ĐẠI NAM
UNIVERSITY

Thực hành

Bài 2: XỬ LÝ DỮ LIỆU VỚI PYTHON

Giảng viên: TS. Nguyễn Ngọc Giang

SĐT: 0862411011

Email: giangnn.cntt@dainam.edu.vn

Nội dung

- Nạp dữ liệu
- Làm sạch dữ liệu
- Phân tích khai phá dữ liệu
- Biến đổi dữ liệu

Nạp dữ liệu: Các nguồn dữ liệu

- Từ file CSV, JSON
- Từ SQL databases
- Từ NoSQL databases
- Từ APIs, Cloud data sources

Nạp dữ liệu từ file CSV

- **CSV -** (comma-separated values) bao gồm các cột dữ liệu được phân cách với nhau bởi dấu phẩy
- **Ví dụ:** *pandas - iris*

Code

```
import pandas as pd
filepath = 'data/iris_data.csv'

# Import the data
data = pd.read_csv(filepath)

# Print a few rows
print(data.iloc[:5])
```

Output

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Nạp dữ liệu từ file CSV: Một số tham số quan trọng

Code

```
# Different delimiters - tab-separated file (.tsv):
data = pd.read_csv(filepath, sep='\t')

# Different delimiters - space-separated file:
data = pd.read_csv(filepath, delim_whitespace=True)

# Don't use first row for column names:
data = pd.read_csv(filepath, header=None)

# Specify column names:
data = pd.read_csv(filepath, names=['Name1', 'Name2'])

# Custom missing values:
data = pd.read_csv(filepath, na_values=['NA', 99])
```

Nạp dữ liệu từ file JSON

- **JSON** - (JavaScript Object Notation) là một định dạng dữ liệu dựa trên văn bản (text-based data format) để trao đổi dữ liệu giữa các ứng dụng.
- Các file JSON có cấu trúc khá tương đồng với dữ liệu kiểu từ điển trong Python.

Code

```
# Read JSON file as dataframe
data = pd.read_json(filepath)

# Write dataframe file to JSON
data.to_json('outputfile.json')
```

Example (with one entry)

```
[{"id": 7267, "name": "10pin Bowling
Lounge", "address": "330 N State
Street", "city": "Chicago", "state":
"IL", "area": "Chicago / Illinois",
"postal_code": "60610", "country":
"US", "phone": "3126440300x", "lat":
41.888634, "lng": -87.628091, "price":
4}]
```

IBM

Cơ sở dữ liệu dạng SQL

- **SQL** - (Structured Query Language) Cơ sở dữ liệu quan hệ lưu trữ thông tin dưới dạng bảng có các hàng và cột đại diện cho những thuộc tính dữ liệu và nhiều mối quan hệ khác nhau giữa các giá trị dữ liệu.
- Có nhiều loại cơ sở dữ liệu SQL khác nhau với chức năng tương tự nhau. Ví dụ: Microsoft SQL Server, Postgres, MySQL, AWS Redshift, Oracle DB, Db2 Family

Code

```
# SQL Data Imports
import sqlite3 as sq3
import pandas as pd

# Initialize path to SQLite database
path = 'data/classic_rock.db'

# Create connection SQL database
con = sq3.Connection(path)

# Write query
query = ''' SELECT * FROM rock_songs;
'''
# Execute query
data = pd.read_sql(query, con)
```

Cơ sở dữ liệu dạng NoSQL

- **NoSQL -** (Not-only SQL) là một hệ thống quản lý dữ liệu không quan hệ sở hữu lược đồ linh hoạt.
- Nhiều CSDL NoSQL lưu trữ dữ liệu dạng file JSON
- **Ví dụ:**
  - Document databases: MongoDB, CouchDB
  - Key-value stores: Riak, Voldemort, Redis
  - Graph databases: Neo4j, HyperGraph
  - Wide-column stores: Cassandra, HBase

Code

```
# SQL Data Imports
from pymongo import MongoClient

# Create a Mongo connection
con = MongoClient()

# Choose database (con.list_database_names()
# will display available databases)
db = con.database_name

# Create a cursor object using a query
cursor = db.collection_name.find(query)

# Expand cursor and construct DataFrame
df = pd.DataFrame(list(cursor))
```

Nạp dữ liệu từ APIs hoặc Cloud Data

Code

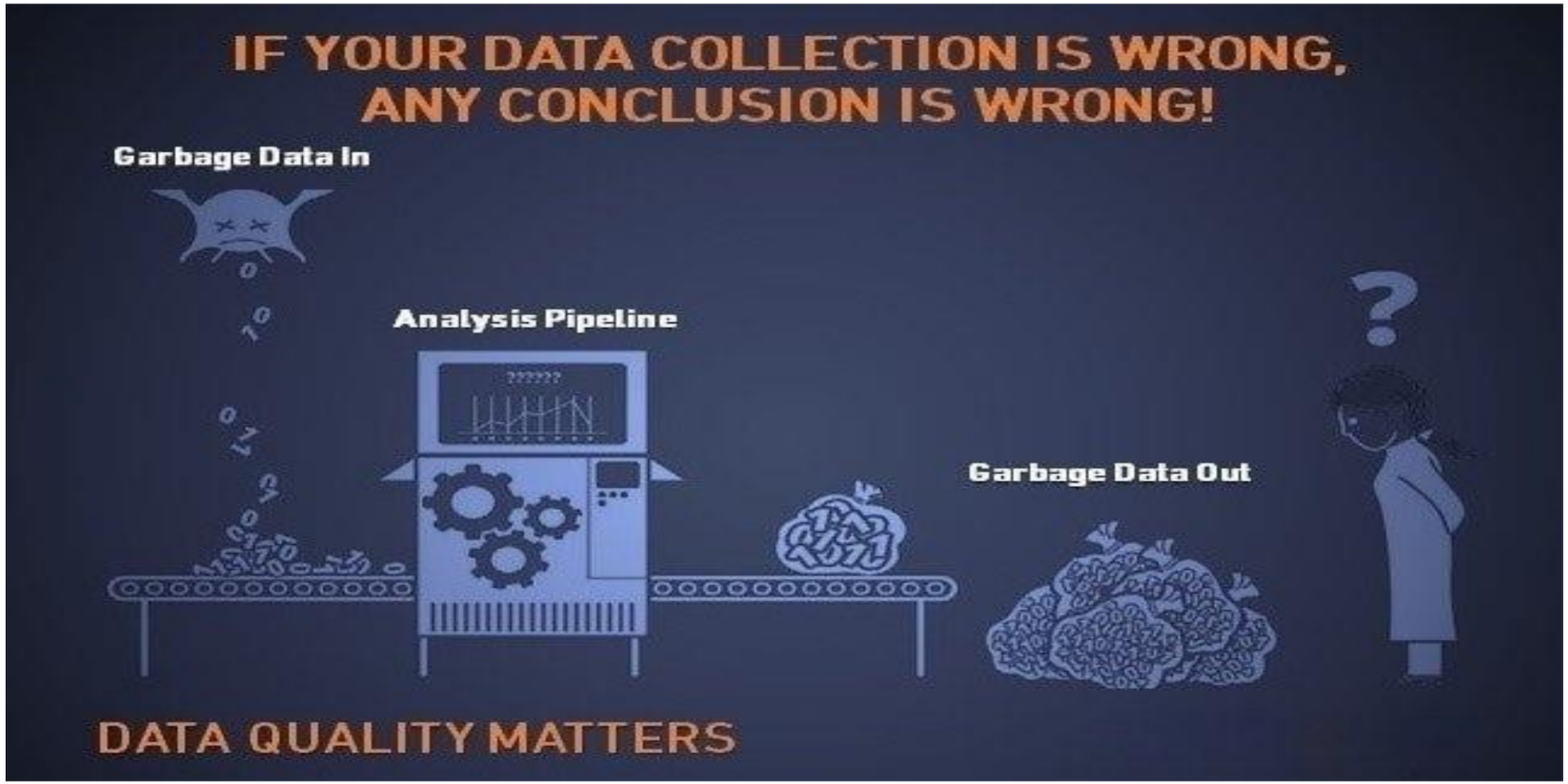
```
#UCI Cars data set - url location
data_url =
'http://archive.ics.uci.edu/ml/machine
-learning-databases/autos/imports-
85.data'

#Read data into Pandas
df=pd.read_csv(data_url, header=None)
```

