# PHẦN 1: THỐNG KÊ MÔ TẢ

## + Thống kê mô tả là gì? Nó khác gì với thống kê suy luận (inferential statistics)?

### Thống kê mô tả (Descriptive Statistics) là gì?

Thống kê mô tả là tập hợp các phương pháp dùng để tóm tắt, tổ chức và trình bày dữ liệu một cách dễ hiểu, giúp mô tả các đặc điểm chính của tập dữ liệu mà không đưa ra kết luận vượt ra ngoài dữ liệu đã thu thập. Nó tập trung vào việc cung cấp một cái nhìn tổng quan về dữ liệu thông qua các số liệu, biểu đồ, hoặc bảng, thường được sử dụng trong giai đoạn khám phá dữ liệu (EDA).

#### 1.1 Các thành phần chính của thống kê mô tả:

Đo lường xu hướng trung tâm (Central Tendency):

Mean (Trung bình): Tổng các giá trị chia cho số lượng giá trị, ví dụ: trung bình điểm thi của một lớp.

Median (Trung vị): Giá trị ở giữa khi dữ liệu được sắp xếp theo thứ tự, ít nhạy cảm với giá trị ngoại lai hơn mean.

Mode (Mode): Giá trị xuất hiện nhiều nhất trong tập dữ liệu.

Đo lường độ phân tán (Dispersion):

Variance (Phương sai): Đo lường mức độ phân tán của dữ liệu so với trung bình.

Standard Deviation (Độ lệch chuẩn): Căn bậc hai của phương sai, dễ diễn giải hơn vì cùng đơn vị với dữ liệu.

Range (Khoảng giá trị): Sự chênh lệch giữa giá trị lớn nhất và nhỏ nhất.

Interquartile Range (IQR): Khoảng cách giữa tứ phân vị thứ ba (Q3) và thứ nhất (Q1), dùng để phát hiện ngoại lai.

Phân bố (Distribution):

Mô tả hình dạng của dữ liệu, như phân bố chuẩn (normal), lệch trái/phải (skewed), hoặc đa đỉnh (multimodal).

Thường được trực quan hóa qua histogram, boxplot, hoặc density plot.

Tóm tắt định tính:

Với dữ liệu phân loại (categorical), thống kê mô tả bao gồm tần suất (frequency) hoặc tỷ lệ phần trăm của từng danh mục, ví dụ: số lượng khách hàng theo khu vực.

#### 1.2 Ví dụ:

Giả sử bạn có dữ liệu về doanh thu hàng tháng của một cửa hàng (triệu VND): [500, 600, 550, 700, 300]. Thống kê mô tả có thể bao gồm:

Mean: (500 + 600 + 550 + 700 + 300) / 5 = 530 triệu.

Median: 550 triệu (sắp xếp: 300, 500, 550, 600, 700).

Range: 700 - 300 = 400 triệu.

Trực quan hóa: Vẽ histogram hoặc bar chart để thấy xu hướng doanh thu.

#### 1.3 Ứng dụng:

Cung cấp cái nhìn tổng quan về dữ liệu trong EDA.

Hỗ trợ báo cáo và trình bày dữ liệu cho các đối tượng không chuyên.

Dùng để kiểm tra chất lượng dữ liệu (phát hiện giá trị thiếu, ngoại lai).

### Thống kê suy luận (Inferential Statistics) là gì?

Thống kê suy luận là tập hợp các phương pháp dùng để đưa ra kết luận hoặc dự đoán về một tổng thể (population) dựa trên dữ liệu từ một mẫu (sample). Nó vượt ra ngoài việc mô tả dữ liệu hiện có để suy ra các đặc tính hoặc kiểm tra giả thuyết về tổng thể, thường kèm theo mức độ không chắc chắn (uncertainty) được biểu thị qua xác suất.

#### 2.1 Các thành phần chính của thống kê suy luận:

Ước lượng (Estimation):

Ước lượng điểm (Point Estimation): Sử dụng một giá trị duy nhất để ước lượng tham số tổng thể, ví dụ: dùng trung bình mẫu để ước lượng trung bình tổng thể.

Ước lượng khoảng (Interval Estimation): Cung cấp một khoảng giá trị (khoảng tin cậy - confidence interval) mà tham số tổng thể có khả năng nằm trong đó, ví dụ: “Trung bình doanh thu tổng thể nằm trong khoảng 500 ± 50 triệu với độ tin cậy 95%.”

Kiểm định giả thuyết (Hypothesis Testing):

Kiểm tra xem một giả thuyết về tổng thể có đúng không, ví dụ: “Liệu trung bình doanh thu của cửa hàng có lớn hơn 600 triệu không?”

Sử dụng p-value hoặc các bài kiểm định như t-test, chi-square test để đánh giá.

Dự đoán (Prediction):

Sử dụng mô hình (như hồi quy tuyến tính, logistic regression) để dự đoán giá trị dựa trên mẫu, ví dụ: dự đoán doanh thu tương lai dựa trên xu hướng hiện tại.

Mô hình hóa (Modeling):

Xây dựng các mô hình thống kê hoặc máy học để suy ra mối quan hệ giữa các biến, như mối quan hệ giữa chi phí quảng cáo và doanh thu.

#### 2.2 Ví dụ:

Tiếp tục với dữ liệu doanh thu [500, 600, 550, 700, 300]:

Thống kê suy luận: Dựa trên mẫu này, bạn muốn suy ra liệu trung bình doanh thu của tất cả các tháng (tổng thể) có lớn hơn 600 triệu không.

Thực hiện t-test để kiểm tra giả thuyết.

Tính khoảng tin cậy 95% cho trung bình tổng thể.

Xây dựng mô hình hồi quy để dự đoán doanh thu dựa trên các yếu tố như mùa vụ.

#### 2.3 Ứng dụng:

Đưa ra kết luận về tổng thể từ dữ liệu mẫu, ví dụ: đánh giá hiệu quả của một chiến dịch quảng cáo.

Kiểm tra giả thuyết, như so sánh doanh thu giữa hai khu vực.

Dự đoán xu hướng hoặc xây dựng mô hình trong phân tích kinh doanh, khoa học, hoặc y học.

### Sự khác biệt giữa Thống kê mô tả và Thống kê suy luận:

|  |  |  |
| --- | --- | --- |
| Tiêu chí | Thống kê mô tả | Thống kê suy luận |
| Mục tiêu | Tóm tắt và mô tả dữ liệu hiện có. | Suy ra đặc tính của tổng thể từ mẫu. |
| Phạm vi | Chỉ áp dụng cho dữ liệu đã thu thập. | Mở rộng kết luận ra tổng thể chưa biết. |
| Ví dụ công cụ | Mean, median, variance, histogram, boxplot. | t-test, ANOVA, confidence interval, regression. |
| Độ không chắc chắn | Không liên quan đến xác suất. | Dựa trên xác suất để đánh giá độ tin cậy. |
| Trực quan hoá | Sử dụng biểu đồ để tóm tắt (bar chart, histogram). | Ít tập trung vào trực quan, thường dùng để kiểm tra mô hình. |
| Ví dụ câu hỏi | “Doanh thu trung bình tháng này là bao nhiêu?” | “Liệu doanh thu trung bình có vượt 600 triệu không?” |
| Ứng dụng trong EDA | Cung cấp cái nhìn ban đầu về dữ liệu. | Kiểm tra giả thuyết hoặc dự đoán sau EDA. |

### Minh họa qua ví dụ thực tế:

Giả sử bạn có dữ liệu doanh thu hàng tháng của một cửa hàng trong 6 tháng: [500, 600, 550, 700, 300, 650].

**Thống kê mô tả**:

Tính: Mean = 550, Median = 575, Standard Deviation ≈ 139.3.

Vẽ histogram để thấy phân bố doanh thu.

Báo cáo: “Doanh thu trung bình là 550 triệu, với phần lớn dao động từ 300 đến 700 triệu.”

Trực quan hoá:

import matplotlib.pyplot as plt

data = [500, 600, 550, 700, 300, 650]

plt.hist(data, bins=5, color='#1f77b4', edgecolor='black')

plt.title('Phân Bố Doanh Thu Hàng Tháng')

plt.xlabel('Doanh Thu (triệu VND)')

plt.ylabel('Tần Suất')

plt.savefig('doanh\_thu\_histogram.png', dpi=300, bbox\_inches='tight')

plt.show()

**Thống kê suy luận:**

Kiểm tra giả thuyết: “Liệu trung bình doanh thu tổng thể có lớn hơn 600 triệu không?” (sử dụng t-test).

Tính khoảng tin cậy 95% cho trung bình tổng thể.

Dự đoán doanh thu tháng tiếp theo dựa trên mô hình hồi quy.

Ví dụ code (t-test):

from scipy.stats import ttest\_1samp

data = [500, 600, 550, 700, 300, 650]

t\_stat, p\_value = ttest\_1samp(data, popmean=600)

print(f"p-value: {p\_value:.4f}")

# Nếu p-value < 0.05, bác bỏ giả thuyết rằng trung bình = 600 triệu.

### Kết luận:

**Thống kê mô tả** giúp bạn hiểu dữ liệu hiện có thông qua các số liệu tóm tắt và trực quan hóa, là bước đầu tiên trong EDA.

**Thống kê suy luận** cho phép đưa ra kết luận về tổng thể hoặc dự đoán, thường được thực hiện sau khi đã hiểu dữ liệu qua thống kê mô tả.

Trong thực tế, hai phương pháp này thường được sử dụng kết hợp: thống kê mô tả để khám phá và làm sạch dữ liệu, sau đó thống kê suy luận để kiểm tra giả thuyết hoặc xây dựng mô hình.

## + Các thước đo thống kê mô tả chính (ví dụ: trung bình, trung vị, phương sai, độ lệch chuẩn) được sử dụng để làm gì? Trong trường hợp nào thì nên dùng trung vị thay vì trung bình?

### 1. Các thước đo thống kê mô tả chính và ý nghĩa

**Trung bình (Mean)**

Là giá trị “đại diện” cho dữ liệu, được tính bằng tổng các giá trị chia cho số lượng quan sát.

Dùng để mô tả xu hướng trung tâm khi dữ liệu phân bố **tương đối đều** và không có giá trị ngoại lai (outlier).  
Ví dụ: Điểm trung bình một lớp = tổng điểm / số sinh viên.

**Trung vị (Median)**

Là giá trị nằm giữa tập dữ liệu khi đã sắp xếp theo thứ tự.

Chia dữ liệu thành hai nửa bằng nhau (50% nhỏ hơn, 50% lớn hơn).

Không bị ảnh hưởng bởi các giá trị ngoại lai.  
Ví dụ: Nếu lương của 7 nhân viên là [5, 6, 7, 8, 100, 110, 120], thì **trung bình = 51,7** nhưng **trung vị = 8** → trung vị phản ánh thực tế “mức lương điển hình” hơn.

**Phương sai (Variance)**

Đo mức độ **phân tán** của dữ liệu quanh giá trị trung bình.

Công thức: s2=∑(xi−xˉ)2n−1s^2 = \frac{\sum (x\_i - \bar{x})^2}{n-1}s2=n−1∑(xi​−xˉ)2​ (với mẫu).

Giá trị càng lớn → dữ liệu càng “loãng” so với trung bình.

**Độ lệch chuẩn (Standard Deviation – SD)**

Là căn bậc hai của phương sai, có cùng đơn vị với dữ liệu.

Dễ diễn giải hơn phương sai.

Cho biết mức độ dao động trung bình của các giá trị quanh trung bình.  
Ví dụ: Chiều cao trung bình = 168 cm, SD = 5 cm → đa số sinh viên cao từ **163–173 cm**.

### 2. Khi nào nên dùng trung vị thay vì trung bình?

Khi dữ liệu **bị lệch (skewed distribution)**.  
Ví dụ: Thu nhập thường có vài người rất giàu → trung bình bị kéo cao, trung vị phản ánh “người điển hình” hơn.

