore being a fan of films  
considered "controversial" I really had  
to see this for myself..|.', 'label': 0}
```



# Nhập nguồn dữ liệu

## From Files

- CSV: example.csv + pandas

Name, Age, City

Alice, 30, New York

Bob, 25, Los Angeles

Charlie, 35, Chicago

David, 40, San Francisco

```
1 import pandas as pd
2
3 # Load the CSV file into a DataFrame
4 df = pd.read_csv('example.csv')
5
6 # Display the DataFrame
7 print("DataFrame:")
8 print(df)
```

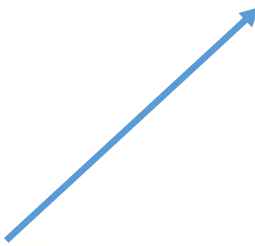
```
DataFrame:
   Name  Age  City
0  Alice  30  New York
1   Bob   25  Los Angeles
2 Charlie  35   Chicago
3  David  40  San Francisco
```

# Nhập nguồn dữ liệu

## From Files

- CSV: example.csv + pandas

```
6 # View the first few rows
7 print("\nFirst few rows:")
8 print(df.head())
9
10 # View the last few rows
11 print("\nLast few rows:")
12 print(df.tail())
13
14 # Get DataFrame information
15 print("\nDataFrame info:")
16 print(df.info())
```



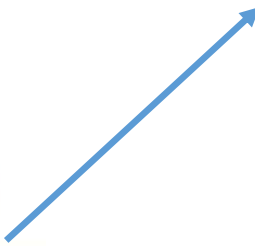
```
DataFrame info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 3 columns):
#   Column  Non-Null Count  Dtype
---  -
0   Name    4 non-null        object
1   Age     4 non-null        int64
2   City    4 non-null        object
dtypes: int64(1), object(2)
memory usage: 224.0+ bytes
None
```

# Nhập nguồn dữ liệu

## From Files

- CSV: example.csv + pandas

```
6 # View the first few rows
7 print("\nFirst few rows:")
8 print(df.head())
9
10 # View the last few rows
11 print("\nLast few rows:")
12 print(df.tail())
13
14 # Get DataFrame information
15 print("\nDataFrame info:")
16 print(df.info())
```



```
DataFrame info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 3 columns):
#   Column  Non-Null Count  Dtype
---  -
0   Name     4 non-null        object
1   Age      4 non-null        int64
2   City     4 non-null        object
dtypes: int64(1), object(2)
memory usage: 224.0+ bytes
None
```

# Nhập nguồn dữ liệu

## From Files

- CSV: example.csv + csv

```
1 import csv
2
3 # Initialize an empty list to store the data
4 data = []
5
6 # Open and read the CSV file
7 with open('example.csv', 'r') as file:
8     reader = csv.DictReader(file) # Read CSV into a dictionary format
9     for row in reader:
10         data.append(row)
11
12 # Display the data
13 print("Data loaded as list of dictionaries:")
14 print(data)
```

```
E:\Sammi\ML>python test.py
Data loaded as list of dictionaries:
[{'Name': 'Alice', 'Age': '30', 'City': 'New York'}, {'Name': 'Bob', 'Age': '25', 'City': 'Los Angeles'}, {'Name': 'Charlie', 'Age': '35', 'City': 'Chicago'}, {'Name': 'David', 'Age': '40', 'City': 'San Francisco'}]
```

# Nhập nguồn dữ liệu

## From Files

- TSV: example.tsv + pandas

Name	Age	City
Alice	30	New York
Bob	25	Los Angeles
Charlie	35	Chicago
David	40	San Francisco

```
1 import pandas as pd
2
3 # Load the CSV file into a DataFrame
4 df = pd.read_csv('example.tsv', delimiter='\t')
5
6
```

# Nhập nguồn dữ liệu

## From Files

- TSV: example.tsv + csv