Làm sạch dữ liệu

- Tầm quan trọng của việc làm sạch dữ liệu
- Các vấn đề của dữ liệu không sạch
- Cách xác định dữ liệu trùng lặp và dữ liệu không cần thiết
- Cách xử lý khi dữ liệu bị thiếu
- Cách xử lý dữ liệu ngoại lệ

Tầm quan trọng của việc làm sạch dữ liệu

- Garbage-in -- Garbage-out



Một số vấn đề của dữ liệu

- Dư thừa dữ liệu
  - Dữ liệu bị trùng lặp
  - Dữ liệu không cần thiết
- Dữ liệu bị thiếu
- Dữ liệu ngoại lệ (outliers)

Dữ liệu bị trùng lặp hoặc không cần thiết

- Tập trung vào các giá trị bị trùng lặp, tìm hiểu nguyên nhân gây ra trùng lặp
- Có thể lọc bỏ dữ liệu bị trùng lặp và dữ liệu không cần thiết (cần chú ý cẩn thận bởi dữ liệu đó có thể trở nên hữu ích trong tương lai)

Dữ liệu bị thiếu

- **Có 3 kiểu dữ liệu bị thiếu**:
  - Dữ liệu bị thiếu ***hoàn toàn ngẫu nhiên***: dữ liệu thiếu không phải do lỗi hệ thống hay người thu thập dữ liệu mà do
  - Dữ liệu bị thiếu ***ngẫu nhiên***
  - Dữ liệu bị thiếu ***không ngẫu nhiên***
- **Cách xử lý dữ liệu bị thiếu**:
  - ***Loại bỏ***: Loại bỏ hoàn toàn các dòng có dữ liệu bị thiếu
  - ***Sinh dữ liệu***: tự sinh dữ liệu để bù đắp vào vị trí dữ liệu bị thiếu. Ví dụ có thể sử dụng các giá trị phổ biến, giá trị trung bình, v.v để điền vào vị trị dữ liệu bị thiếu.
  - ***Đánh dấu vị trí dữ liệu bị thiếu***: tạo ra một thuộc tính dạng category đánh dấu cho các mẫu có dữ liệu bị thiếu.

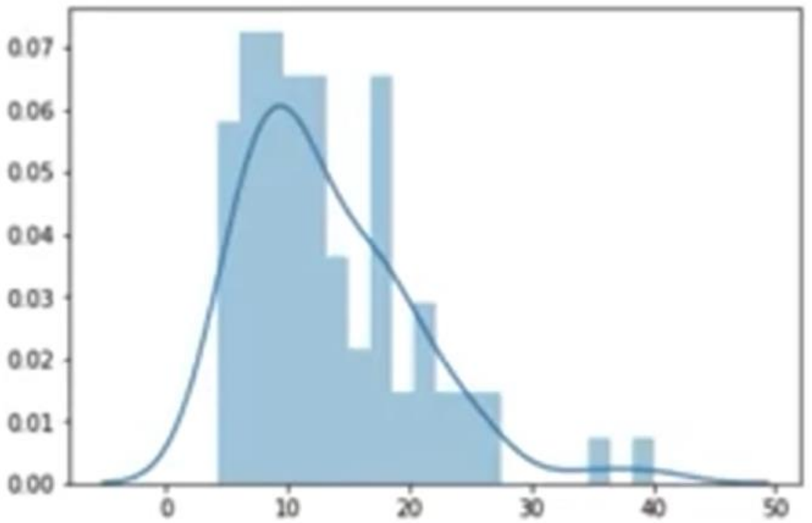
Dữ liệu ngoại lệ

- Dữ liệu ngoại lệ là những giá trị dữ liệu được ghi nhận có sự khác biệt bất thường so với những giá trị dữ liệu khác, không theo một quy tắc chung nào và có thể gây ra sai lệch trong kết quả phân tích và việc xây dựng các mô hình dự đoán.
- Tác động của dữ liệu ngoại lệ:
  - ➢ Làm sai lệch và tạo ra thiên kiến tiêu cực cho toàn bộ kết quả của một phân tích
  - ➢ Trong một số trường hợp đặc biệt, các giá trị ngoại lệ có thể cung cấp một thông tin giá trị cho kết quả phân tích
  - ➢ Loại bỏ dữ liệu ngoại lệ sẽ giúp kết quả phân tích chính xác và sát với thực tế hơn

Cách nhận biết dữ liệu ngoại lệ

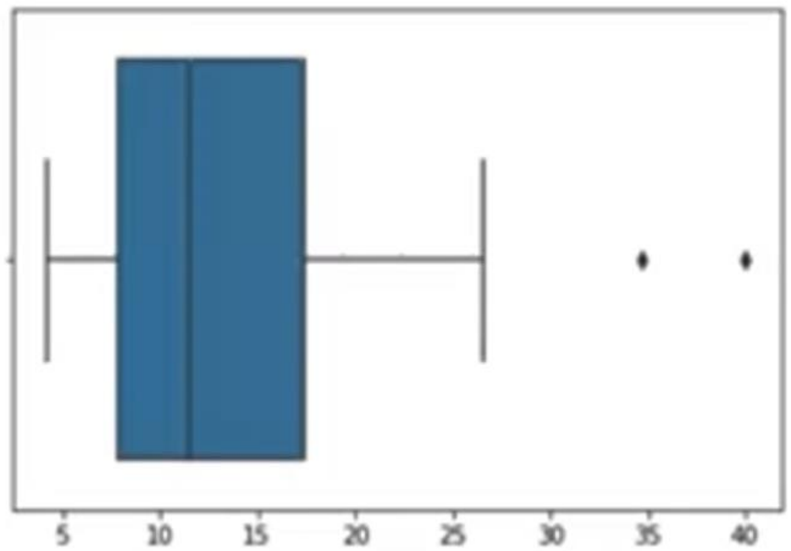
- Sử dụng biểu đồ

```
# plot a histogram and density plot
sns.distplot(data, bins=20);
```