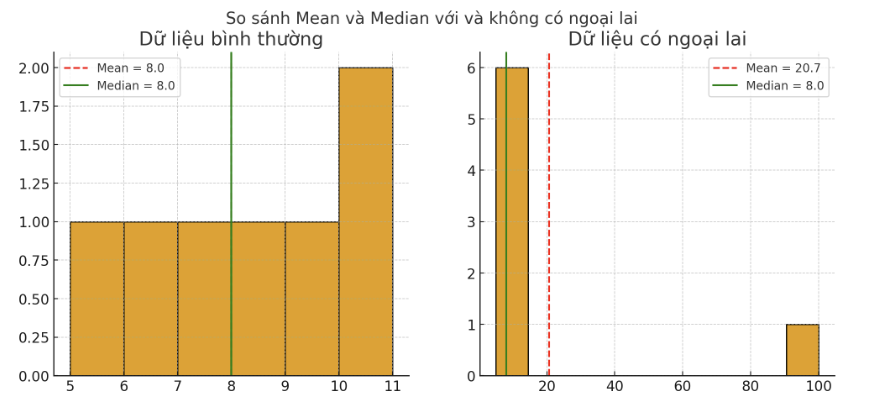
Khi có **giá trị ngoại lai (outliers)**.  
Ví dụ: Điểm thi của cả lớp 40 sinh viên là 7–8 điểm, chỉ có 1 bạn được 0 → trung bình bị kéo xuống, trung vị vẫn giữ nguyên.

Khi muốn mô tả **vị trí trung tâm “ổn định”** của dữ liệu thay vì “giá trị đại diện nhạy cảm” như trung bình.

Tóm lại:

**Trung bình** tốt khi dữ liệu “đẹp”, phân bố chuẩn hoặc gần chuẩn.

**Trung vị** tốt khi dữ liệu lệch, có ngoại lai hoặc khi muốn mô tả “điển hình”.

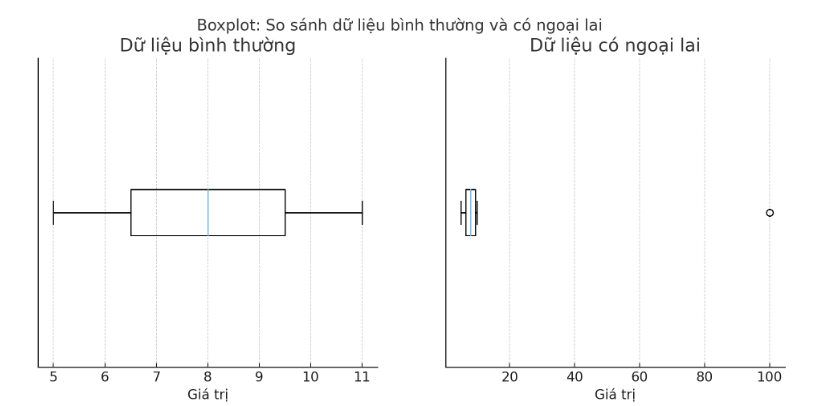


Kết quả: Khi có ngoại lai (giá trị 100), **trung bình tăng vọt từ 8 → 20.7**, trong khi **trung vị vẫn giữ nguyên = 8** → minh chứng rằng trung vị ổn định hơn khi dữ liệu bị lệch.

Ở **dữ liệu bình thường**, Mean (đường đỏ) và Median (đường xanh) gần như trùng nhau.

Ở **dữ liệu có ngoại lai**, Mean bị kéo lệch sang phải (giá trị cao), trong khi Median vẫn giữ nguyên vị trí trung tâm.

Đây chính là lý do tại sao khi dữ liệu có ngoại lai hoặc bị lệch, ta nên dùng **Median** thay vì **Mean**.



**Dữ liệu bình thường**: hộp (IQR) nằm gọn, không có điểm ngoại lai.

**Dữ liệu có ngoại lai**: xuất hiện điểm nằm ngoài hộp (giá trị 100) → ngoại lai, làm kéo dài đường whisker.

Boxplot cho thấy Median (đường ngang trong hộp) vẫn ổn định, còn Mean thì dễ bị lệch bởi ngoại lai.

## + Làm thế nào để xác định phân bố của một tập dữ liệu? Các loại phân bố phổ biến là gì (ví dụ: phân bố chuẩn, lệch trái, lệch phải)?

### 1. Cách xác định phân bố của tập dữ liệu

Để xác định phân bố của một tập dữ liệu, bạn có thể sử dụng các phương pháp sau:

#### a. Trực quan hóa dữ liệu

Trực quan hóa là cách trực tiếp và hiệu quả để nhận diện hình dạng phân bố của dữ liệu. Các công cụ phổ biến bao gồm:

**Biểu đồ Histogram**:

Vẽ histogram để thấy tần suất xuất hiện của các giá trị trong các khoảng (bins).

Hình dạng của histogram giúp nhận biết phân bố (chuẩn, lệch trái, lệch phải, đa đỉnh, v.v.).

**Biểu đồ hộp (Boxplot)**:

Hiển thị trung vị (Q2), tứ phân vị (Q1, Q3), và giá trị ngoại lai.

Boxplot giúp xác định độ lệch (skewness) và sự phân tán của dữ liệu.

Nếu trung vị gần Q1 hơn Q3, dữ liệu có thể lệch phải; nếu gần Q3 hơn Q1, dữ liệu có thể lệch trái.

**Biểu đồ mật độ (Density Plot)**:

Tương tự histogram nhưng mượt mà hơn, giúp dễ dàng nhận ra hình dạng phân bố.

**Biểu đồ Q-Q (Quantile-Quantile Plot)**:

So sánh các phân vị của dữ liệu với phân vị của phân bố chuẩn lý thuyết.

Nếu các điểm nằm gần đường thẳng, dữ liệu có khả năng theo phân bố chuẩn.

#### b. Tính các chỉ số thống kê mô tả

Các chỉ số thống kê giúp định lượng đặc điểm phân bố:

**Trung bình (Mean)**, **Trung vị (Median)**, **Mode**:

Nếu **trung bình ≈ trung vị ≈ mode**, dữ liệu có thể theo phân bố chuẩn hoặc đối xứng.

Nếu **trung bình > trung vị**, dữ liệu có khả năng **lệch phải** (positive skew).

Nếu **trung bình < trung vị**, dữ liệu có khả năng **lệch trái** (negative skew).

Độ lệch (Skewness):



Công thức: Skewness=

Skewness ≈ 0: Phân bố đối xứng (gần chuẩn).

Skewness > 0: Lệch phải.

Skewness < 0: Lệch trái.

**Độ nhọn (Kurtosis)**:

Đo lường độ “nhọn” hoặc “phẳng” của phân bố so với phân bố chuẩn.

Kurtosis = 3 (hoặc 0 khi chuẩn hóa): Phân bố chuẩn.

Kurtosis > 3: Phân bố nhọn hơn (leptokurtic).

Kurtosis < 3: Phân bố phẳng hơn (platykurtic).

**Phạm vi (Range)** và **Khoảng tứ phân vị (IQR)**:

Cung cấp thông tin về độ phân tán, hỗ trợ đánh giá mức độ lan tỏa của dữ liệu.

#### c. Kiểm định thống kê

Các kiểm định thống kê giúp xác định xem dữ liệu có phù hợp với một phân bố cụ thể (thường là phân bố chuẩn) hay không:

**Kiểm định Shapiro-Wilk**: Kiểm tra xem dữ liệu có theo phân bố chuẩn hay không (p-value < 0.05 bác bỏ giả thuyết phân bố chuẩn).

**Kiểm định Kolmogorov-Smirnov**: So sánh phân bố dữ liệu với phân bố lý thuyết.

**Kiểm định Anderson-Darling**: Nhạy hơn với phân bố chuẩn, đặc biệt ở đuôi phân bố.

**Kiểm định D’Agostino’s K²**: Kết hợp skewness và kurtosis để kiểm tra phân bố chuẩn.

#### d. Phân tích đặc điểm của dữ liệu

**Tính chất của biến**: Xác định biến là liên tục (continuous) hay rời rạc (discrete), vì điều này ảnh hưởng đến loại phân bố có thể xảy ra.

**Bối cảnh dữ liệu**: Hiểu ngữ cảnh thu thập dữ liệu (ví dụ, thu nhập thường lệch phải, thời gian chờ thường lệch phải, điểm số có thể gần chuẩn).

### 2. Các loại phân bố phổ biến

Dưới đây là các loại phân bố phổ biến, đặc biệt là phân bố chuẩn, lệch trái, và lệch phải, cùng với đặc điểm và ví dụ:

#### a. Phân bố chuẩn (Normal Distribution)

**Đặc điểm**:

Hình chuông, đối xứng quanh trung bình.

Trung bình = trung vị = mode.

Skewness ≈ 0, Kurtosis ≈ 3 (hoặc 0 khi chuẩn hóa).

Khoảng 68% dữ liệu nằm trong khoảng ±1 độ lệch chuẩn, 95% trong ±2 độ lệch chuẩn, 99.7% trong ±3 độ lệch chuẩn.

**Ví dụ**:

Chiều cao, cân nặng của một quần thể đồng nhất.

Điểm số trong các bài kiểm tra chuẩn hóa (như IQ).

**Ứng dụng**:

Phân bố chuẩn là nền tảng cho nhiều kiểm định thống kê (như t-test, ANOVA).

Dữ liệu gần chuẩn thường dễ phân tích và mô hình hóa.

#### b. Phân bố lệch phải (Positively Skewed Distribution)

**Đặc điểm**:

Đuôi phân bố kéo dài về phía bên phải (giá trị lớn).

Trung bình > trung vị > mode.

Skewness > 0.

Thường xuất hiện trong dữ liệu có giới hạn dưới (như 0) nhưng không có giới hạn trên.

**Ví dụ**:

Thu nhập cá nhân (nhiều người có thu nhập thấp, ít người có thu nhập rất cao).

Thời gian chờ (như thời gian chờ tại bệnh viện).

Thời gian sự cố (thời gian đến khi một thiết bị hỏng).

**Ứng dụng**:

Có thể cần biến đổi dữ liệu (như log hoặc căn bậc hai) để đưa về gần phân bố chuẩn trước khi phân tích.

#### c. Phân bố lệch trái (Negatively Skewed Distribution)

**Đặc điểm**:

Đuôi phân bố kéo dài về phía bên trái (giá trị nhỏ).

Trung bình < trung vị < mode.

Skewness < 0.

Thường xuất hiện trong dữ liệu có giới hạn trên nhưng không có giới hạn dưới.

**Ví dụ**:

Điểm số trong các bài kiểm tra dễ (nhiều người đạt điểm cao, ít người điểm thấp).

Tuổi nghỉ hưu (nhiều người nghỉ hưu ở tuổi cao, ít người nghỉ sớm).

**Ứng dụng**:

Tương tự lệch phải, có thể cần biến đổi để phân tích.

#### d. Phân bố đều (Uniform Distribution)

**Đặc điểm**:

Tất cả giá trị trong một khoảng có xác suất xảy ra như nhau.

Không có đỉnh rõ ràng, histogram gần giống hình chữ nhật.

Skewness ≈ 0 (đối xứng).

**Ví dụ**:

Kết quả của một lần tung xúc xắc (mỗi mặt từ 1 đến 6 có xác suất bằng nhau).

Số ngẫu nhiên được tạo bởi máy tính trong một khoảng cố định.

**Ứng dụng**:

Thường dùng trong mô phỏng hoặc thử nghiệm ngẫu nhiên.

#### e. Phân bố đa đỉnh (Bimodal/Multimodal Distribution)

**Đặc điểm**:

Có hai hoặc nhiều đỉnh (mode) trong histogram.

Có thể chỉ ra sự tồn tại của nhiều nhóm hoặc cụm trong dữ liệu.

**Ví dụ**:

Chiều cao của một quần thể gồm cả nam và nữ (hai đỉnh tương ứng với hai giới).

Điểm số của một bài kiểm tra có hai nhóm học sinh (giỏi và yếu).

**Ứng dụng**:

Gợi ý cần phân tích dữ liệu theo các nhóm riêng biệt.

