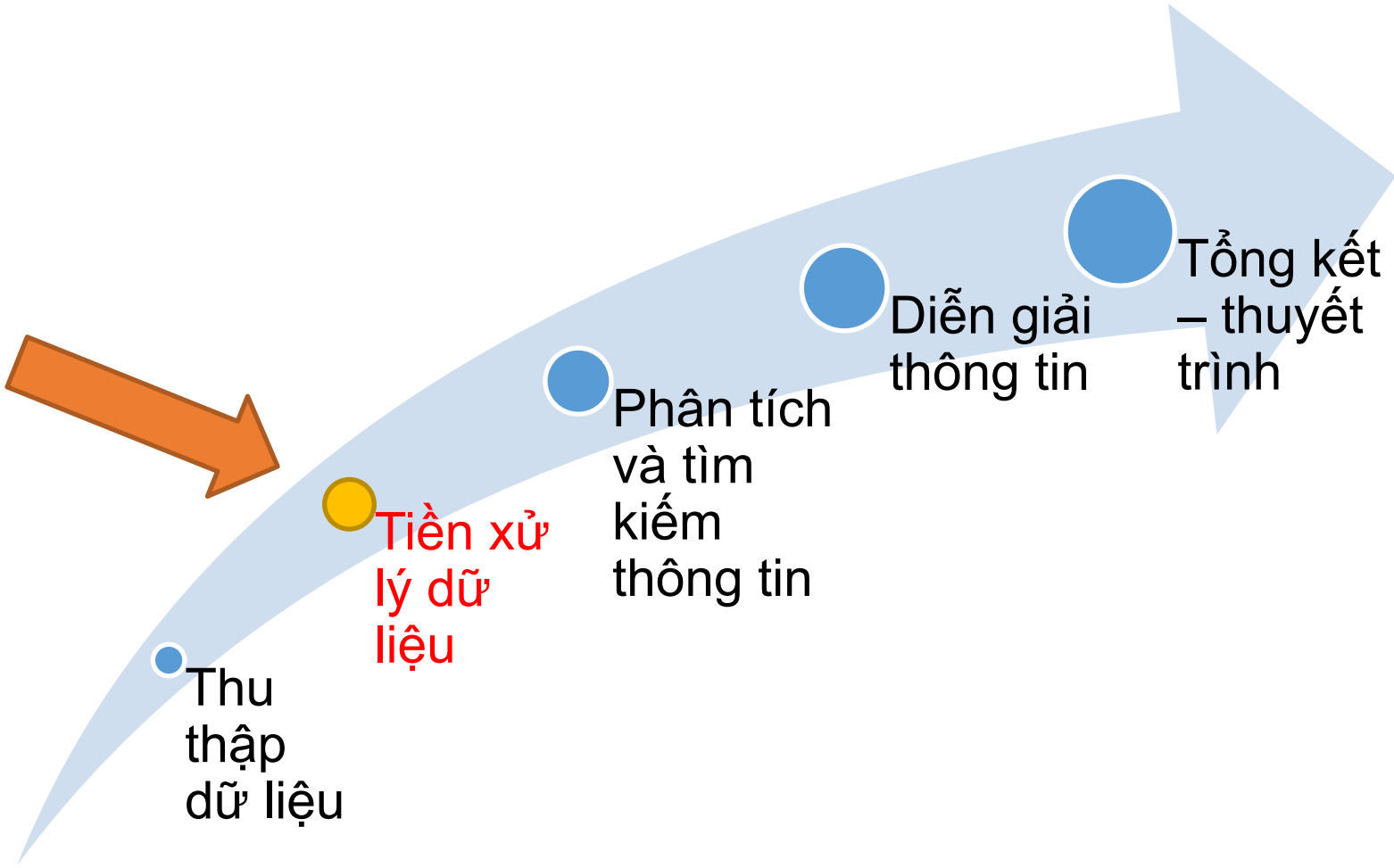
Bài 4
Tiền xử lý dữ liệu

Khóa học: Phân tích dữ liệu với Python

Mục tiêu bài học

1. Tiền xử lý dữ liệu
   - Trình bày được các vấn đề liên quan đến dữ liệu
   - Giải thích được tại sao cần tiền xử lý dữ liệu
   - Thực hiện được các thao tác xử lý dữ liệu khuyết thiếu
   - Thực hiện được các thao tác xử lý dữ liệu ngoại lai
2. Chuẩn hóa được dữ liệu
   - Trình bày được ý nghĩa của việc chuẩn hóa dữ liệu
   - Trình bày được các phương pháp chuẩn hóa
   - Áp dụng được các phương pháp chuẩn hóa dữ liệu

1. Giới thiệu


Thu thập dữ liệu
Tiền xử lý dữ liệu
Phân tích và tìm kiếm thông tin
Diễn giải thông tin
Tổng kết – thuyết trình

Các vấn đề với dữ liệu

Dữ liệu thu được từ thực tế có vấn đề:

- Không hoàn chỉnh: Thiếu thuộc tính
  - Do tại thời điểm thu thập không có.
  - Các vấn đề do sai sót từ phần mềm, người thu thập dữ liệu
- Nhiều lỗi: Giá trị thuộc tính bị sai kiểu
  - Do việc nhập dữ liệu
  - Do việc truyền dữ liệu
- Không đồng nhất
  - Dữ liệu đến từ nhiều nguồn

Tại sao cần tiền xử lý dữ liệu?

- Nếu dữ liệu không sạch, không được chuẩn hóa, kết quả phân tích sẽ bị ảnh hưởng, không đáng tin cậy

- Các kết quả phân tích không chính xác sẽ dẫn đến các quyết định không tối ưu hoặc sai.

2. Làm sạch dữ liệu

- Xử lý dữ liệu khuyết thiếu
- Xử lý dữ liệu ngoại lai

Xử lý dữ liệu khuyết thiếu

- Dữ liệu khuyết thiếu là những dữ liệu bị thiếu, được hiển thị như Nan, Null, N/A, ...

- Nguyên nhân:
  - người dùng quên điền,