Histogram

```
# plot a boxplot
sns.boxplot(data);
```



Box Plot

Cách nhận biết dữ liệu ngoại lệ

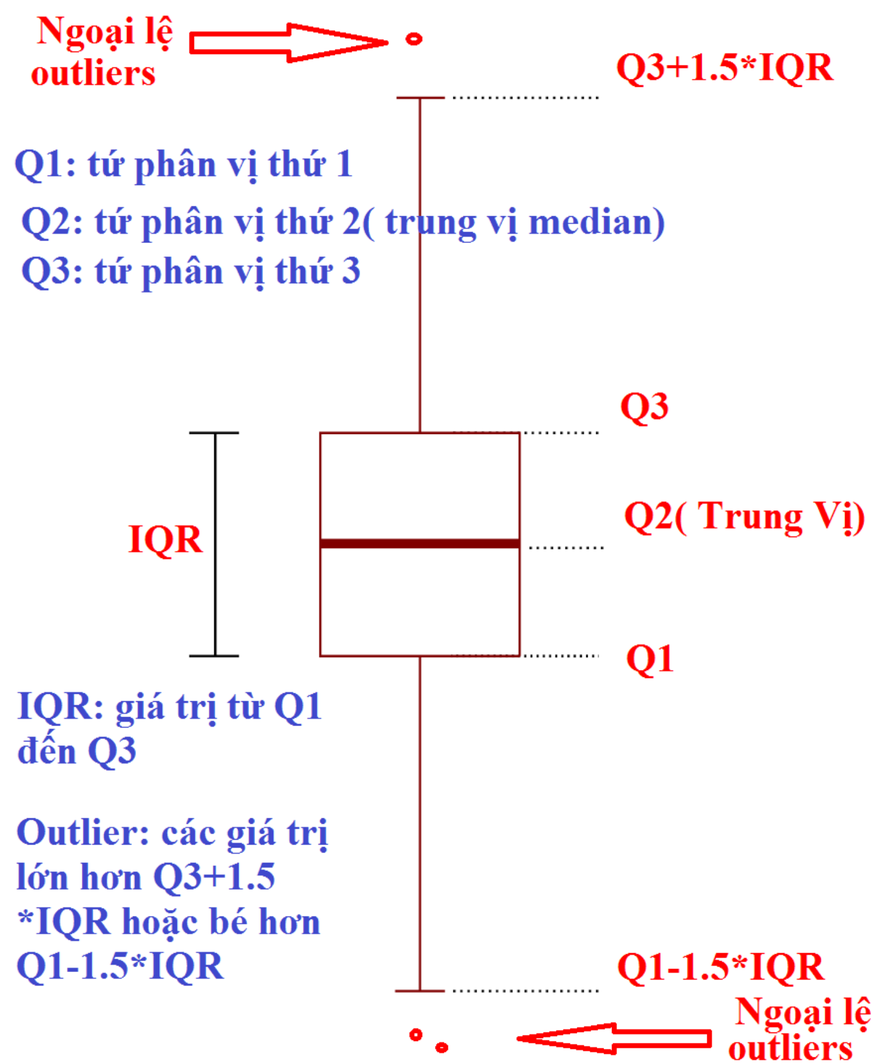
- Sử dụng các phương pháp thống kê:
  - Interquartile Range (IQR)

```
import numpy as np

# calculate the interquartile range
q25, q50, q75 = np.percentile(data, [25, 50, 75])
iqr = q75 - q25

# calculate the min / max limits to be considered an outlier
min = q25 - 1.5*(iqr)
max = q75 + 1.5*(iqr)

print(min, q25, q50, q75, max)
```

```
-6.6 7.8 11.5 17.4 31.8
```

```
# identify the points
[x for x in data['Unemployment'] if x > max]
```