#### f. Phân bố lũy thừa (Power Law Distribution)

**Đặc điểm**:

Đuôi dài, thường lệch phải mạnh.

Một số ít giá trị rất lớn, phần lớn giá trị nhỏ.

**Ví dụ**:

Quy mô dân số của các thành phố (vài thành phố lớn, nhiều thành phố nhỏ).

Lượt truy cập các trang web (vài trang rất phổ biến, nhiều trang ít truy cập).

**Ứng dụng**:

Thường gặp trong các lĩnh vực như kinh tế, mạng xã hội, hoặc khoa học mạng.

### 3. Ví dụ minh họa

Giả sử tập dữ liệu: [1, 2, 2, 3, 3, 4, 4, 4, 5, 10]

**Bước 1: Trực quan hóa**

**Dữ liệu**

Tập dữ liệu: [1, 2, 2, 3, 3, 4, 4, 4, 5, 10]

**Histogram**: Hiển thị tần suất xuất hiện của các giá trị, giúp nhận diện hình dạng phân bố (lệch phải, lệch trái, hay đối xứng).

**Boxplot**: Hiển thị trung vị, tứ phân vị, và giá trị ngoại lai, giúp xác định độ lệch và phân tán.

**a. Histogram**

Histogram sẽ đếm tần suất các giá trị trong các khoảng (bins). Vì dữ liệu có các giá trị từ 1 đến 10, tôi sẽ chia thành các khoảng hợp lý (ví dụ, các khoảng 1-2, 2-3, ..., 9-10) để thể hiện tần suất.

Dựa trên tập dữ liệu:

Giá trị 1: 1 lần

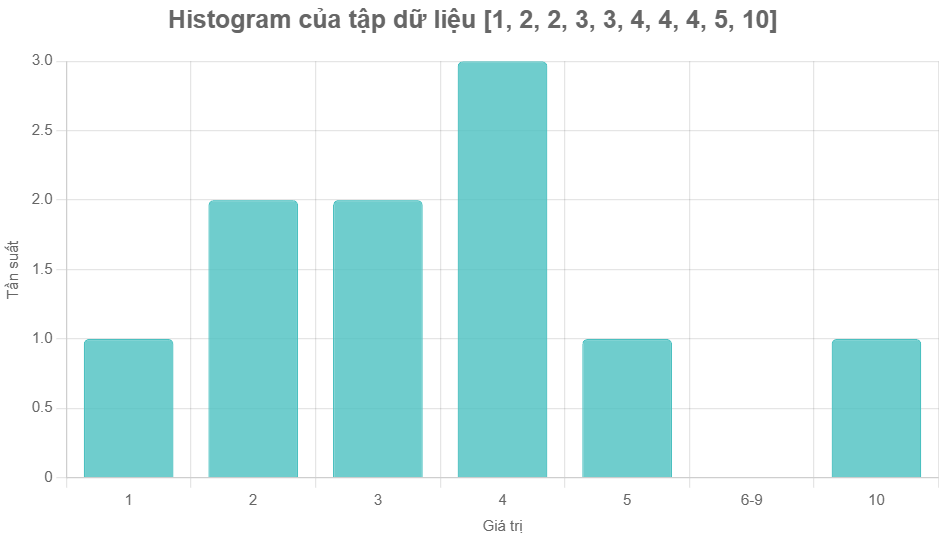
Giá trị 2: 2 lần

Giá trị 3: 2 lần

Giá trị 4: 3 lần

Giá trị 5: 1 lần

Giá trị 10: 1 lần



**Giải thích Histogram**:

Biểu đồ cho thấy tần suất cao nhất ở giá trị 4 (3 lần), và có một đuôi dài bên phải với giá trị 10 (1 lần).

Đuôi dài bên phải (giá trị 10) và sự tập trung ở các giá trị nhỏ hơn (1-5) cho thấy phân bố **lệch phải** (positive skew).

**b. Boxplot**

Để vẽ boxplot, chúng ta cần tính các thành phần:

**Tập dữ liệu đã sắp xếp**: [1, 2, 2, 3, 3, 4, 4, 4, 5, 10]

**Q1 (tứ phân vị thứ nhất)**: Vị trí 25% (khoảng vị trí 2.5) → Trung bình của 2 và 2 = **2**.

**Q2 (trung vị)**: Vị trí 50% (vị trí 5.5) → Trung bình của 3 và 4 = **3.5**.

**Q3 (tứ phân vị thứ ba)**: Vị trí 75% (khoảng vị trí 7.5) → Trung bình của 4 và 5 = **4.5**.

**IQR (khoảng tứ phân vị)**: Q3−Q1=4.5−2=2.5 Q3 - Q1 = 4.5 - 2 = 2.5 Q3−Q1=4.5−2=2.5.

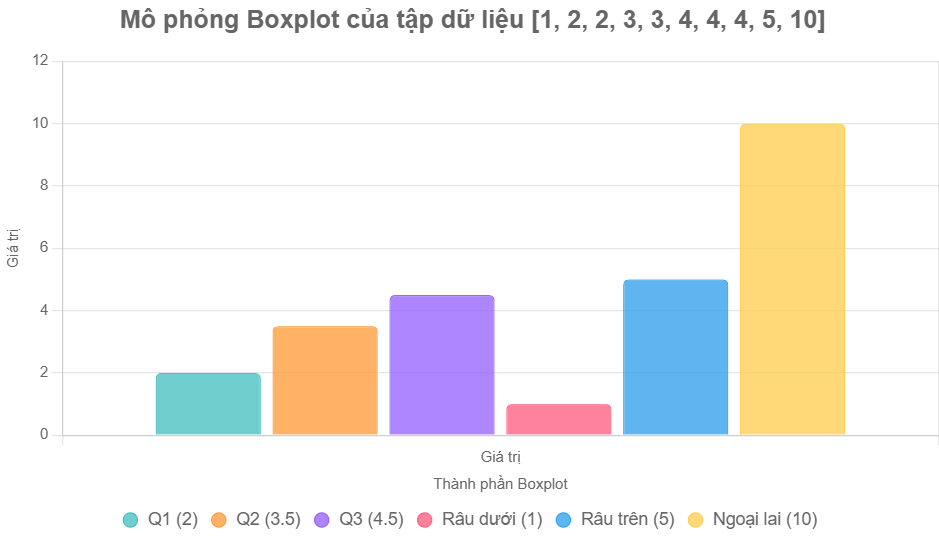
**Râu (Whiskers)**:

Râu dưới: Q1−1.5×IQR=2−1.5×2.5=2−3.75=−1.75 (nhưng giới hạn bởi giá trị nhỏ nhất = 1).

Râu trên: Q3+1.5×IQR=4.5+1.5×2.5=4.5+3.75=8.25 (giá trị tối đa trong phạm vi = 5, vì 10 là ngoại lai).

**Giá trị ngoại lai**: Giá trị 10 > 8.25, nên 10 là ngoại lai.

Boxplot không được hỗ trợ trực tiếp trong Chart.js, nhưng tôi sẽ mô phỏng bằng cách hiển thị các thành phần chính (Q1, Q2, Q3, râu, và ngoại lai) dưới dạng một biểu đồ cột đơn giản với chú thích.



**Giải thích Boxplot**:

**Hộp**: Từ Q1 (2) đến Q3 (4.5), chứa 50% dữ liệu ở giữa, với trung vị Q2 (3.5).

**Râu**: Từ giá trị nhỏ nhất (1) đến giá trị lớn nhất trong phạm vi không ngoại lai (5).

**Ngoại lai**: Giá trị 10 nằm ngoài Q3+1.5×IQR=8.25, xác nhận đuôi dài bên phải.

**Nhận xét**: Trung vị (3.5) gần Q1 (2) hơn Q3 (4.5), và sự hiện diện của ngoại lai ở giá trị 10 củng cố rằng phân bố **lệch phải**.

**Bước 2: Phân tích phân bố**

**Histogram**: Đuôi dài bên phải (giá trị 10) và sự tập trung ở các giá trị nhỏ (1-5) cho thấy phân bố lệch phải.

**Boxplot**: Trung vị gần Q1 hơn Q3, và có ngoại lai ở giá trị cao (10), xác nhận phân bố lệch phải.

**Chỉ số thống kê** (từ ví dụ trước):

Trung bình (3.8) > Trung vị (3.5) > Mode (4), skewness > 0, phù hợp với phân bố lệch phải.

**Kết luận**: Tập dữ liệu có phân bố **lệch phải** (positively skewed), với đuôi dài ở các giá trị lớn và sự tập trung ở các giá trị nhỏ.

### 4. Lưu ý khi xác định phân bố

**Kích thước mẫu**: Mẫu nhỏ có thể không phản ánh đúng phân bố thực tế.

**Giá trị ngoại lai**: Có thể làm sai lệch hình dạng phân bố, cần xử lý trước (xem phương pháp xử lý giá trị thiếu hoặc ngoại lai).

**Biến đổi dữ liệu**: Nếu dữ liệu lệch mạnh, có thể áp dụng biến đổi (log, căn bậc hai) để đưa về gần phân bố chuẩn.

**Ngữ cảnh**: Phân bố phụ thuộc vào đặc điểm của biến (ví dụ, thu nhập thường lệch phải, điểm số có thể gần chuẩn).

### 5. Kết luận

**Xác định phân bố** cần kết hợp trực quan hóa (histogram, boxplot, Q-Q plot) và các chỉ số thống kê (trung bình, trung vị, skewness, kurtosis, kiểm định).

**Phân bố phổ biến**:

**Phân bố chuẩn**: Đối xứng, hình chuông, trung bình = trung vị = mode.

**Lệch phải**: Đuôi dài bên phải, trung bình > trung vị.

**Lệch trái**: Đuôi dài bên trái, trung bình < trung vị.

**Phân bố đều**: Xác suất đồng đều.

**Đa đỉnh**: Nhiều đỉnh, gợi ý nhiều nhóm dữ liệu.

**Lũy thừa**: Đuôi dài mạnh, ít giá trị lớn, nhiều giá trị nhỏ.

Hiểu rõ phân bố giúp lựa chọn phương pháp phân tích phù hợp (ví dụ, kiểm định tham số cho phân bố chuẩn, kiểm định phi tham số cho phân bố không chuẩn).

## + Độ lệch chuẩn và phạm vi (range) có ý nghĩa gì trong việc đánh giá sự phân tán của dữ liệu?

Độ lệch chuẩn và phạm vi (range) là hai thước đo quan trọng để đánh giá sự phân tán của dữ liệu trong thống kê, mỗi cái cung cấp một góc nhìn khác nhau về mức độ biến động của dữ liệu:

### Độ lệch chuẩn (Standard Deviation):

**Ý nghĩa**: Độ lệch chuẩn đo lường mức độ mà các giá trị trong tập dữ liệu lệch khỏi giá trị trung bình (mean). Nó cho biết dữ liệu phân tán chặt chẽ hay rộng rãi quanh giá trị trung bình.

**Cách hiểu**:

Nếu độ lệch chuẩn nhỏ, dữ liệu tập trung gần giá trị trung bình, nghĩa là sự phân tán thấp.

Nếu độ lệch chuẩn lớn, dữ liệu phân tán xa giá trị trung bình, nghĩa là sự phân tán cao.

**Ưu điểm**:

Phản ánh chính xác hơn sự phân tán vì nó xem xét tất cả các giá trị trong tập dữ liệu, không chỉ giá trị cực đại hay cực tiểu.

Nhạy với các biến động nhỏ trong dữ liệu.

**Ứng dụng**: Được sử dụng rộng rãi trong phân tích thống kê, tài chính, khoa học, v.v., đặc biệt khi cần so sánh mức độ biến động giữa các tập dữ liệu.

### Phạm vi (Range):

**Ý nghĩa**: Phạm vi là hiệu số giữa giá trị lớn nhất (max) và giá trị nhỏ nhất (min) trong tập dữ liệu. Nó cho biết khoảng cách từ giá trị thấp nhất đến cao nhất.

**Cách hiểu**:

Phạm vi lớn cho thấy dữ liệu có sự khác biệt lớn giữa các giá trị cực đại và cực tiểu.