  - dữ liệu bị mất,
  - không thu thập được thông tin, ...

- Tại sao cần xử lý?

  Hầu hết các thuật toán học máy đều không thể hoạt động hoặc hoạt động không chính xác với dữ liệu khuyết thiếu.

Xử lý dữ liệu khuyết thiếu

- Dữ liệu khuyết thiếu là những dữ liệu bị thiếu, được hiển thị như Nan, Null, N/A, ...

- Nguyên nhân:
  - người dùng quên điền,
  - dữ liệu bị mất,
  - không thu thập được thông tin, ...

- Tại sao cần xử lý?

  Hầu hết các thuật toán học máy đều không thể hoạt động hoặc hoạt động không chính xác với dữ liệu khuyết thiếu.

Xử lý dữ liệu khuyết thiếu

- Dữ liệu khuyết thiếu trong pandas:
  - np.nan
  - None
  - NaT
  - Nan
  - NA
- Ví dụ:

Tạo dữ liệu khuyết thiếu với pandas

```
>> pd.Series([1, 2, np.nan, 4], dtype=pd.Int64Dtype())
```

Xử lý dữ liệu khuyết thiếu

- Kiểm tra dữ liệu khuyết thiếu
  - Kiểm tra dữ liệu bị khuyết: isna(), isnull()
  - Kiểm tra dữ liệu không bị khuyết: notna(), notnull()
- Ví dụ:

```
>> s = pd.Series([5, 6, np.nan])
```

Kiểm tra khuyết:

```
>>s.isna()
```

Kiểm tra không bị khuyết:

```
>> s.notna()
```

Xử lý dữ liệu khuyết thiếu

- Xóa dòng chứa ít nhất 1 giá trị bị khuyết:
  dropna()
- Xóa cột chứa ít nhất 1 giá trị bị khuyết:
  dropna(axis='columns') hoặc dropna(axis=1)
- Xóa các dòng mà chứa toàn giá trị bị khuyết:
  dropna(how='all')
- Chỉ giữ lại những dòng có ít nhất n giá trị bị khuyết:
  dropna(thresh=n)
- Xóa những dòng nếu có dữ liệu khuyết trên một số cột:
  df.dropna(subset=[danh sách cột])