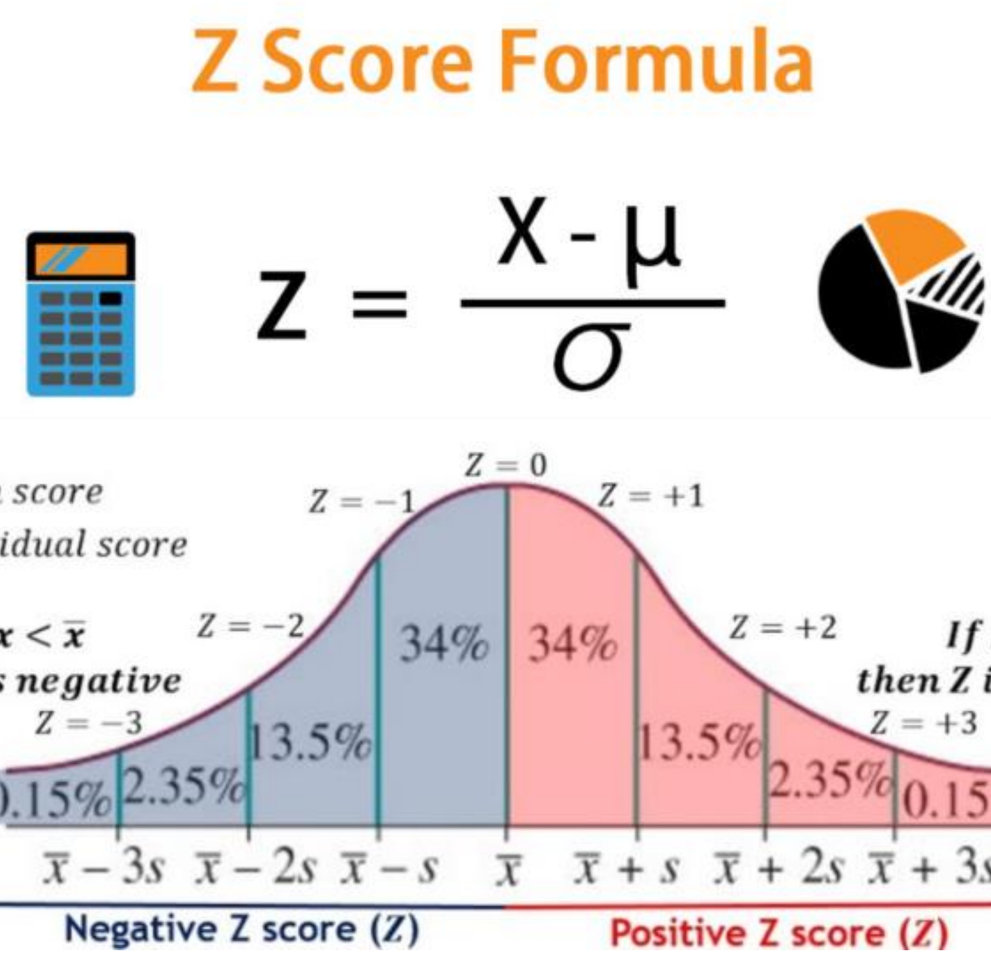
```
[40.0, 34.700000000000003]
```



Cách nhận biết dữ liệu ngoại lệ

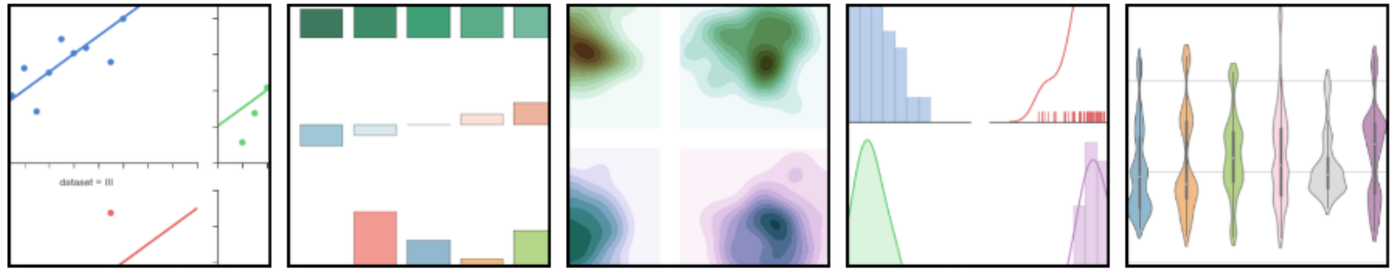
- Sử dụng các phương pháp thống kê:
  - Z-Score

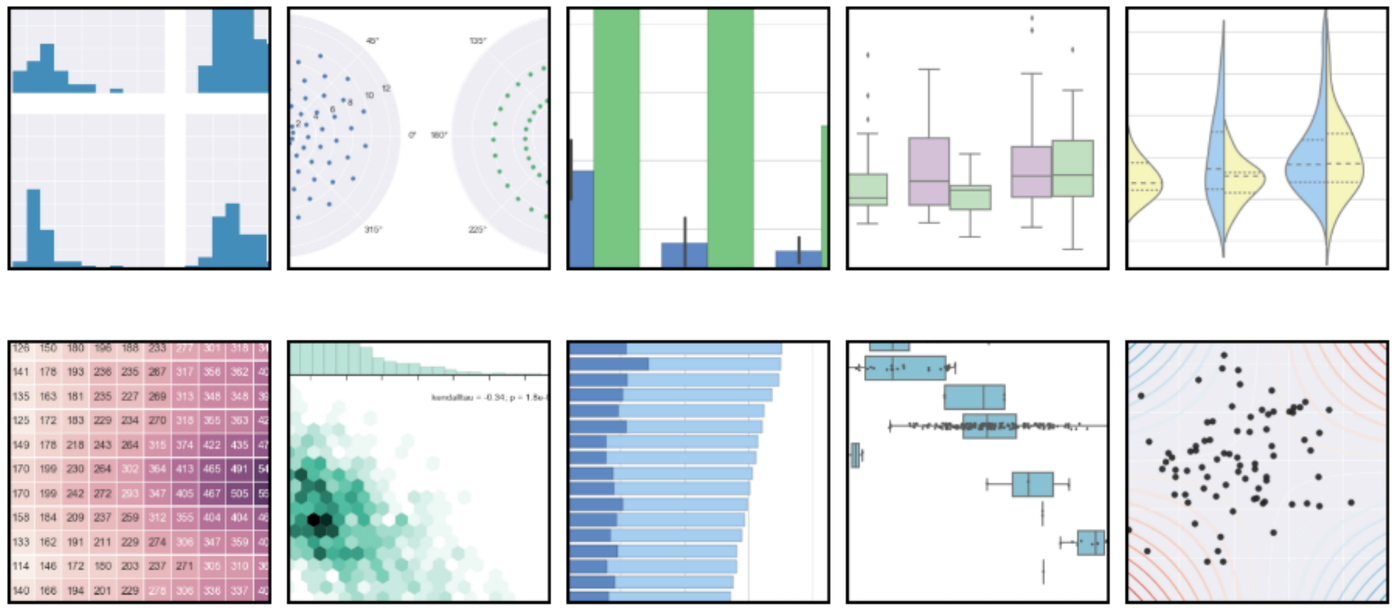
```
# Extract the column of interest for outlier detection (e.g., sepal
length)
column_name = "sepal_length"
data = iris_df[column_name]

# Calculate the Z-scores
z_scores = (data - data.mean()) / data.std()

# Define a threshold for identifying outliers (e.g., Z-score threshold
of 2)
threshold = 2

# Identify the outliers
outliers = iris_df[abs(z_scores) > threshold]
```



Phân tích khai phá dữ liệu - EDA

- EDA (Exploratory data analysis) là một hướng phân tích dữ liệu với mục đích là tìm ra được các đặc tính chính, quan trọng của dữ liệu.
- Thường sử dụng các đồ thị
  - ➢ Cho chúng ta cái nhìn và cảm nhận đầu tiên về dữ liệu
  - ➢ Có thể đưa ra các quyết định về dữ liệu như: dữ liệu đã có thể sử dụng được chưa, có cần thiết xử lý thêm không, có cần thu thập thêm dữ liệu không?
  - ➢ Giúp xác định các mẫu hình và xu hướng trong dữ liệu





Các kỹ thuật và công cụ sử dung trong EDA

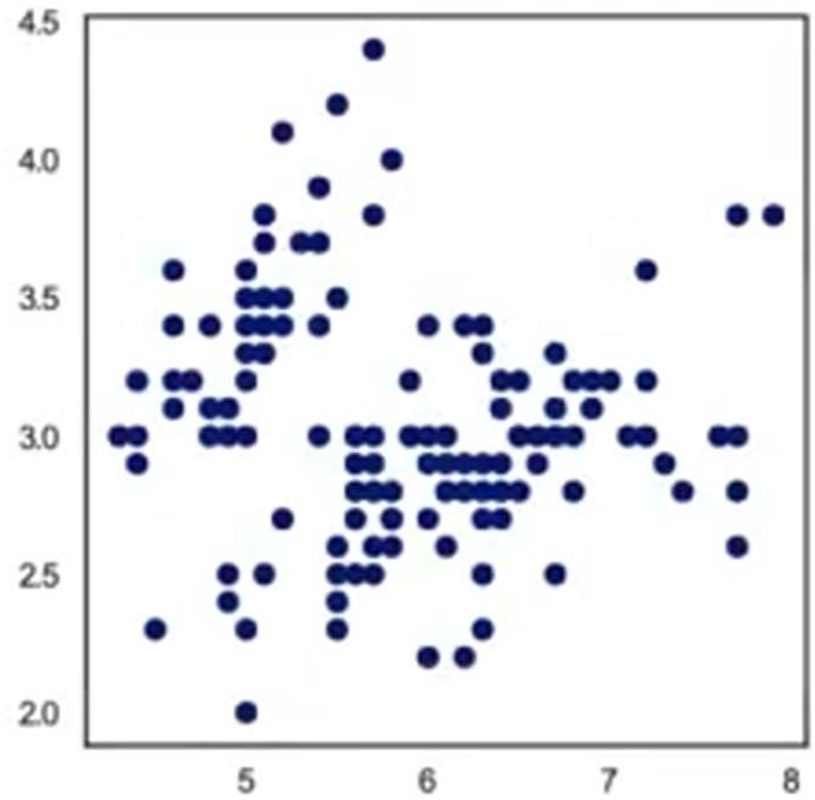
- **Thống kê tổng hợp**:
  - **Kỹ thuật**: Average, Median, Min, Max, Correlations, etc.
  - **Công cụ**: Pandas
- **Hình ảnh hóa**:
  - **Kỹ thuật**: Histograms, Scatter Plots, Box Plots, etc.
  - **Công cụ**: Matplotlib, Seaborn