Phạm vi nhỏ cho thấy các giá trị trong tập dữ liệu khá gần nhau.

**Ưu điểm**:

Dễ tính toán và dễ hiểu.

Cung cấp cái nhìn nhanh về mức độ lan tỏa của dữ liệu.

**Hạn chế**:

Chỉ dựa vào hai giá trị (max và min), nên không phản ánh được sự phân bố của các giá trị khác trong tập dữ liệu.

Nhạy cảm với các giá trị ngoại lai (outliers), có thể làm sai lệch bức tranh về sự phân tán.

### So sánh và ý nghĩa trong đánh giá sự phân tán:

**Độ lệch chuẩn** cung cấp thông tin chi tiết hơn về sự phân tán vì nó xem xét toàn bộ tập dữ liệu, trong khi **phạm vi** chỉ cho một cái nhìn tổng quát và đơn giản.

Ví dụ: Nếu hai tập dữ liệu có cùng phạm vi nhưng độ lệch chuẩn khác nhau, tập dữ liệu có độ lệch chuẩn nhỏ hơn sẽ có các giá trị tập trung hơn quanh trung bình.

Trong thực tế, độ lệch chuẩn thường được ưu tiên hơn phạm vi khi cần phân tích kỹ lưỡng, nhưng phạm vi hữu ích khi cần đánh giá nhanh hoặc khi dữ liệu có ít giá trị ngoại lai.

Tóm lại, cả hai thước đo đều quan trọng, nhưng độ lệch chuẩn thường được sử dụng trong các phân tích phức tạp, trong khi phạm vi phù hợp cho các đánh giá sơ bộ hoặc trực quan.

### Minh họa bằng 2 bộ dữ liệu có cùng Range nhưng SD khác nhau để thấy rõ sự khác biệt:

Tạo hai bộ dữ liệu có cùng phạm vi nhưng độ lệch chuẩn khác nhau, sau đó giải thích và trình bày trực quan để làm rõ sự phân tán.

**Hai bộ dữ liệu**

**Bộ dữ liệu 1**: [2, 3, 4, 5, 6]

**Bộ dữ liệu 2**: [2, 2, 4, 6, 6]

Cả hai bộ dữ liệu này có:

**Phạm vi (Range)**: Giá trị lớn nhất (6) - Giá trị nhỏ nhất (2) = **4**.

Tuy nhiên, sự phân tán của các giá trị quanh giá trị trung bình sẽ khác nhau, dẫn đến độ lệch chuẩn khác nhau.

**Bước 1: Tính toán các thông số**

**Bộ dữ liệu 1: [2, 3, 4, 5, 6]**

**Trung bình (Mean)**: (2 + 3 + 4 + 5 + 6) / 5 = 20 / 5 = 4



Độ lệch chuẩn (SD): Công thức:

Các độ lệch bình phương: (2-4)^2 = 4, (3-4)^2 = 1, (4-4)^2 = 0, (5-4)^2 = 1, (6-4)^2 = 4

Tổng: 4 + 1 + 0 + 1 + 4 = 10

Phương sai: 10/5 = 2

Độ lệch chuẩn: sqrt{2} ​≈1.414

**Bộ dữ liệu 2: [2, 2, 4, 6, 6]**

**Trung bình (Mean)**: (2 + 2 + 4 + 6 + 6) / 5 = 20 / 5 = 4

**Độ lệch chuẩn (SD)**:

Các độ lệch bình phương: (2-4)^2 = 4, (2-4)^2 = 4, (4-4)^2 = 0, (6-4)^2 = 4, (6-4)^2 = 4

Tổng: 4 + 4 + 0 + 4 + 4 = 16

Phương sai: 16 / 5 = 3.2

Độ lệch chuẩn: sqrt{3.2} ​≈1.789

**Kết quả**

**Phạm vi (Range)**: Cả hai bộ dữ liệu đều có phạm vi = 4.

**Độ lệch chuẩn (SD)**:

Bộ dữ liệu 1: ≈1.414\approx 1.414≈1.414

Bộ dữ liệu 2: ≈1.789\approx 1.789≈1.789

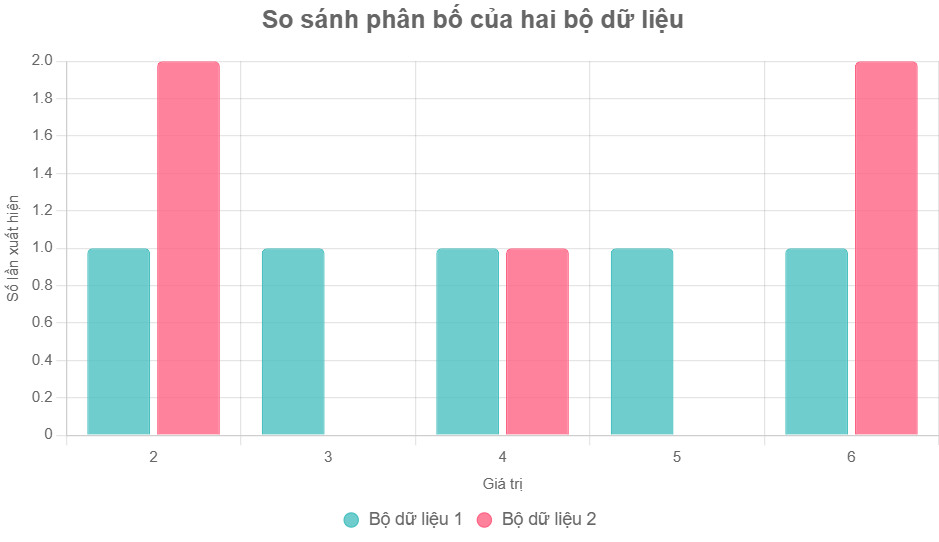
**Phân tích**

**Bộ dữ liệu 1** ([2, 3, 4, 5, 6]): Các giá trị phân bố đều quanh trung bình (4), khoảng cách giữa các giá trị khá đồng đều (tăng dần từng đơn vị). Điều này dẫn đến độ lệch chuẩn nhỏ hơn, cho thấy dữ liệu ít phân tán hơn.

**Bộ dữ liệu 2** ([2, 2, 4, 6, 6]): Các giá trị tập trung nhiều hơn ở hai đầu (2 và 6), với ít giá trị gần trung bình (4). Điều này làm tăng độ lệch chuẩn, cho thấy dữ liệu phân tán nhiều hơn, mặc dù phạm vi vẫn bằng 4.

**Minh hoạ trực quan:**

Dưới đây là biểu đồ cột (bar chart) minh họa hai bộ dữ liệu để thấy rõ sự khác biệt trong phân bố giá trị:



**Giải thích biểu đồ**

**Bộ dữ liệu 1**: Các cột có chiều cao đều (mỗi giá trị xuất hiện 1 lần), thể hiện sự phân bố đồng đều, dẫn đến độ lệch chuẩn nhỏ hơn.

**Bộ dữ liệu 2**: Các cột cao hơn ở hai đầu (2 và 6 xuất hiện 2 lần), cho thấy dữ liệu tập trung ở các giá trị cực đại và cực tiểu, dẫn đến độ lệch chuẩn lớn hơn.

**Kết luận**

Mặc dù cả hai bộ dữ liệu có cùng phạm vi (4), độ lệch chuẩn của **Bộ dữ liệu 2** lớn hơn, phản ánh sự phân tán cao hơn do các giá trị tập trung ở hai đầu. Điều này cho thấy **độ lệch chuẩn** cung cấp thông tin chi tiết hơn về sự phân tán so với **phạm vi**, vốn chỉ dựa vào giá trị lớn nhất và nhỏ nhất.

## + Sự khác biệt giữa các thước đo như Q1, Q2, Q3 trong biểu đồ hộp (boxplot) là gì?

Trong thống kê, các thước đo **Q1**, **Q2**, **Q3** là các **tứ phân vị** (quartiles) được sử dụng để mô tả sự phân bố của dữ liệu, đặc biệt trong **biểu đồ hộp (boxplot)**. Chúng giúp chia tập dữ liệu thành các phần bằng nhau và cung cấp thông tin về sự phân tán, trung tâm, và hình dạng của dữ liệu.

### 1. Định nghĩa các tứ phân vị

**Q1 (Tứ phân vị thứ nhất)**:

Là giá trị mà 25% dữ liệu nằm dưới nó (hay còn gọi là **tứ phân vị dưới**).

Q1 đại diện cho giá trị ở vị trí 25% của tập dữ liệu đã được sắp xếp theo thứ tự tăng dần.

**Q2 (Tứ phân vị thứ hai)**:

Là **trung vị** (median) của tập dữ liệu, tức là giá trị mà 50% dữ liệu nằm dưới và 50% nằm trên.

Q2 chia tập dữ liệu thành hai nửa bằng nhau và đại diện cho giá trị trung tâm.

**Q3 (Tứ phân vị thứ ba)**:

Là giá trị mà 75% dữ liệu nằm dưới nó (hay còn gọi là **tứ phân vị trên**).

Q3 đại diện cho giá trị ở vị trí 75% của tập dữ liệu đã được sắp xếp.

### 2. Vai trò trong biểu đồ hộp (Boxplot)

Biểu đồ hộp sử dụng Q1, Q2, Q3 cùng với các giá trị khác (như giá trị cực đại, cực tiểu, và giá trị ngoại lai) để biểu diễn sự phân bố của dữ liệu. Các thành phần chính của boxplot liên quan đến Q1, Q2, Q3 bao gồm:

**Hộp (Box)**: Được tạo bởi Q1 (cạnh dưới) và Q3 (cạnh trên), thể hiện **khoảng tứ phân vị (IQR)**, nơi chứa 50% dữ liệu ở giữa (từ 25% đến 75%).

**Đường giữa hộp**: Là Q2 (trung vị), biểu thị giá trị trung tâm của tập dữ liệu.

**Râu (Whiskers)**: Thường kéo dài từ Q1 đến giá trị nhỏ nhất và từ Q3 đến giá trị lớn nhất trong phạm vi không bị coi là ngoại lai (thường tính bằng Q1−1.5×IQR Q1 - 1.5 \times IQR Q1−1.5×IQR và Q3+1.5×IQR Q3 + 1.5 \times IQR Q3+1.5×IQR).

**Giá trị ngoại lai (Outliers)**: Các điểm dữ liệu nằm ngoài phạm vi của râu, tức là nhỏ hơn Q1−1.5×IQR Q1 - 1.5 \times IQR Q1−1.5×IQR hoặc lớn hơn Q3+1.5×IQR Q3 + 1.5 \times IQR Q3+1.5×IQR.

### 3. Sự khác biệt giữa Q1, Q2, Q3

**Vị trí trong phân bố**:

**Q1**: Đánh dấu ranh giới giữa 25% dữ liệu thấp nhất và 75% còn lại.

**Q2**: Là trung tâm của dữ liệu, chia đôi tập dữ liệu.

**Q3**: Đánh dấu ranh giới giữa 75% dữ liệu thấp hơn và 25% dữ liệu cao nhất.

**Ý nghĩa về phân tán**:

**Khoảng tứ phân vị (IQR = Q3 - Q1)**: Đo lường mức độ phân tán của 50% dữ liệu ở giữa. IQR lớn hơn cho thấy dữ liệu phân tán rộng hơn.

**Q2**: Không chỉ là trung tâm mà còn giúp đánh giá tính đối xứng của phân bố (nếu Q2 gần giữa Q1 và Q3, dữ liệu có thể đối xứng; nếu lệch, dữ liệu có thể bị lệch).

**Ứng dụng**:

Q1 và Q3 giúp xác định phạm vi dữ liệu chính và phát hiện giá trị ngoại lai.

Q2 cung cấp thông tin về giá trị trung tâm, ít bị ảnh hưởng bởi các giá trị cực đại hơn trung bình cộng.

Cả ba tứ phân vị giúp so sánh sự phân bố giữa các tập dữ liệu khác nhau.