```
1 import csv
2
3 # Initialize an empty list to store the data
4 data = []
5
6 # Open and read the TSV file
7 with open('example.tsv', 'r') as file:
8     reader = csv.DictReader(file, delimiter='\t')
9     for row in reader:
10         data.append(row)
11
12 # Display the data
13 print("Data loaded as list of dictionaries:")
14 print(data)
15
```

```
E:\Sammi\ML>python test.py
Data loaded as list of dictionaries:
[{'Name': 'Alice', 'Age': '30', 'City': 'New York'}, {'Name': 'Bob', 'Age': '25', 'City': 'Los Angeles'}, {'Name': 'Charlie', 'Age': '35', 'City': 'Chicago'}, {'Name': 'David', 'Age': '40', 'City': 'San Francisco'}]
```

# Nhập nguồn dữ liệu

## From Files

- JSON: example.json + pandas

```
1  [
2      {"Name": "Alice", "Age": 30, "City": "New York"},
3      {"Name": "Bob", "Age": 25, "City": "Los Angeles"},
4      {"Name": "Charlie", "Age": 35, "City": "Chicago"},
5      {"Name": "David", "Age": 40, "City": "San Francisco"}
6  ]
```

```
1  import pandas as pd
2
3  # Load the JSON file into a DataFrame
4  df = pd.read_json('example.json')
5
6  # Display the DataFrame
7  print("DataFrame:")
8  print(df)
```



# Nhập nguồn dữ liệu

## From Files

- JSON: example.json + json

```
1  [
2    {"Name": "Alice", "Age": 30, "City": "New York"},
3    {"Name": "Bob", "Age": 25, "City": "Los Angeles"},
4    {"Name": "Charlie", "Age": 35, "City": "Chicago"},
5    {"Name": "David", "Age": 40, "City": "San Francisco"}
6  ]
```

```
1  import json
2
3  # Load the JSON file
4  with open('example.json', 'r') as file:
5      data = json.load(file)
6
7  # Display the data
8  print("Loaded data:")
9  print(data)
```

# Nhập nguồn dữ liệu

## From Files

- JSONL: example.json + pandas

```
1 {"Name": "Alice", "Age": 30, "City": "New York"}
2 {"Name": "Bob", "Age": 25, "City": "Los Angeles"}
3 {"Name": "Charlie", "Age": 35, "City": "Chicago"}
4 {"Name": "David", "Age": 40, "City": "San Francisco"}
```

```
1 import pandas as pd
2
3 # Load the JSONL file into a DataFrame
4 df = pd.read_json('example.jsonl', lines=True)
5
6 # Display the DataFrame
7 print("DataFrame:")
8 print(df)
```

# Nhập nguồn dữ liệu

## From Files

- JSONL: example.jsonl + json

```
1 {"Name": "Alice", "Age": 30, "City": "New York"}
2 {"Name": "Bob", "Age": 25, "City": "Los Angeles"}
3 {"Name": "Charlie", "Age": 35, "City": "Chicago"}
4 {"Name": "David", "Age": 40, "City": "San Francisco"}

1 import json
2
3 # Initialize an empty list to store the data
4 data = []
5
6 # Open and read the JSONL file
7 with open('example.jsonl', 'r') as file:
8     for line in file:
9         # Parse each line as a JSON object and append it to the list
10        data.append(json.loads(line))
11
12 # Display the loaded data
13 print("Loaded data:")
14 print(data)
```

# Xuất nguồn dữ liệu

## From Files

- CSV/TSV/JSON/JSONL + pandas