Xử lý dữ liệu khuyết thiếu

Thay thế dữ liệu khuyết: fillna()

- Thay thế giá trị bị khuyết bằng một giá trị vô hướng:
  fillna(value)
- Thay thế bằng các giá trị trước:
  fillna(method="ffill")
- Thay thế bằng các giá trị phía sau:
  fillna(method=”bfill")
- Thay thế bằng các giá trị xác định trên mỗi cột:
  fillna(value= {"A": 0, "B": 1, "C": 2, "D": 3})

Xử lý dữ liệu ngoại lai

- Dữ liệu ngoại lai là gì?
  - mẫu dữ liệu đặc biệt, cách xa khỏi phần lớn dữ liệu khác trong tập dữ liệu
  - chưa có một định nghĩa toán học cụ thể nào để xác định một điểm ngoại lai
- Ví dụ:

80,71,79,61,78,73,77,74,76,75,**160**,79,80,78,75,78,86,80,
82,69,**100**,72,74,75,**180**,72,71,**12**.

- Cách xử lý: Xóa hay sửa?

  Tùy thuộc bài toán

Xử lý dữ liệu ngoại lai

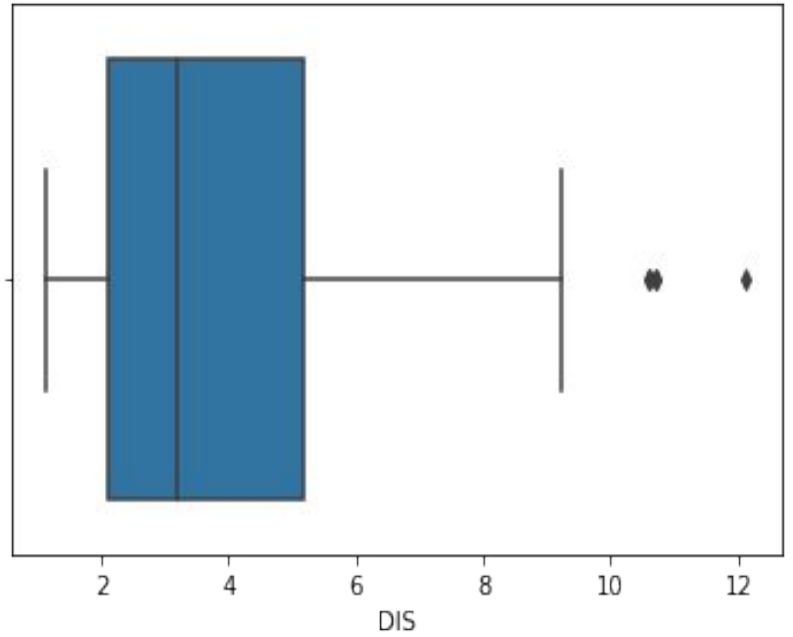
- Xác định dữ liệu ngoại lai
  - Phương pháp trực quan hóa: Box plot, Scatter Plot
  - Phương pháp toán học: Z-Score, IQR-Score

Xử lý dữ liệu ngoại lai

Xác định dữ liệu ngoại lai bằng box plot

- phương pháp mô tả dữ liệu số bằng đồ thị thông qua các tứ phần tư
- Các giá trị ngoại lai có thể được vẽ dưới dạng các điểm riêng lẻ, tách biệt khỏi hộp
- Vẽ bằng seaborn:

>> seaborn.boxplot(x=data)