### 4. Ví dụ minh họa

Giả sử một tập dữ liệu: [2, 3, 4, 5, 6, 7, 8, 9, 10]

Sắp xếp theo thứ tự tăng dần: [2, 3, 4, 5, 6, 7, 8, 9, 10]

**Q1**: Giá trị tại vị trí 25% (khoảng vị trí 2.5) → Trung bình của 3 và 4 = **3.5**

**Q2 (trung vị)**: Giá trị tại vị trí 50% (vị trí 5) = **6**

**Q3**: Giá trị tại vị trí 75% (khoảng vị trí 7.5) → Trung bình của 8 và 9 = **8.5**

**IQR**: Q3−Q1=8.5−3.5=5 Q3 - Q1 = 8.5 - 3.5 = 5 Q3−Q1=8.5−3.5=5

Trong biểu đồ hộp:

Hộp kéo dài từ 3.5 (Q1) đến 8.5 (Q3).

Đường giữa hộp ở 6 (Q2).

Râu kéo dài từ giá trị nhỏ nhất (2) đến lớn nhất (10), trừ khi có ngoại lai.

### 5. Ý nghĩa trong phân tích dữ liệu

**Q1, Q2, Q3** cung cấp cái nhìn tổng quan về sự phân bố mà không bị ảnh hưởng nhiều bởi giá trị ngoại lai, không giống như trung bình cộng hay độ lệch chuẩn.

**Boxplot** sử dụng các tứ phân vị để trực quan hóa sự phân tán, đối xứng, và phát hiện các giá trị bất thường.

**So sánh tập dữ liệu**: Khi so sánh nhiều tập dữ liệu, Q1, Q2, Q3 trong boxplot giúp dễ dàng nhận ra sự khác biệt về trung tâm, độ phân tán, và hình dạng phân bố.

**Kết luận**

**Q1**: Đại diện cho 25% dữ liệu thấp nhất, là cạnh dưới của hộp.

**Q2**: Là trung vị, biểu thị giá trị trung tâm và chia đôi dữ liệu.

**Q3**: Đại diện cho 75% dữ liệu thấp hơn, là cạnh trên của hộp.

Cùng nhau, chúng tạo nên biểu đồ hộp, cung cấp một cách trực quan và hiệu quả để đánh giá sự phân bố, phân tán, và các đặc điểm khác của dữ liệu.

## + Làm thế nào để xử lý giá trị thiếu (missing values) trước khi tính toán các chỉ số thống kê mô tả?

Xử lý giá trị thiếu (missing values) là một bước quan trọng trước khi tính toán các chỉ số thống kê mô tả (như trung bình, trung vị, độ lệch chuẩn, tứ phân vị, phạm vi, v.v.) để đảm bảo kết quả chính xác và không bị sai lệch.

### 1. Hiểu bản chất của giá trị thiếu

Trước khi xử lý, cần xác định:

**Loại giá trị thiếu**:

**MCAR (Missing Completely at Random)**: Giá trị thiếu không phụ thuộc vào bất kỳ yếu tố nào (ngẫu nhiên hoàn toàn).

**MAR (Missing at Random)**: Giá trị thiếu phụ thuộc vào các biến khác trong dữ liệu nhưng không phụ thuộc vào chính giá trị bị thiếu.

**MNAR (Missing Not at Random)**: Giá trị thiếu liên quan trực tiếp đến giá trị của chính nó.

**Tỷ lệ giá trị thiếu**: Nếu tỷ lệ thiếu nhỏ (ví dụ <5%), xử lý đơn giản có thể đủ. Nếu tỷ lệ lớn, cần phương pháp phức tạp hơn.

**Mục đích phân tích**: Tùy thuộc vào mục tiêu (ví dụ, tính trung bình hay xây dựng mô hình), phương pháp xử lý sẽ khác nhau.

### 2. Các phương pháp xử lý giá trị thiếu

Dưới đây là các phương pháp phổ biến:

**a. Loại bỏ giá trị thiếu (Deletion)**

**Cách thực hiện**:

**Listwise deletion (Loại bỏ toàn bộ hàng)**: Xóa toàn bộ hàng (bản ghi) chứa giá trị thiếu.

**Pairwise deletion**: Chỉ loại bỏ giá trị thiếu trong các tính toán cụ thể, giữ lại dữ liệu cho các phép tính khác.

**Khi sử dụng**:

Phù hợp khi tỷ lệ giá trị thiếu nhỏ và dữ liệu bị thiếu là MCAR.

Ví dụ: Nếu một tập dữ liệu có 100 hàng và chỉ 2 hàng thiếu giá trị, loại bỏ 2 hàng này thường không ảnh hưởng lớn.

**Ưu điểm**:

Đơn giản, dễ thực hiện.

Không cần giả định về dữ liệu.

**Nhược điểm**:

Làm giảm kích thước tập dữ liệu, có thể dẫn đến mất thông tin quan trọng.

Không phù hợp nếu dữ liệu thiếu không ngẫu nhiên (MAR hoặc MNAR) hoặc tỷ lệ thiếu lớn.

**b. Thay thế bằng giá trị thống kê (Imputation with Statistical Values)**

**Cách thực hiện**:

**Thay bằng trung bình (Mean Imputation)**: Thay giá trị thiếu bằng trung bình của cột.

**Thay bằng trung vị (Median Imputation)**: Thay giá trị thiếu bằng trung vị, phù hợp với dữ liệu bị lệch (skewed).

**Thay bằng mode (Mode Imputation)**: Thay giá trị thiếu bằng giá trị xuất hiện nhiều nhất, thường dùng cho dữ liệu phân loại (categorical).

**Khi sử dụng**:

Phù hợp khi dữ liệu thiếu là MCAR hoặc MAR và tỷ lệ thiếu không quá lớn.

Ví dụ: Trong một cột điểm số, thay các giá trị thiếu bằng trung bình hoặc trung vị của cột đó.

**Ưu điểm**:

Giữ nguyên kích thước tập dữ liệu.

Đơn giản và nhanh chóng.

**Nhược điểm**:

Làm giảm sự biến thiên của dữ liệu (ví dụ, thay bằng trung bình làm giảm độ lệch chuẩn).

Không phản ánh tốt mối quan hệ giữa các biến.

Có thể gây sai lệch nếu dữ liệu bị lệch hoặc có giá trị ngoại lai.

**c. Thay thế bằng giá trị dự đoán (Predictive Imputation)**

**Cách thực hiện**:

Sử dụng các thuật toán học máy (như hồi quy tuyến tính, KNN, hoặc Random Forest) để dự đoán giá trị thiếu dựa trên các biến khác trong tập dữ liệu.

Ví dụ: Dùng mô hình KNN để tìm các hàng tương tự và thay giá trị thiếu bằng giá trị trung bình của các hàng lân cận.

**Khi sử dụng**:

Phù hợp với dữ liệu MAR, khi có mối quan hệ rõ ràng giữa các biến.

Thường dùng khi dữ liệu phức tạp và tỷ lệ giá trị thiếu đáng kể.

**Ưu điểm**:

Chính xác hơn vì xem xét mối quan hệ giữa các biến.

Phù hợp với dữ liệu phức tạp.

**Nhược điểm**:

Phức tạp, đòi hỏi thời gian và tài nguyên tính toán.

Có thể dẫn đến sai lệch nếu mô hình dự đoán không phù hợp.

**d. Thay thế bằng giá trị cố định (Constant Imputation)**

**Cách thực hiện**:

Thay giá trị thiếu bằng một giá trị cố định, ví dụ 0, một giá trị cụ thể, hoặc một nhãn như “Unknown” cho dữ liệu phân loại.

**Khi sử dụng**:

Phù hợp khi giá trị thiếu mang ý nghĩa cụ thể (ví dụ, “0” có thể biểu thị không có giá trị trong một số ngữ cảnh).

**Ưu điểm**:

Đơn giản và giữ nguyên kích thước dữ liệu.

**Nhược điểm**:

Có thể làm sai lệch phân bố dữ liệu.

Chỉ phù hợp trong một số trường hợp cụ thể.

**e. Sử dụng giá trị nội suy (Interpolation)**

**Cách thực hiện**:

Ước lượng giá trị thiếu dựa trên các giá trị lân cận, thường áp dụng cho dữ liệu chuỗi thời gian.

Các phương pháp: Nội suy tuyến tính, nội suy đa thức, hoặc nội suy spline.

**Khi sử dụng**:

Phù hợp với dữ liệu có tính liên tục, như chuỗi thời gian (ví dụ, nhiệt độ hàng ngày).

**Ưu điểm**:

Phù hợp với dữ liệu có xu hướng hoặc mẫu rõ ràng.

Giữ được tính liên tục của dữ liệu.

**Nhược điểm**:

Không phù hợp với dữ liệu không liên tục hoặc dữ liệu phân loại.

Kết quả có thể không chính xác nếu xu hướng không rõ ràng.

**f. Giữ nguyên giá trị thiếu**

**Cách thực hiện**:

Không xử lý giá trị thiếu mà sử dụng các thuật toán thống kê hoặc mô hình học máy có khả năng xử lý giá trị thiếu (như một số cây quyết định hoặc mô hình XGBoost).

**Khi sử dụng**:

Phù hợp khi thuật toán phân tích có thể tự xử lý giá trị thiếu.

Hoặc khi việc thay thế giá trị thiếu có thể làm sai lệch dữ liệu.

**Ưu điểm**:

Tránh làm sai lệch dữ liệu do thay thế không chính xác.

**Nhược điểm**:

Không phải tất cả các chỉ số thống kê (như trung bình, độ lệch chuẩn) có thể tính toán trực tiếp với giá trị thiếu.

Yêu cầu công cụ/phương pháp hỗ trợ.

### 3. Quy trình xử lý giá trị thiếu trước khi tính toán thống kê

**Xác định giá trị thiếu**:

Kiểm tra tập dữ liệu để xác định tỷ lệ và mô hình của giá trị thiếu (dùng các công cụ như Python với pandas.isna() hoặc R với is.na()).

**Đánh giá tác động**:

Xác định xem giá trị thiếu có ảnh hưởng lớn đến chỉ số thống kê hay không.

Ví dụ: Trung bình và độ lệch chuẩn nhạy với giá trị thiếu, trong khi trung vị và tứ phân vị ít bị ảnh hưởng hơn.

**Chọn phương pháp phù hợp**:

Nếu tỷ lệ thiếu nhỏ và ngẫu nhiên (MCAR), có thể dùng loại bỏ hoặc thay bằng trung bình/trung vị.

Nếu tỷ lệ thiếu lớn hoặc dữ liệu MAR/MNAR, cân nhắc dự đoán hoặc nội suy.

**Kiểm tra sau xử lý**:

So sánh phân bố dữ liệu trước và sau khi xử lý (dùng biểu đồ như histogram, boxplot).

Đảm bảo các chỉ số thống kê (như Q1, Q2, Q3, độ lệch chuẩn) không bị sai lệch nghiêm trọng.

**Ghi nhận phương pháp xử lý**:

Ghi lại cách xử lý giá trị thiếu để đảm bảo tính minh bạch và tái lập trong phân tích.

### 4. Ví dụ minh họa

Giả sử tập dữ liệu: [2, 4, NaN, 6, 8]

**Loại bỏ hàng chứa NaN**:

Dữ liệu còn lại: [2, 4, 6, 8]

Trung bình: (2+4+6+8)/4=5(2 + 4 + 6 + 8) / 4 = 5(2+4+6+8)/4=5

Trung vị: 5

Độ lệch chuẩn: ≈2.582\approx 2.582≈2.582

**Thay bằng trung bình**:

Trung bình của [2, 4, 6, 8] = 5, thay NaN bằng 5.

Dữ liệu mới: [2, 4, 5, 6, 8]

Trung bình: 5

Trung vị: 5

Độ lệch chuẩn: ≈2.236\approx 2.236≈2.236 (nhỏ hơn do dữ liệu ít phân tán hơn).

**Thay bằng nội suy tuyến tính** (nếu là chuỗi thời gian):

NaN nằm giữa 4 và 6, nội suy tuyến tính: (4+6)/2=5(4 + 6) / 2 = 5(4+6)/2=5

Kết quả tương tự như thay bằng trung bình trong trường hợp này.