Biểu đồ phân tán với Matplotlib

Code

```
# Pandas DataFrame approach
import matplotlib.pyplot as plt
plt.plot(data.sepal_length,
         data.sepal_width,
         ls ='', marker='o')
```

Output

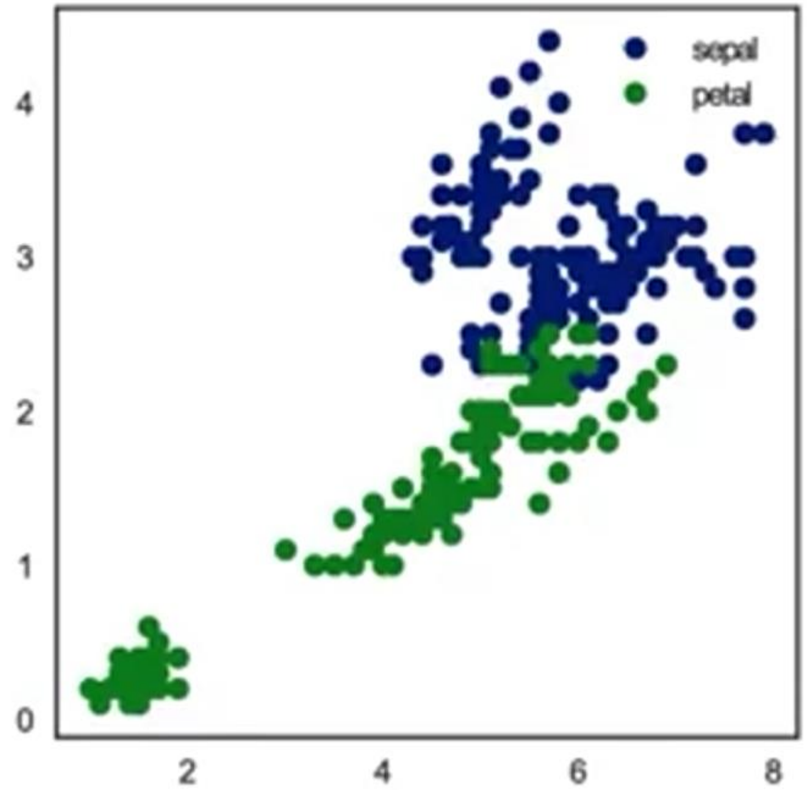


Biểu đồ phân tán với nhiều lớp

Code

```
# First plot statement
plt.plot(data.sepal_length,
         data.sepal_width,
         ls ='', marker='o',
         label='sepal')

# Second plot statement
plt.plot(data.petal_length,
         data.petal_width,
         ls ='', marker='o',
         label='petal')
```

Output

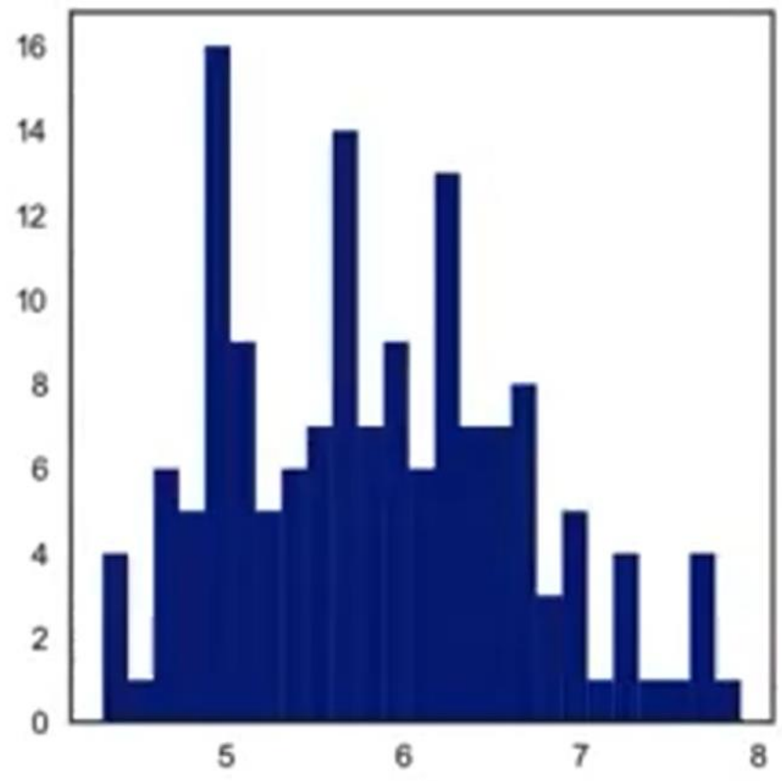


Biểu đồ Histogram với Matplotlib

Code

```
# Pandas DataFrame approach
plt.hist(data.sepal_length, bins=25)
```

Output

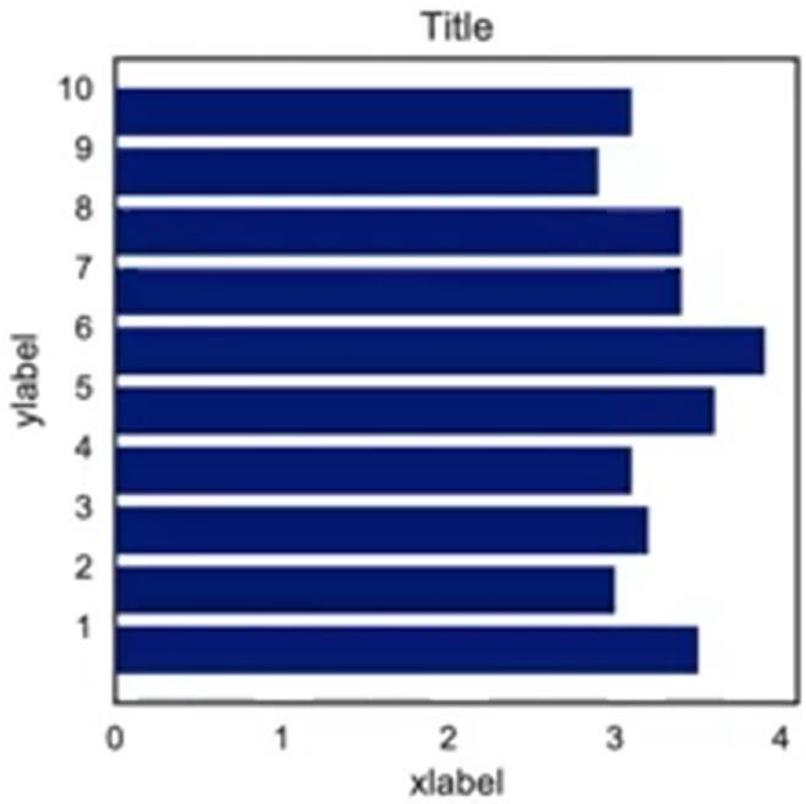


Một số tùy biến

Code

```
# matplotlib syntax
fig, ax = plt.subplots()
ax.barh(np.arange(10),
        data.sepal_width.iloc[:10])

# Set position of ticks and tick labels
ax.set_yticks(np.arange(0.4,10.4,1.0))
ax.set_yticklabels(np.arange(1,11))
ax.set(xlabel='xlabel', ylabel='ylabel',
       title='Title')
```