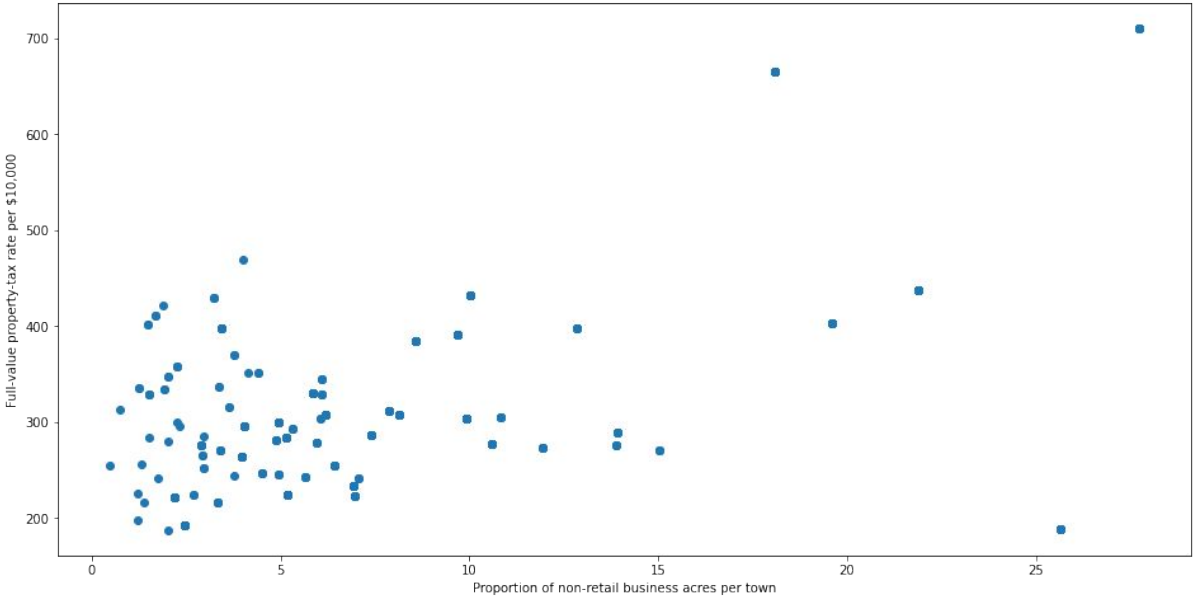


Biểu đồ bên cho thấy 3 điểm ngoại lai trong khoảng từ 10-12

Xử lý dữ liệu ngoại lai

Xác định dữ liệu ngoại lai bằng scatter plot

- Biểu đồ sử dụng hệ tọa độ Descartes để hiển thị giá trị cho hai biến điển hình cho một tập dữ liệu
- Dữ liệu được hiển thị dưới dạng tập hợp các điểm



Biểu đồ trên cho thấy hầu hết các điểm dữ liệu nằm ở phía dưới bên trái nhưng có những điểm nằm xa so với hầu hết dữ liệu, như góc trên cùng bên phải.

Xử lý dữ liệu ngoại lai

Xác định dữ liệu ngoại lai bằng Z-Score

- số độ lệch chuẩn có dấu mà giá trị của một điểm quan sát hoặc dữ liệu cao hơn giá trị trung bình của những gì đang được quan sát
- chúng ta sẽ chuẩn hóa dữ liệu, đưa dữ liệu về xung quanh điểm 0, rồi tìm những điểm cách xa 0.
- Thông thường, những điểm ngoại lai cách 0 ngưỡng là 3 hoặc -3.

Xử lý dữ liệu ngoại lai

Xác định dữ liệu ngoại lai bằng Z-Score

- Có thể tính Z-Score bằng thư viện scipy

```
>> from scipy import stats
>> import numpy as np
>> z = np.abs(stats.zscore(data))
>> print(z)
```

- Xác định ngoại lai theo ngưỡng

```
>> np.where(z > threshold)
```

Xử lý dữ liệu ngoại lai

Xác định dữ liệu ngoại lai bằng IQR Score

- Một thước đo phân tán thống kê, được tính bằng sự khác biệt giữa phân vị 75% (Q3) và 25% (Q1), hoặc giữa phần tư trên và dưới

  IQR = Q3 – Q1

- Thước đo độ phân tán tương tự như độ lệch chuẩn hoặc phương sai, nhưng mạnh hơn nhiều so với các giá trị ngoại lai

- Điểm ngoại lai sẽ là những điểm mà nằm dưới Q1 ít nhất là 1.5*IQR, hoặc nằm trên Q3 ít nhất là 1.5*IQR.