### 5. Lưu ý khi tính toán các chỉ số thống kê

**Trung bình và độ lệch chuẩn**: Nhạy cảm với cách xử lý giá trị thiếu, đặc biệt nếu thay bằng trung bình hoặc trung vị, vì có thể làm giảm sự biến thiên.

**Trung vị, Q1, Q3**: Ít nhạy hơn, nhưng nếu loại bỏ quá nhiều dữ liệu, các giá trị này có thể không đại diện cho tập dữ liệu gốc.

**Phạm vi (Range)**: Có thể bị ảnh hưởng nếu giá trị thiếu nằm ở các cực trị (max/min).

**Boxplot**: Xử lý giá trị thiếu trước khi vẽ để đảm bảo Q1, Q2, Q3 và râu được tính toán chính xác.

### 6. Kết luận

Không có phương pháp xử lý giá trị thiếu nào là tối ưu cho mọi trường hợp. Lựa chọn phương pháp phụ thuộc vào bản chất dữ liệu, tỷ lệ giá trị thiếu, và mục tiêu phân tích.

**Đề xuất**:

Nếu tỷ lệ thiếu nhỏ và ngẫu nhiên, **loại bỏ** hoặc **thay bằng trung bình/trung vị** là đủ.

Nếu dữ liệu phức tạp hoặc tỷ lệ thiếu lớn, cân nhắc **dự đoán bằng mô hình** hoặc **nội suy**.

Luôn kiểm tra tác động của phương pháp xử lý lên các chỉ số thống kê và phân bố dữ liệu để đảm bảo kết quả đáng tin cậy.

## + Bạn có thể giải thích cách đọc và diễn giải một biểu đồ histogram hoặc boxplot từ dữ liệu thực tế không?

**Tập dữ liệu thực tế**

Giả sử chúng ta có điểm số của 50 học sinh trong một bài kiểm tra, với thang điểm từ 0 đến 100:

Dữ liệu: [45, 50, 55, 60, 62, 65, 65, 68, 70, 70, 72, 75, 75, 78, 80, 80, 82, 85, 85, 85, 88, 90, 90, 92, 95, 95, 98, 100, 60, 62, 65, 68, 70, 72, 75, 78, 80, 82, 85, 88, 90, 92, 95, 50, 55, 60, 65, 70, 75, 80, 85]

Dữ liệu này có thể đại diện cho điểm số thực tế trong một lớp học, với một số học sinh đạt điểm cao, một số đạt điểm trung bình, và một số ít điểm thấp.

### 1. Biểu đồ Histogram

**a. Vẽ Histogram**

Histogram chia dữ liệu thành các khoảng (bins) và hiển thị tần suất (số lần xuất hiện) của các giá trị trong mỗi khoảng. Tôi sẽ chia dữ liệu thành các khoảng điểm (ví dụ: 40-50, 50-60, ..., 90-100) để vẽ histogram.

Dựa trên dữ liệu, tần suất trong các khoảng như sau:

40-50: 2 (45, 50)

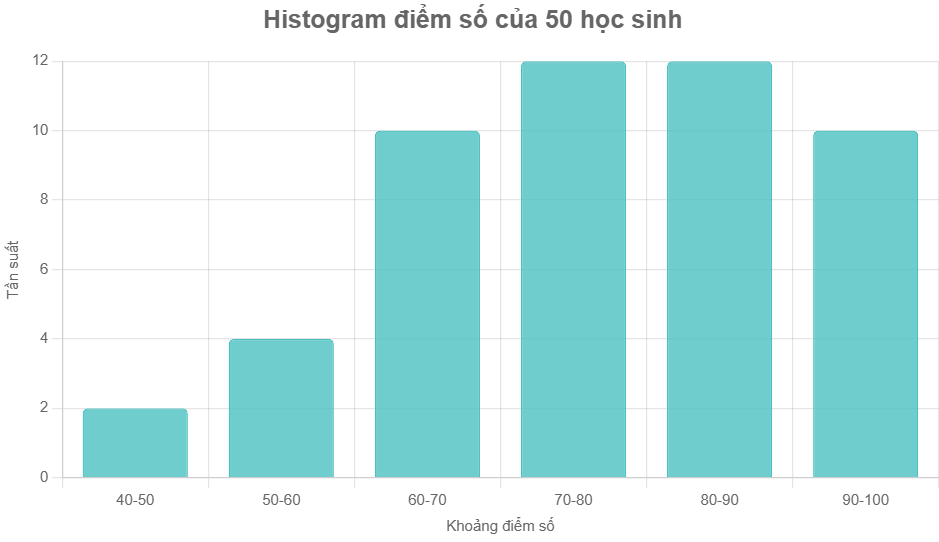
50-60: 4 (50, 55, 55, 60)

60-70: 10 (60, 60, 62, 62, 65, 65, 65, 68, 68, 70)

70-80: 12 (70, 70, 72, 72, 75, 75, 75, 78, 78, 80, 80, 80)

80-90: 12 (80, 82, 82, 85, 85, 85, 85, 88, 88)

90-100: 10 (90, 90, 92, 92, 95, 95, 95, 98, 100)



**b. Cách đọc Histogram**

**Trục X (ngang)**: Đại diện cho các khoảng giá trị của dữ liệu (điểm số từ 40 đến 100, chia thành các khoảng 10 điểm).

**Trục Y (dọc)**: Đại diện cho tần suất, tức là số lần xuất hiện của điểm số trong mỗi khoảng.

**Hình dạng phân bố**:

Histogram cho thấy tần suất cao nhất ở khoảng 70-80 và 80-90 (mỗi khoảng có 12 học sinh), tiếp theo là 60-70 và 90-100 (10 học sinh mỗi khoảng).

Có ít học sinh ở khoảng thấp (40-50: 2 học sinh) và khoảng 50-60 (4 học sinh).

Phân bố hơi **lệch trái** vì có nhiều giá trị tập trung ở phía điểm cao (70-100) và đuôi dài hơn ở phía điểm thấp (40-60).

**c. Diễn giải Histogram**

**Hình dạng phân bố**: Phân bố không hoàn toàn đối xứng, có xu hướng **lệch trái** nhẹ (negative skew) vì đuôi bên trái (điểm thấp) dài hơn một chút so với bên phải.

**Tập trung dữ liệu**: Phần lớn học sinh đạt điểm từ 70 đến 90, cho thấy đa số học sinh có kết quả tốt hoặc trung bình khá.

**Độ phân tán**: Dữ liệu trải dài từ 40 đến 100, nhưng tập trung chủ yếu ở khoảng 60-100, cho thấy bài kiểm tra có thể không quá khó (ít học sinh điểm rất thấp).

**Bối cảnh thực tế**: Nếu đây là bài kiểm tra dễ, phân bố lệch trái là hợp lý vì nhiều học sinh đạt điểm cao.

### 2. Biểu đồ Boxplot

**a. Tính toán các thành phần Boxplot**

Để vẽ boxplot, cần tính các giá trị: Q1, Q2, Q3, IQR, râu, và giá trị ngoại lai (nếu có).

**Sắp xếp dữ liệu**: [45, 50, 50, 55, 55, 60, 60, 60, 62, 62, 65, 65, 65, 65, 68, 68, 70, 70, 70, 70, 72, 72, 75, 75, 75, 75, 78, 78, 80, 80, 80, 80, 82, 82, 85, 85, 85, 85, 88, 88, 90, 90, 92, 92, 95, 95, 95, 98, 100]

**Q1 (tứ phân vị thứ nhất)**: Vị trí 25% (khoảng vị trí 12.5) → Trung bình của giá trị thứ 12 và 13 = (65 + 65) / 2 = **65**.

**Q2 (trung vị)**: Vị trí 50% (khoảng vị trí 25.5) → Trung bình của giá trị thứ 25 và 26 = (75 + 75) / 2 = **75**.

**Q3 (tứ phân vị thứ ba)**: Vị trí 75% (khoảng vị trí 37.5) → Trung bình của giá trị thứ 37 và 38 = (85 + 85) / 2 = **85**.

**IQR (khoảng tứ phân vị)**: Q3−Q1=85−65=20 Q3 - Q1 = 85 - 65 = 20 Q3−Q1=85−65=20.

**Râu**:

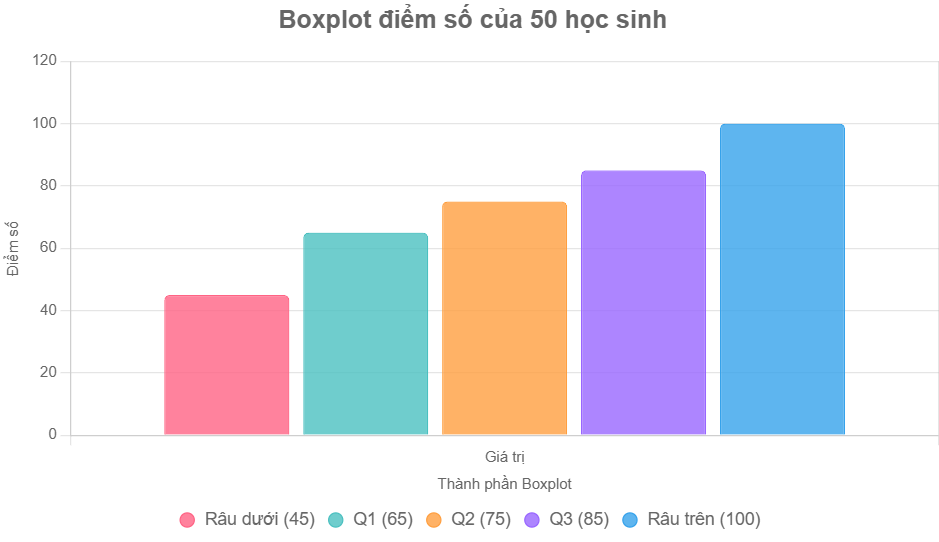
Râu dưới: Q1−1.5×IQR=65−1.5×20=65−30=35 → Giá trị nhỏ nhất là 45 (> 35), nên râu dưới là **45**.

Râu trên: Q3+1.5×IQR=85+1.5×20=85+30=115 → Giá trị lớn nhất là 100 (< 115), nên râu trên là **100**.

**Giá trị ngoại lai**: Không có giá trị nào nằm ngoài khoảng [35, 115], nên không có ngoại lai.

**b. Vẽ Boxplot**

Vì Chart.js không hỗ trợ boxplot trực tiếp, tôi sẽ mô phỏng bằng cách hiển thị các thành phần chính (Q1, Q2, Q3, râu) dưới dạng biểu đồ cột với chú thích.



**c. Cách đọc Boxplot**

**Hộp (Box)**: Kéo dài từ Q1 (65) đến Q3 (85), chứa 50% dữ liệu ở giữa (từ 25% đến 75% của điểm số).

**Đường giữa hộp (Q2)**: Trung vị ở 75, nghĩa là 50% học sinh đạt điểm dưới 75 và 50% đạt điểm trên 75.

**Râu (Whiskers)**:

Râu dưới: Từ 45 (điểm thấp nhất) đến Q1 (65).

Râu trên: Từ Q3 (85) đến 100 (điểm cao nhất).

**Ngoại lai**: Không có giá trị nào nằm ngoài khoảng [35, 115], nên không có ngoại lai.

**Độ lệch**:

Trung vị (75) gần Q3 (85) hơn Q1 (65), cho thấy dữ liệu hơi **lệch trái** (khoảng cách từ Q2 đến Q1 lớn hơn từ Q2 đến Q3).