Output

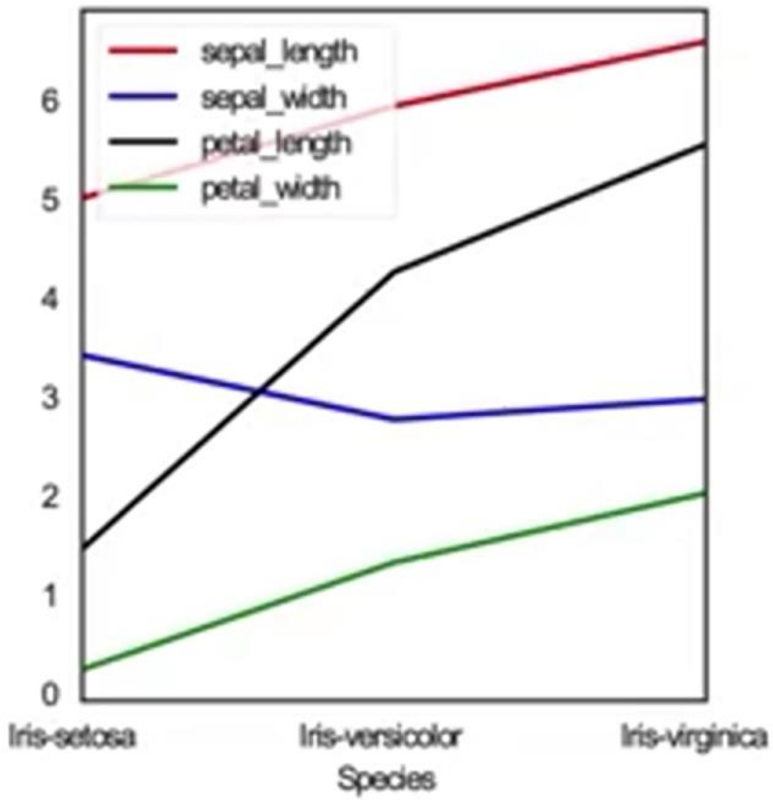


Vẽ biểu đồ theo nhóm

Code

```
# Pandas DataFrame approach
data.groupby('species').mean()
.plot(color=['red','blue',
      'black','green'],
      fontsize=10.0, figsize=(4,4))
```

Output

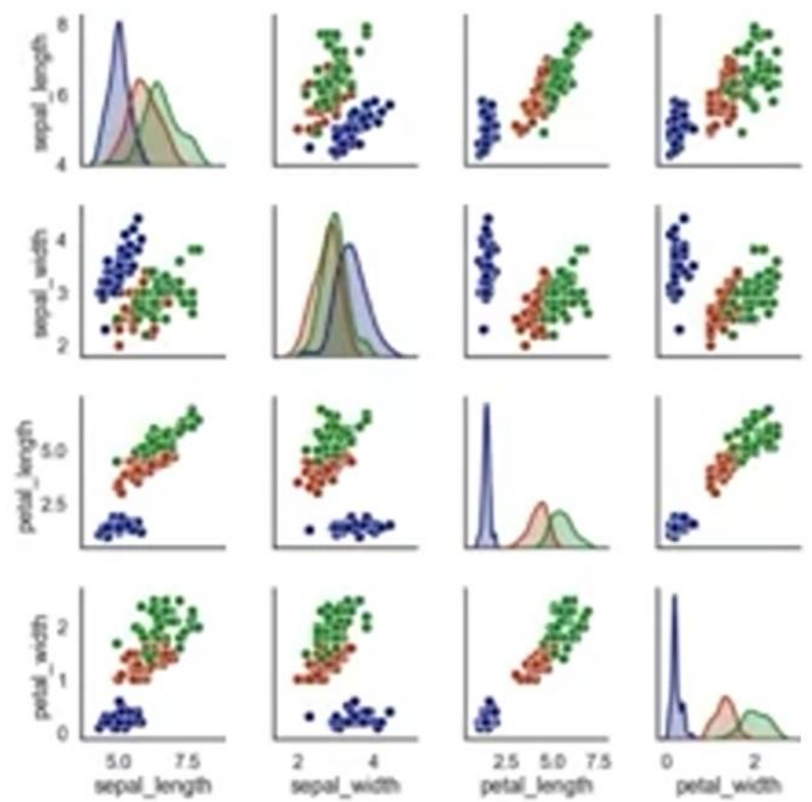


Biểu đồ cặp các thuộc tính

Code

```
# Seaborn plot, feature correlations
sns.pairplot(data,
             hue='species', size=3)
```

Output

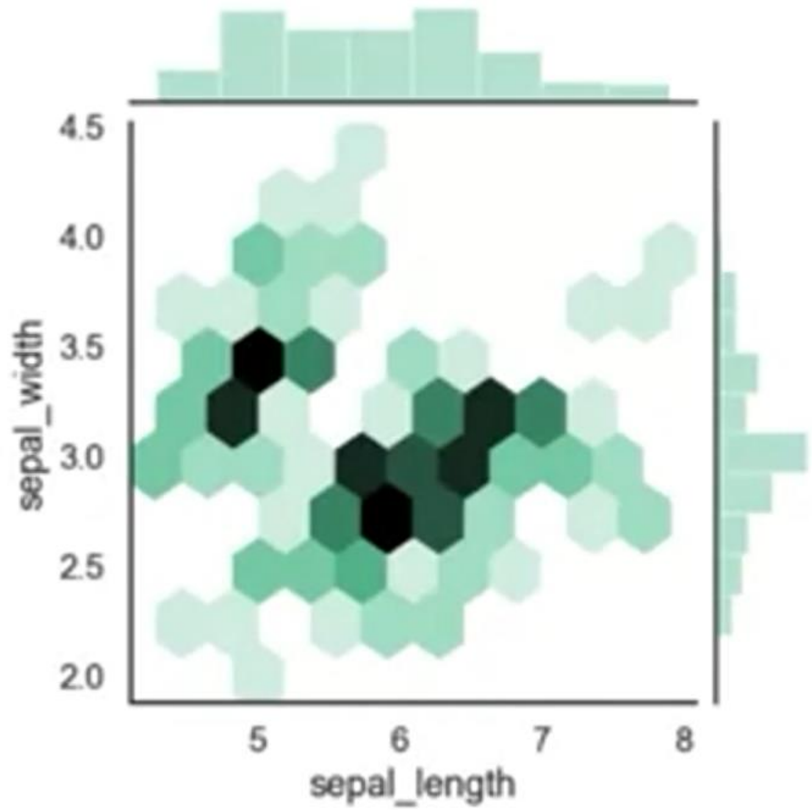


Biểu đồ Hexbin

Code

```
# Seaborn hexbin plot
sns.jointplot(x=data['sepal_length'],
              y=data['sepal_width'],
              kind='hex')
```