Xử lý dữ liệu ngoại lai

Xác định dữ liệu ngoại lai bằng IQR Score

- Tính toán IQR Score

```
>> Q1 = data.quantile(0.25)
>> Q3 = data.quantile(0.75)
>> IQR = Q3 - Q1
>> print(IQR)
```

- Xác định ngoại lai

```
>> data < (Q1 - 1.5 * IQR)) | (data > (Q3 + 1.5 * IQR))
```

Xử lý dữ liệu ngoại lai

Nên xóa chúng hay sửa dữ liệu ngoại lai?

- Sửa: Tùy thuộc bài toán, yêu cầu cụ thể sẽ thay thế theo dữ liệu thực tế.
- Xóa:
  - Xóa theo Z-Score
  - Xóa theo IQR SCore

Xử lý dữ liệu ngoại lai

- Xóa dữ liệu ngoại lai theo Z-Score:

```
>> data = data[(z < 3).all(axis=1)]
```

- Xóa dữ liệu theo IQR Score:

```
>> data = data[~((data < (Q1 - 1.5 * IQR)) | (data > (Q3 + 1.5 * IQR))).any(axis=1)]
```

Chuẩn hóa dữ liệu

- Dữ liệu tại sao cần chuẩn hóa?
  - Các điểm dữ liệu đôi khi được đo đạc với những đơn vị khác nhau, m và feet chẳng hạn.
  - Có hai thành phần (của vector dữ liệu) chênh lệch nhau quá lớn, một thành phần có khoảng giá trị từ 0 đến 1000, thành phần kia chỉ có khoảng giá trị từ 0 đến 1 chẳng hạn.

Cần chuẩn hóa dữ liệu trước khi thực hiện các bước tiếp theo.

Chuẩn hóa dữ liệu

- Phương pháp chuẩn hóa:
  - Z-Score scaling
  - Min-max scaling
  - Robust scaling

Z-Score scaling

- Scale dữ liệu về một phân bố chuẩn trong đó giá trị trung bình của các quan sát bằng 0 và độ lệch chuẩn = 1.

$$z = (x - u) / s$$

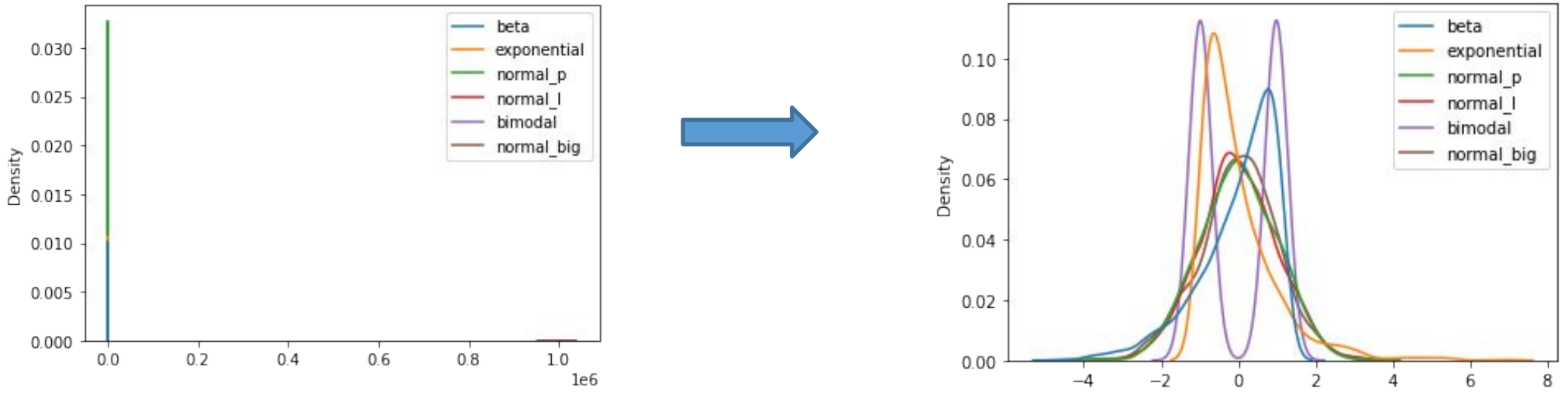
Trong đó:

  - u là mean của dữ liệu huấn luyện
  - s là độ lệch chuẩn
  - x là điểm dữ liệu cần chuẩn hóa
  - z là dữ liệu được chuẩn hóa
- Mang lại hiệu quả tốt hơn đối với tác vụ phân lớp so với tác vụ hồi quy.
- Chỉ hoạt động tốt nếu dữ liệu được phân bố theo phân phối chuẩn.

Z-Score scaling

- Thực hiện Z-Score scaling bằng scikit-learn

```
>> from sklearn.preprocessing import StandardScaler
>> s_scaler = StandardScaler()
>> df_s = s_scaler.fit_transform(df)
```



Min-max scaling

- Chuyển đổi các đặc trưng bằng cách scale mỗi đặc trưng về 1 phạm vi nhất định.
- Công thức của nó là:

  X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))

  X_scaled = X_std * (max - min) + min

Trong đó:

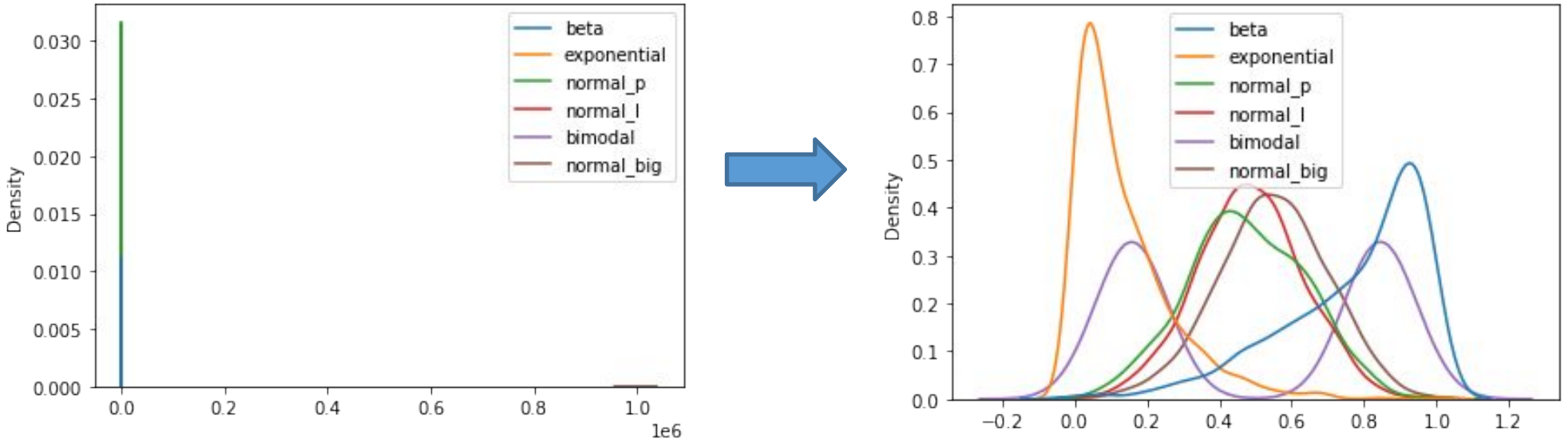
- X là điểm dữ liệu cần chuẩn hóa,
- X_scaled là dữ liệu được chuẩn hóa,
- X_std là tỉ lệ chuẩn hóa,
- max và min là khoảng chuẩn hóa của giá trị.