**d. Diễn giải Boxplot**

**Trung tâm dữ liệu**: Trung vị là 75, cho thấy điểm số trung bình của lớp nằm ở mức khá tốt.

**Độ phân tán**: IQR = 20 (từ 65 đến 85) cho thấy 50% học sinh có điểm số nằm trong khoảng này, tương đối tập trung.

**Độ lệch**: Trung vị gần Q3 hơn Q1, xác nhận phân bố lệch trái nhẹ, với nhiều học sinh đạt điểm cao hơn trung vị.

**Phạm vi**: Điểm số trải dài từ 45 đến 100, cho thấy sự khác biệt đáng kể giữa học sinh thấp nhất và cao nhất.

**Ngoại lai**: Không có ngoại lai, nghĩa là không có điểm số nào bất thường so với phân bố chung.

**Bối cảnh thực tế**: Boxplot cho thấy bài kiểm tra có thể không quá khó, vì phần lớn học sinh đạt điểm từ 65 trở lên, và không có học sinh nào đạt điểm cực thấp (dưới 35).

### 3. So sánh và diễn giải tổng hợp

**Histogram**:

Cho thấy phân bố tổng quát, với tần suất cao ở khoảng 70-90, xác nhận đa số học sinh đạt điểm khá.

Đuôi dài hơn ở phía điểm thấp (40-60) cho thấy phân bố lệch trái nhẹ.

Giúp dễ dàng nhận ra các khoảng điểm phổ biến và mức độ tập trung của dữ liệu.

**Boxplot**:

Cung cấp thông tin chi tiết về trung tâm (trung vị = 75), độ phân tán (IQR = 20), và phạm vi (45-100).

Xác nhận phân bố lệch trái qua vị trí của trung vị (gần Q3 hơn Q1).

Hữu ích để phát hiện ngoại lai (trong trường hợp này không có).

**Tính toán bổ sung để xác nhận phân bố:**

**Trung bình**: (45+50+...+100)/50≈74.6(45 + 50 + ... + 100) / 50 \approx 74.6(45+50+...+100)/50≈74.6

**Trung vị**: 75

**Mode**: 85 (xuất hiện 4 lần)

**Skewness**: Trung bình (74.6) < Trung vị (75) < Mode (85), xác nhận phân bố **lệch trái** (negative skew).

### 4. Diễn giải trong bối cảnh thực tế

**Kết quả học tập**: Phân bố lệch trái cho thấy bài kiểm tra có thể dễ hoặc học sinh được chuẩn bị tốt, vì nhiều học sinh đạt điểm cao (70-100) và ít học sinh đạt điểm thấp (40-60).

**Hành động cải thiện**: Giáo viên có thể tập trung hỗ trợ một số ít học sinh có điểm thấp (dưới 60) để cải thiện kết quả tổng thể.

**So sánh với các lớp khác**: Histogram và boxplot có thể được dùng để so sánh phân bố điểm số giữa các lớp hoặc các kỳ thi khác, giúp đánh giá mức độ khó của bài kiểm tra hoặc chất lượng giảng dạy.

### 5. Lưu ý khi đọc và diễn giải

**Histogram**:

Kích thước khoảng (bin size) ảnh hưởng đến hình dạng histogram. Nếu khoảng quá lớn hoặc quá nhỏ, phân bố có thể bị méo mó.

Cần xem xét bối cảnh dữ liệu để diễn giải (ví dụ, điểm số lệch trái có thể do bài kiểm tra dễ).

**Boxplot**:

Nhạy với giá trị ngoại lai, nên cần kiểm tra xem ngoại lai có ý nghĩa thực tế hay là lỗi dữ liệu.

Vị trí trung vị so với Q1 và Q3 giúp xác định độ lệch nhanh chóng.

**Kết hợp cả hai**: Histogram cung cấp cái nhìn tổng quát về hình dạng phân bố, trong khi boxplot chi tiết hơn về trung tâm, phân tán, và ngoại lai.

## + Khi gặp một tập dữ liệu có giá trị ngoại lai (outliers), bạn sẽ xử lý chúng như thế nào trước khi thực hiện thống kê mô tả?

Giá trị ngoại lai (outliers) là những điểm dữ liệu khác biệt đáng kể so với phần còn lại của tập dữ liệu, có thể làm sai lệch các chỉ số thống kê mô tả như trung bình, độ lệch chuẩn, hoặc phạm vi. Xử lý giá trị ngoại lai trước khi tính toán thống kê mô tả là một bước quan trọng để đảm bảo kết quả phản ánh chính xác đặc điểm của tập dữ liệu. Dưới đây là các bước và phương pháp xử lý giá trị ngoại lai, cùng với hướng dẫn cách áp dụng trong bối cảnh thực tế.

### 1. Xác định giá trị ngoại lai

Trước khi xử lý, cần xác định giá trị ngoại lai bằng các phương pháp sau:

**a. Sử dụng biểu đồ trực quan**

**Boxplot**:

Giá trị ngoại lai thường được xác định là các điểm nằm ngoài khoảng [Q1−1.5×IQR, Q3+1.5×IQR], trong đó IQR = Q3 - Q1 là khoảng tứ phân vị.

Ví dụ: Nếu Q1=20, Q3 = 50, thì IQR = 30, và giá trị ngoại lai là các điểm < 20 − 1.5 × 30 = −25 hoặc > 50 + 1.5 × 30 = 95.

**Histogram hoặc Scatter Plot**:

Giúp nhận diện các giá trị bất thường bằng cách quan sát các điểm xa rời khỏi phần lớn dữ liệu hoặc các đỉnh bất thường trong phân bố.

**b. Phương pháp thống kê**

**Z-score (Điểm chuẩn hóa)**:

Tính Z = (x−μ) / σ​, với μ là trung bình và σ là độ lệch chuẩn.

Giá trị có |Z| > 3 (hoặc ngưỡng khác, ví dụ 2.5) thường được coi là ngoại lai, vì chúng nằm ngoài 99.7% dữ liệu trong phân bố chuẩn.

**Phương pháp IQR**:

Như đã đề cập trong boxplot, các giá trị ngoài [Q1−1.5×IQR,Q3+1.5×IQR] là ngoại lai.

**Kiểm định thống kê**: Các kiểm định như Grubbs’ Test hoặc Dixon’s Q Test có thể được dùng để xác định ngoại lai trong các tập dữ liệu nhỏ.

**c. Hiểu bối cảnh dữ liệu**

Xác định xem giá trị ngoại lai có ý nghĩa thực tế hay không (ví dụ, điểm số 100 trong bài kiểm tra là hợp lý, nhưng 1000 là lỗi nhập liệu).

Kiểm tra xem ngoại lai có phải do lỗi đo lường, nhập liệu sai, hay phản ánh một hiện tượng thực tế (như thu nhập cực cao của một cá nhân trong tập dữ liệu).

### 2. Các phương pháp xử lý giá trị ngoại lai

Sau khi xác định, có nhiều cách để xử lý giá trị ngoại lai, tùy thuộc vào mục đích phân tích và bản chất của dữ liệu. Dưới đây là các phương pháp phổ biến:

**a. Loại bỏ giá trị ngoại lai (Deletion)**

**Cách thực hiện**:

Xóa các điểm dữ liệu được xác định là ngoại lai khỏi tập dữ liệu.

Ví dụ: Nếu giá trị 1000 trong tập điểm số [45, 50, ..., 100, 1000] là lỗi, loại bỏ 1000.

**Khi sử dụng**:

Phù hợp khi ngoại lai rõ ràng là lỗi nhập liệu hoặc không có ý nghĩa thực tế.

Tỷ lệ ngoại lai nhỏ (ví dụ, <5% dữ liệu).

**Ưu điểm**:

Đơn giản, đảm bảo các chỉ số thống kê không bị sai lệch bởi các giá trị bất thường.

**Nhược điểm**:

Làm giảm kích thước mẫu, có thể mất thông tin quan trọng nếu ngoại lai mang ý nghĩa thực tế.

Không phù hợp nếu ngoại lai là một phần tự nhiên của dữ liệu (ví dụ, thu nhập cao trong phân tích kinh tế).

**b. Thay thế giá trị ngoại lai (Imputation)**

**Cách thực hiện**:

Thay giá trị ngoại lai bằng một giá trị hợp lý, như:

**Trung bình hoặc trung vị**: Thay ngoại lai bằng trung bình hoặc trung vị của tập dữ liệu (không tính ngoại lai).

**Ranh giới IQR**: Thay ngoại lai bằng giá trị gần nhất trong khoảng [Q1−1.5×IQR,Q3+1.5×IQR] (ví dụ, thay giá trị > Q3+1.5×IQR bằng Q3+1.5×IQR.

**Dự đoán**: Sử dụng mô hình học máy (như KNN, hồi quy) để dự đoán giá trị thay thế dựa trên các biến khác.

**Khi sử dụng**:

Khi muốn giữ nguyên kích thước tập dữ liệu.

Phù hợp với dữ liệu có ngoại lai do lỗi đo lường hoặc khi ngoại lai không quá quan trọng.

**Ưu điểm**:

Giữ nguyên số lượng quan sát.

Giảm tác động của ngoại lai lên các chỉ số như trung bình và độ lệch chuẩn.

**Nhược điểm**:

Có thể làm sai lệch phân bố nếu thay thế không chính xác.

Làm giảm sự biến thiên tự nhiên của dữ liệu.

**c. Biến đổi dữ liệu (Transformation)**

**Cách thực hiện**:

Áp dụng biến đổi toán học để giảm tác động của ngoại lai, ví dụ:

**Log transformation**: log(x) hoặc log(x+1) để giảm độ lớn của các giá trị cực đại.

**Căn bậc hai**: sqrt{x} để nén các giá trị lớn.

**Chuẩn hóa (Standardization)**: Chuyển dữ liệu về dạng Z-score để giảm ảnh hưởng của các giá trị xa.

Ví dụ: Nếu tập dữ liệu có giá trị ngoại lai 1000, áp dụng log(1000) = 3 sẽ làm giá trị này gần hơn với các giá trị khác.

**Khi sử dụng**:

Phù hợp với dữ liệu lệch mạnh (như thu nhập, thời gian chờ) hoặc khi ngoại lai là một phần tự nhiên của dữ liệu.

**Ưu điểm**:

Giữ nguyên tất cả dữ liệu và giảm tác động của ngoại lai mà không cần xóa.

Giúp dữ liệu gần hơn với phân bố chuẩn, thuận lợi cho phân tích thống kê.

**Nhược điểm**:

Có thể làm phức tạp việc diễn giải kết quả (ví dụ, giá trị log không trực quan như giá trị gốc).

Không phù hợp với dữ liệu có giá trị âm hoặc bằng 0 (trừ khi điều chỉnh).

**d. Giữ nguyên giá trị ngoại lai**

**Cách thực hiện**:

Không xử lý ngoại lai mà phân tích dữ liệu với sự hiện diện của chúng, sử dụng các chỉ số thống kê ít nhạy cảm với ngoại lai (như trung vị, IQR thay vì trung bình, độ lệch chuẩn).

**Khi sử dụng**:

Khi ngoại lai có ý nghĩa thực tế quan trọng (ví dụ, thu nhập cực cao của một CEO trong phân tích tài chính).

Khi sử dụng các phương pháp thống kê phi tham số hoặc mô hình học máy có khả năng xử lý ngoại lai (như cây quyết định, Random Forest).

**Ưu điểm**:

Bảo toàn thông tin gốc của dữ liệu.

Phù hợp với các phân tích cần giữ nguyên đặc điểm tự nhiên của dữ liệu.

**Nhược điểm**:

Các chỉ số như trung bình và độ lệch chuẩn có thể bị sai lệch nghiêm trọng.

Cần chọn phương pháp phân tích phù hợp để giảm thiểu tác động của ngoại lai.

**e. Phân tích riêng biệt (Segmentation)**

**Cách thực hiện**:

Tách tập dữ liệu thành hai nhóm: nhóm không có ngoại lai và nhóm chứa ngoại lai, sau đó phân tích riêng.

Ví dụ: Phân tích điểm số học sinh thông thường và học sinh có điểm cực cao/thấp riêng biệt.

**Khi sử dụng**:

Khi ngoại lai đại diện cho một nhóm đặc biệt (ví dụ, học sinh xuất sắc hoặc yếu kém).

**Ưu điểm**:

Cho phép phân tích chi tiết từng nhóm, giữ được ý nghĩa của ngoại lai.