Output

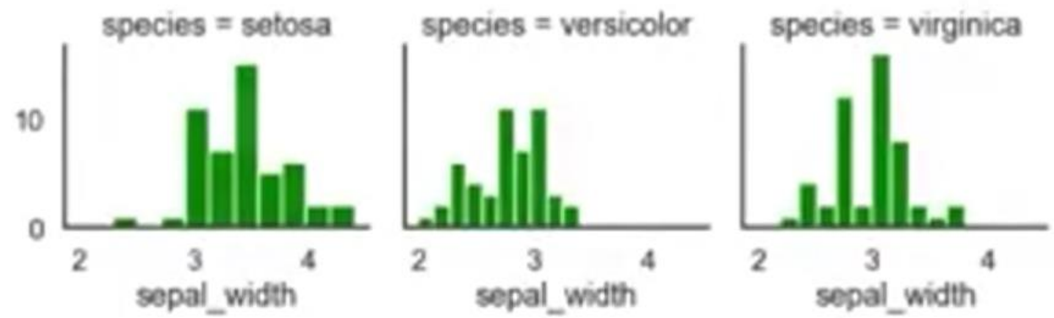


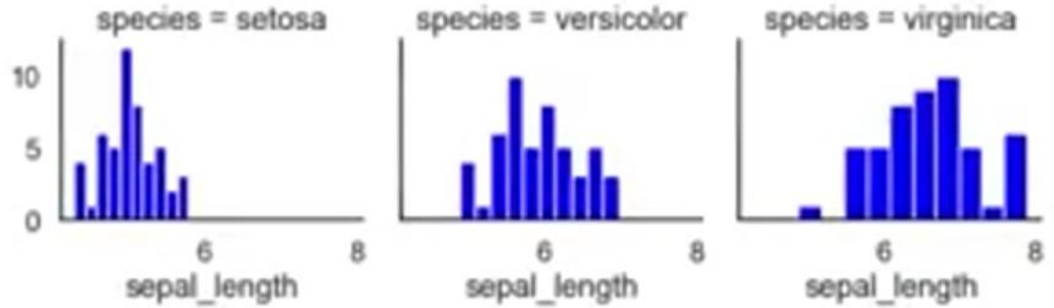
Biểu đồ Facet Grid

Code

```
# Seaborn plot, Facet Grid
# First plot statement
plot = sns.FacetGrid(data,
                     col='species',
                     margin_titles=True)
plot.map(plt.hist, 'sepal_width',
         color='green')
# Second plot statement
plot = sns.FacetGrid(data,
                     col='species',
                     margin_titles=True)
plot.map(plt.hist, 'sepal_length',
         color='blue')
```

Output



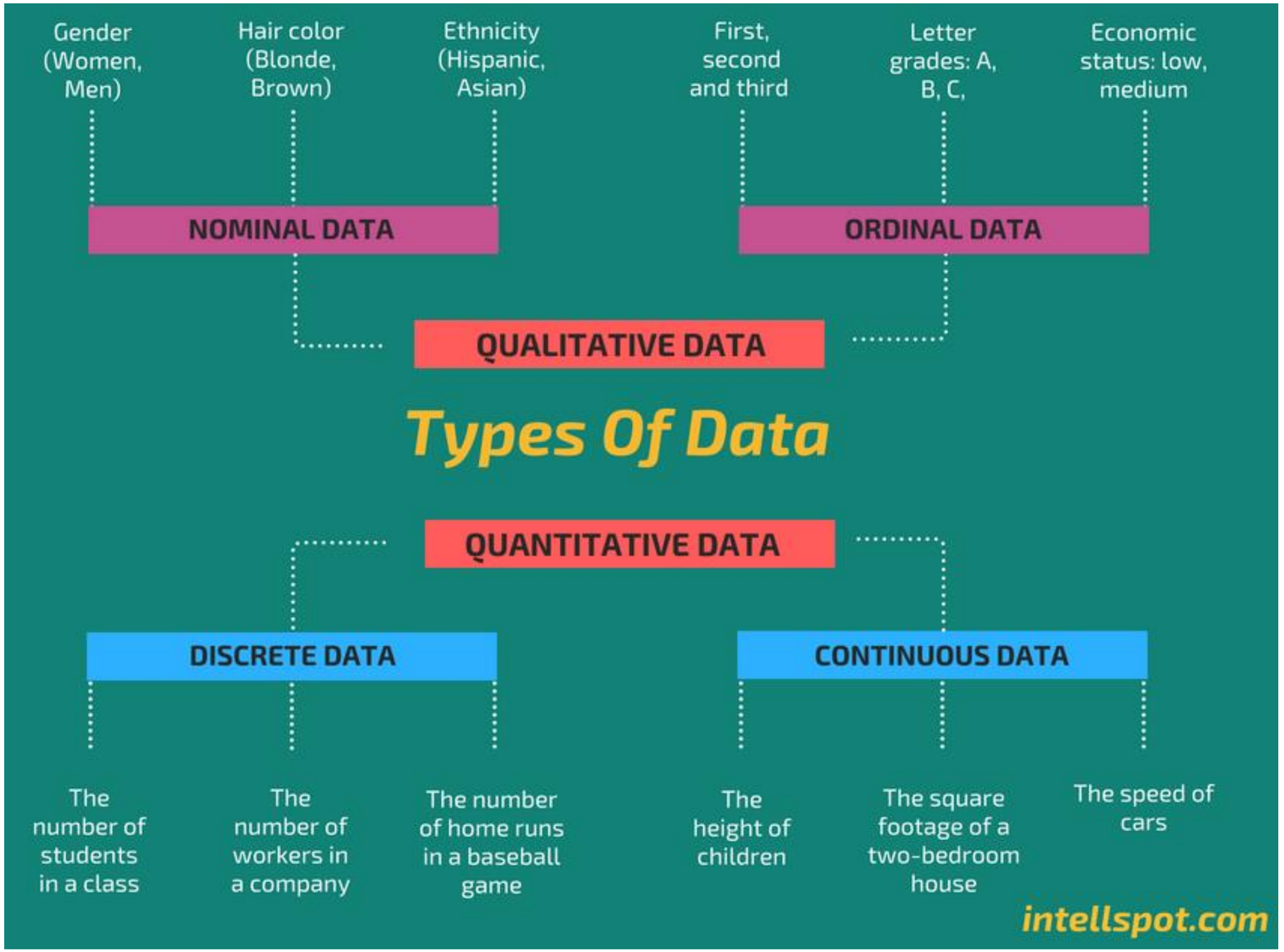


Biến đổi dữ liệu

- Biến đổi dữ liệu (data transformation) là quá trình chuyển đổi dữ liệu thô thành một định dạng hoặc cấu trúc phù hợp cho việc xây dựng mô hình và khám phá dữ liệu.
- Thuật toán có nhiều khả năng bị sai lệch khi phân phối dữ liệu bị lệch về tỷ lệ, đơn vị đo …
- Biến đổi dữ liệu về cùng một tỷ lệ cho phép thuật toán so sánh mối quan hệ tương đối giữa các điểm dữ liệu tốt hơn