- Nhược điểm: khá nhạy cảm với dữ liệu ngoại lai

Min-max scaling

- Thực hiện Min-max scaling bằng scikit-learn

```
>> from sklearn.preprocessing import MinMaxScaler
>> s_scaler = MinMaxScaler()
>> df_s = s_scaler.fit_transform(df)
```



Robust scaling

- Chuyển đổi các đặc trưng bằng cách scale mỗi đặc trưng sử dụng thống kê, loại bỏ ngoại lai.

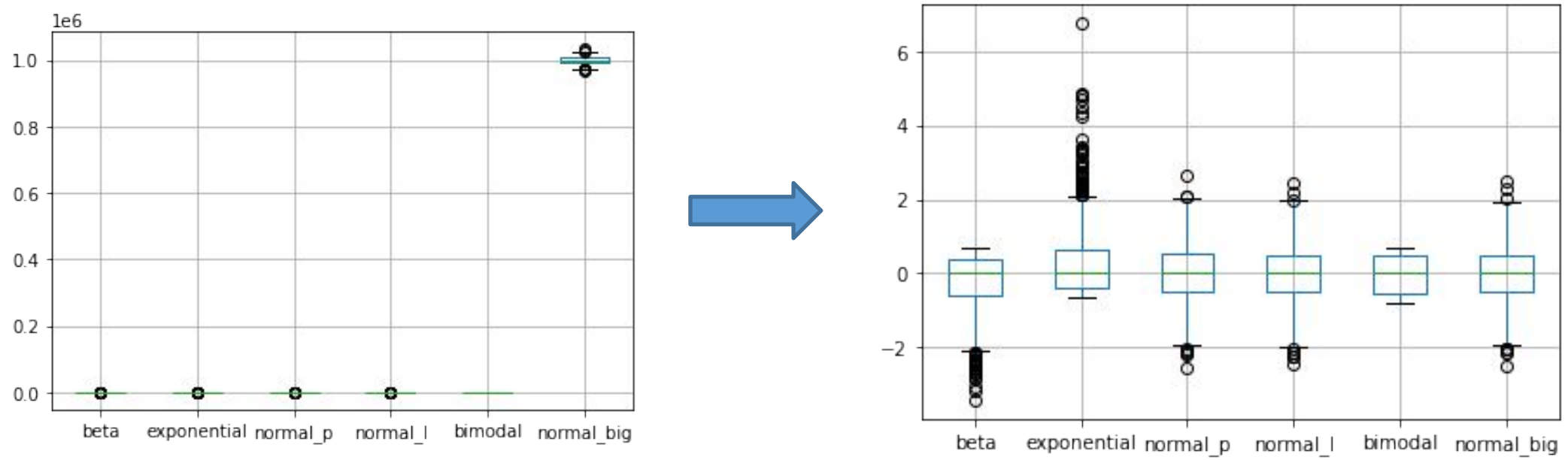
- Sau khi chuẩn hóa với RobustScaler, có thể áp dụng thêm StandardScaler hoặc MinMaxScaler.

- Bộ chuẩn hóa này chia tỷ lệ dữ liệu thành các phần tư (mặc định là IQR: Interquartile Range). IQR là phạm vi giữa phần tư thứ nhất (25%) và phần tư thứ ba (75%).

Robust scaling

- Thực hiện Robust scaling bằng scikit-learn

```
>> from sklearn.preprocessing import RobustScaler
>> s_scaler = RobustScaler()
>> df_s = s_scaler.fit_transform(df)
```



Tóm tắt

Qua bài học này, chúng ta đã tìm hiểu những kiến thức sau:

- Kiểm tra, xóa bỏ, thay thế dữ liệu khuyết thiếu
- Phát hiện dữ liệu ngoại lai bằng phương pháp trực quan hóa và toán học, loại bỏ dữ liệu ngoại lai.
- Chuẩn hóa dữ liệu theo 3 phương pháp là z-score scaling, min-max scaling và robust scaling