```
1 import pandas as pd
2
3 # Sample DataFrame
4 df = pd.DataFrame({
5     'Name': ['Alice', 'Bob', 'Charlie', 'David'],
6     'Age': [30, 25, 35, 40],
7     'City': ['New York', 'Los Angeles', 'Chicago', 'San Francisco']
8 })
9
10 # Write to CSV
11 df.to_csv('example.csv', index=False)
12
13 # Write to TSV
14 df.to_csv('example.tsv', sep='\t', index=False)
15
16 # Write to JSON
17 df.to_json('example.json', orient='records', lines=False)
18
19 # Write to JSON Lines (JSONL)
20 df.to_json('example.jsonl', orient='records', lines=True)
21
22 print("Data successfully written to files.")
```

# Xuất nguồn dữ liệu

## From Files

- CSV/TSV/JSON/JSONL + csv & json

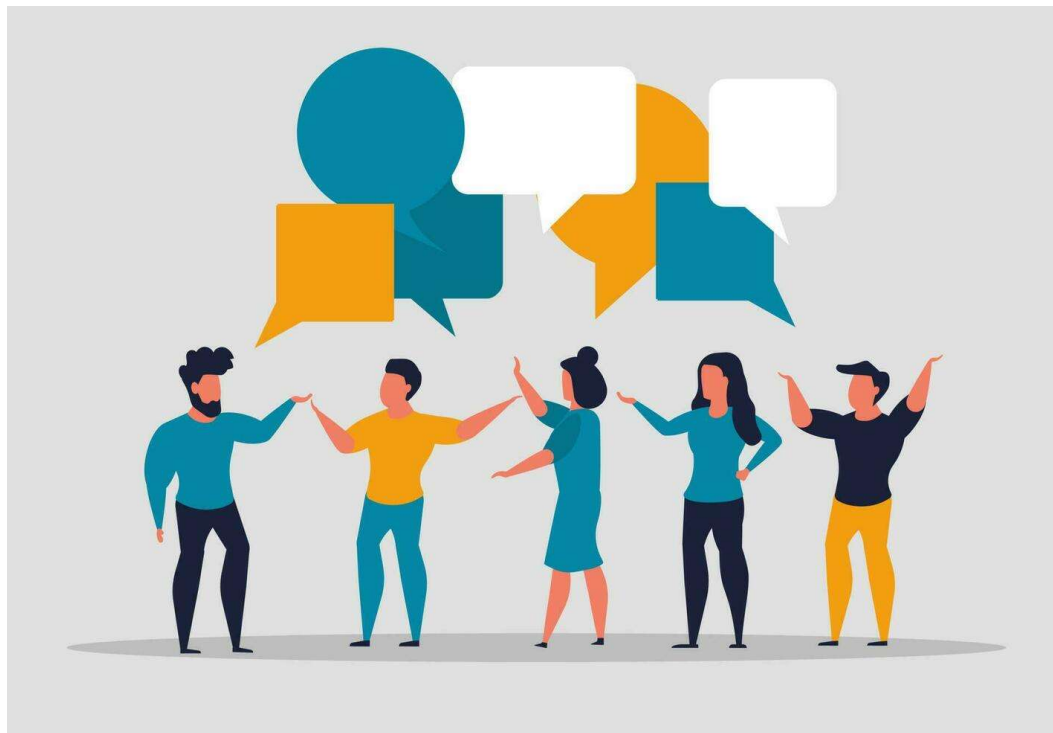
```
1 import csv
2
3 # Write data to a CSV or TSV file
4 with open('example.csv', 'w', newline='') as file:
5     writer = csv.DictWriter(file, fieldnames=["Name", "Age", "City"])
6     #writer = csv.DictWriter(file, fieldnames=["Name", "Age", "City"], delimiter='\t')
7     writer.writeheader()
8     writer.writerows(data)
```

```
1 import json
2
3 # Write data to a JSON file
4 with open('example.json', 'w') as file:
5     json.dump(data, file, indent=4) # indent=4 for pretty-printing
6
7 # Write data to a JSON Lines (JSONL) file
8 with open('example.jsonl', 'w') as file:
9     for record in data:
10         json.dump(record, file)
11         file.write('\n') # Write a newline character after each JSON object
```

# Thảo luận

## CSV vs. TSV vs. JSON vs. JSONL

- Which one? Why?



TẠ HOÀNG THẮNG - Đại học Đà Lạt, Khoa CNTT