Biến đổi dữ liệu

- Các loại biến đổi
  - ➢ Encoding (Mã hóa): biến đổi các thuộc tính không phải dạng số sang dạng số
  - ➢ Scaling (Thay đổi tỷ lệ): thay đổi tỷ lệ của các dữ liệu dạng số để chúng trở nên tương đồng với nhau



Biến đổi mã hóa (encoding)

- Encoding thường được sử dụng để biến đổi dữ liệu dạng category (loại). Có 2 loại cơ bản:
  - Dạng danh nghĩa: các giá trị phân loại không có thứ tự (e.g. Red, Blue, Green; True, False)
  - Dạng thứ tự: Các giá trị phân loại có thứ tự (e.g. High, Medium, Low)

Biến đổi mã hóa: Các phương pháp

- **Mã hóa nhị phân**: chuyển đổi các biến thành giá trị 0 hoặc 1. Thích hợp với các thuộc tính chỉ có thể nhận 1trong 2 giá trị (e.g. True, False)
- **Mã hóa One-hot**: chuyển đổi các biến thành nhiều biến chỉ nhận giá trị 0 hoặc 1 cho mỗi loại. Cách này sẽ tạo ra nhiều biến mới.
- **Mã hóa thứ tự**: khi muốn biến đổi các thuộc tính dạng phân loại có thứ tự ta có thể tạo một biến số nguyên nhận giá trị tương ứng với số lượng loại của biến đó (e.g. 0,1,2,3,…)

Biến đổi tỷ lệ (scaling)

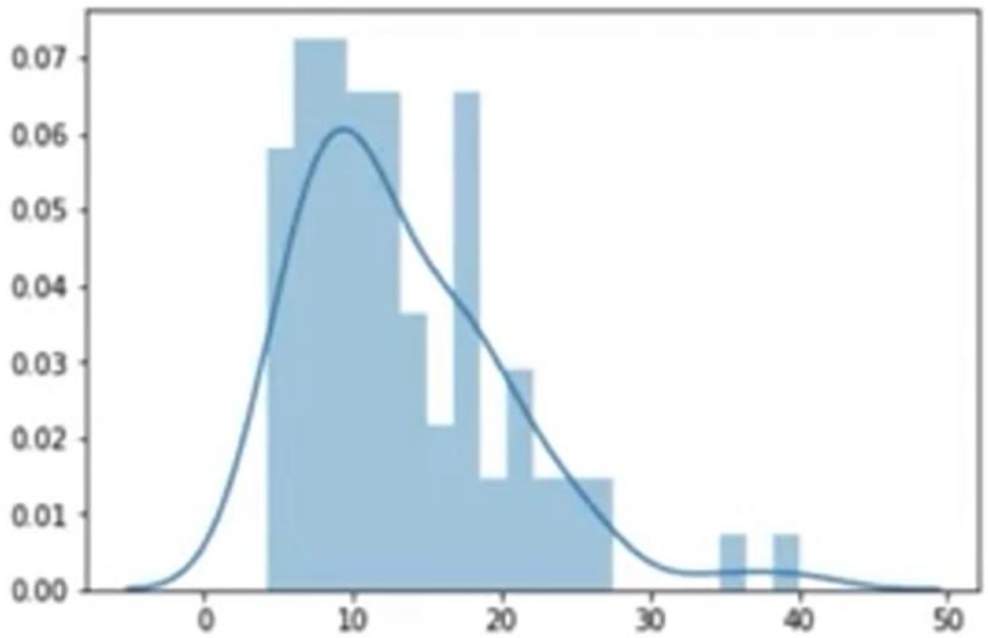
- Biến đổi tỷ lệ cho phép các biến sau khi biến đổi có thể so sánh được với nhau.
- Các thuộc tính dạng số liên tục thường có các thang tỷ lệ khác nhau.

Biến đổi tỷ lệ: Các phương pháp

- **Standard scaling**: biến đổi các thuộc tính về dạng thuộc tính có phân bố theo phân phối chuẩn (bằng cách trừ đi giá trị trung bình sau đó chia cho phương sai)
- **Min-max scaling**: biến đổi các thuộc tính về dạng thuộc tính có giới hạn trong khoảng (0,1) bằng cách ánh xạ giá trị nhỏ nhất thành 0 và giá trị lớn nhất thành 1. Kiểu biến đổi này rất nhạy cảm với các dữ liệu ngoại lệ.
- **Robust scaling**: gần giống với min-max scaling, nhưng thay vào đó ánh xạ khoảng tứ phân vị (được tính ra bằng cách lấy giá trị tứ phân vị thứ ba trừ đi giá trị tứ phân vị thứ nhất) thành (0,1). Điều này có nghĩa là biến sẽ nhận các giá trị nằm ngoài khoảng (0,1)

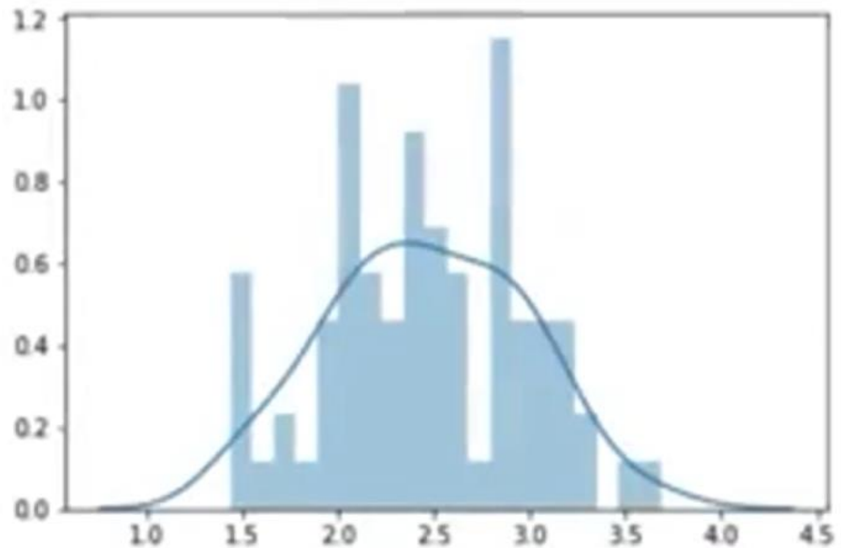
Biến đổi logarithm với dữ liệu bị lệch phải

```
# plot a histogram and density plot
sns.distplot(data, bins=20);
```



Right (positive) skewed

```
import math
log_data = [math.log(d) for d in data['Unemployment']]

# plot transformed plots
sns.distplot(log_data, bins=20);
```



Normal

Biến đổi dữ liệu: Tổng kết

Feature Type	Transformation	Library
Continous: numerical values	Standard, Min-max, Robust Scaling	from sklearn.preprocessing import StandardScaler, MinMaxScaler, RobustScaler
Nominal: categorical, unordered features	Binary, One-hot Encoding (0,1)	from sklearn.preprocessing import LabelEncoder, LabelBinarizer, OneHotEncoder
Ordinal: categorical, ordered features	Ordinal Encoding (0,1,2,3)	from sklearn.preprocessing import OrdinalEncoder


Thank
you!