**Nhược điểm**:

Làm phức tạp phân tích, đặc biệt với tập dữ liệu nhỏ.

Có thể không phù hợp nếu ngoại lai không thuộc một nhóm đặc biệt.

### 3. Quy trình xử lý giá trị ngoại lai trước khi tính toán thống kê mô tả

Dưới đây là quy trình tổng quát:

**Xác định ngoại lai**:

Sử dụng boxplot, Z-score, hoặc IQR để tìm các giá trị ngoại lai.

Kiểm tra bối cảnh: Ngoại lai là lỗi hay có ý nghĩa thực tế?

**Đánh giá tác động**:

Tính thử các chỉ số thống kê (trung bình, độ lệch chuẩn, trung vị, v.v.) với và không có ngoại lai để xem mức độ ảnh hưởng.

Ví dụ: Trung bình và độ lệch chuẩn nhạy cảm với ngoại lai, trong khi trung vị và IQR ít bị ảnh hưởng.

**Chọn phương pháp xử lý**:

Nếu ngoại lai là lỗi (như nhập liệu sai): Loại bỏ hoặc thay thế.

Nếu ngoại lai có ý nghĩa thực tế: Giữ nguyên hoặc biến đổi dữ liệu.

Nếu cần so sánh: Phân tích riêng biệt hoặc sử dụng chỉ số ít nhạy cảm (trung vị, IQR).

**Kiểm tra sau xử lý**:

Vẽ lại histogram hoặc boxplot để kiểm tra phân bố sau khi xử lý.

So sánh các chỉ số thống kê trước và sau xử lý để đảm bảo không làm sai lệch dữ liệu.

**Ghi nhận phương pháp**:

Ghi lại cách xử lý ngoại lai để đảm bảo tính minh bạch và tái lập trong phân tích.

### 4. Ví dụ minh họa

Giả sử tập dữ liệu điểm số: [45, 50, 55, 60, 65, 70, 75, 80, 85, 1000] (1000 là ngoại lai có thể do lỗi nhập liệu).

**Bước 1: Xác định ngoại lai**

**Boxplot**:

Q1 = 55, Q2 = 67.5, Q3 = 80, IQR = 80 - 55 = 25.

Ranh giới: [Q1−1.5×IQR,Q3+1.5×IQR]=[55−1.5×25,80+1.5×25]=[17.5,117.5]

Giá trị 1000 > 117.5, là ngoại lai.

**Z-score**:

Trung bình (có 1000) ≈ 124.5, độ lệch chuẩn ≈ 297.5.

Z-score của 1000: Z=(1000−124.5) / 297.5 ≈ 2.94, gần ngưỡng 3, xác nhận là ngoại lai.

**Bước 2: Xử lý ngoại lai**

**Phương án 1: Loại bỏ**:

Dữ liệu mới: [45, 50, 55, 60, 65, 70, 75, 80, 85]

Trung bình: 65, Trung vị: 65, Độ lệch chuẩn: ≈12.9, Phạm vi: 85 - 45 = 40.

**Phương án 2: Thay thế bằng Q3 + 1.5 \times IQR**:

Thay 1000 bằng 117.5.

Dữ liệu mới: [45, 50, 55, 60, 65, 70, 75, 80, 85, 117.5]

Trung bình: 70.25, Trung vị: 67.5, Độ lệch chuẩn: ≈21.7.

**Phương án 3: Biến đổi log**:

Áp dụng log10​(x): [1.65, 1.70, 1.74, 1.78, 1.81, 1.85, 1.88, 1.90, 1.93, 3]

Trung bình log: ≈1.82, Độ lệch chuẩn log: ≈0.39.

**Phương án 4: Giữ nguyên**:

Trung bình: 124.5, Trung vị: 67.5, Độ lệch chuẩn: ≈297.5.

Trung bình bị sai lệch nghiêm trọng, nhưng trung vị vẫn đáng tin cậy.

**Bước 3: Diễn giải**

Nếu 1000 là lỗi nhập liệu, **loại bỏ** hoặc **thay thế** là hợp lý để đảm bảo trung bình và độ lệch chuẩn chính xác.

Nếu 1000 có ý nghĩa (ví dụ, điểm bonus), có thể **giữ nguyên** và dùng trung vị/IQR hoặc **biến đổi log** để phân tích.

### 5. Lưu ý khi xử lý ngoại lai

**Bối cảnh dữ liệu**: Luôn xem xét ý nghĩa thực tế của ngoại lai trước khi xử lý.

**Tác động đến chỉ số thống kê**:

Trung bình và độ lệch chuẩn nhạy cảm với ngoại lai.

Trung vị, Q1, Q3, và IQR ít bị ảnh hưởng, nên ưu tiên khi ngoại lai chưa được xử lý.

**Kiểm tra sau xử lý**: So sánh phân bố (histogram, boxplot) trước và sau xử lý để đảm bảo không làm mất đặc điểm tự nhiên của dữ liệu.

**Phương pháp phân tích**: Nếu giữ nguyên ngoại lai, hãy chọn các phương pháp thống kê phi tham số hoặc mô hình ít nhạy cảm với ngoại lai.

### 6. Kết luận

**Xác định ngoại lai**: Dùng boxplot, Z-score, hoặc kiểm định thống kê, kết hợp với bối cảnh dữ liệu.

**Phương pháp xử lý**:

**Loại bỏ**: Phù hợp với lỗi nhập liệu, tỷ lệ ngoại lai nhỏ.

**Thay thế**: Giữ kích thước mẫu, dùng trung vị hoặc ranh giới IQR.

**Biến đổi**: Giảm tác động của ngoại lai, đặc biệt với dữ liệu lệch mạnh.

**Giữ nguyên**: Khi ngoại lai có ý nghĩa thực tế, dùng trung vị/IQR.

**Phân tích riêng**: Khi ngoại lai đại diện cho một nhóm đặc biệt.

**Trước khi tính thống kê mô tả**: Ưu tiên xử lý ngoại lai nếu chúng làm sai lệch trung bình hoặc độ lệch chuẩn; nếu không, sử dụng các chỉ số ít nhạy cảm như trung vị và IQR.

# PHẦN 2: XỬ LÝ VÀ TRỰC QUAN HÓA DỮ LIỆU

## + Trực quan hóa dữ liệu có vai trò gì trong phân tích dữ liệu? Tại sao nó quan trọng trong khám phá dữ liệu (EDA)?

### Vai trò của trực quan hóa dữ liệu trong phân tích dữ liệu:

Trực quan hóa dữ liệu (data visualization) là quá trình sử dụng các biểu đồ, đồ thị, bản đồ hoặc các hình thức trực quan khác để biểu diễn dữ liệu một cách dễ hiểu, chuyển đổi các tập hợp dữ liệu phức tạp thành thông tin trực quan, dễ tiếp cận. Trong phân tích dữ liệu, trực quan hóa đóng vai trò cốt lõi với các chức năng sau:

* Rút ngắn thời gian hiểu dữ liệu: Dữ liệu thô thường bao gồm các bảng số liệu lớn hoặc thông tin phức tạp, khó nắm bắt chỉ bằng cách đọc. Trực quan hóa, thông qua các biểu đồ như histogram, scatter plot hay dashboard tương tác, giúp người phân tích nhanh chóng nhận diện xu hướng, mẫu hình và các đặc điểm chính của dữ liệu mà không cần xử lý thủ công hoặc tính toán phức tạp.
* Phát hiện xu hướng, ngoại lai và mối quan hệ: Trực quan hóa cho phép nhận diện các bất thường (outliers), phân bố lệch, hoặc các mối tương quan tiềm ẩn mà các phương pháp thống kê đơn thuần có thể bỏ sót. Ví dụ, một biểu đồ phân tán (scatter plot) có thể tiết lộ mối quan hệ phi tuyến giữa hai biến, trong khi heatmap correlation matrix làm nổi bật mức độ tương quan giữa nhiều biến.
* Hỗ trợ truyền đạt và ra quyết định: Trực quan hóa giúp trình bày kết quả phân tích một cách rõ ràng, dễ hiểu cho cả chuyên gia và người không chuyên. Các công cụ như Power BI, Tableau, hoặc Python (Seaborn, Plotly) tạo ra các biểu đồ trực quan, giúp các nhà quản lý, nhà khoa học hoặc nhà hoạch định chính sách đưa ra quyết định dựa trên dữ liệu một cách hiệu quả. Ví dụ, một dashboard doanh thu có thể chỉ ra ngay khu vực nào cần đầu tư thêm.
* Tăng cường hiệu quả phân tích: Trong quy trình phân tích dữ liệu, trực quan hóa hỗ trợ mọi giai đoạn, từ kiểm tra dữ liệu thô, đánh giá giả định mô hình (như hồi quy hoặc phân loại), đến trình bày kết quả cuối cùng. Nó cũng giúp phát hiện lỗi dữ liệu (missing values, duplicates) trước khi áp dụng các mô hình phức tạp.

Tóm lại, trực quan hóa dữ liệu không chỉ là công cụ trình bày mà còn là phương tiện tư duy, giúp các nhà phân tích khai thác thông tin sâu sắc từ dữ liệu, từ đó thúc đẩy các quyết định chiến lược.

### Tại sao trực quan hóa dữ liệu quan trọng trong khám phá dữ liệu (EDA)?

Khám phá dữ liệu (Exploratory Data Analysis - EDA) là bước đầu tiên trong phân tích dữ liệu, nhằm hiểu cấu trúc, chất lượng và đặc trưng của dữ liệu mà không phụ thuộc vào các giả thuyết trước. Trực quan hóa là công cụ không thể thiếu trong EDA vì những lý do sau:

* Khám phá phân bố và mẫu hình: EDA tập trung vào việc hiểu cách dữ liệu được phân bố, các mối quan hệ giữa các biến và các mẫu tiềm ẩn. Các biểu đồ như histogram (phân bố), scatter plot (tương quan), hoặc boxplot (ngoại lai) giúp nhà phân tích nhanh chóng hình dung các đặc điểm này. Ví dụ, một histogram có thể cho thấy dữ liệu có phân bố chuẩn hay lệch, trong khi một pair plot tiết lộ các mối quan hệ đa biến.
* Phát hiện vấn đề dữ liệu: Trong EDA, trực quan hóa giúp nhận diện các vấn đề như giá trị thiếu (missing values), giá trị ngoại lai, hoặc dữ liệu không nhất quán. Ví dụ, một biểu đồ cột (bar chart) có thể hiển thị tỷ lệ missing values theo biến, trong khi boxplot giúp phát hiện các điểm dữ liệu bất thường.
* Tạo giả thuyết và định hướng phân tích: Trực quan hóa kích thích tư duy khám phá, giúp nhà phân tích đặt ra các câu hỏi mới hoặc giả thuyết để kiểm tra. Chẳng hạn, một biểu đồ phân tán cho thấy mối quan hệ tuyến tính giữa hai biến có thể dẫn đến việc áp dụng mô hình hồi quy trong các bước phân tích tiếp theo.
* Xử lý dữ liệu lớn và phức tạp: Với các tập dữ liệu lớn, việc phân tích thủ công hoặc chỉ dựa vào thống kê mô tả (như mean, variance) là không đủ. Trực quan hóa tóm tắt hàng triệu điểm dữ liệu thành các hình ảnh dễ hiểu, giúp tập trung vào các khía cạnh quan trọng mà không bị choáng ngợp bởi khối lượng thông tin.
* Tăng tính tương tác trong EDA: Các công cụ trực quan hóa hiện đại (như Plotly, Tableau) cho phép tương tác với dữ liệu, chẳng hạn như phóng to, lọc, hoặc thay đổi góc nhìn, giúp nhà phân tích khám phá dữ liệu một cách linh hoạt và sâu sắc hơn.

Nếu không có trực quan hóa, EDA sẽ bị giới hạn ở các phép tính thống kê khô khan, làm giảm khả năng phát hiện các insight quan trọng. Theo triết lý của John Tukey, người tiên phong trong EDA, trực quan hóa giúp "dữ liệu kể câu chuyện của chính nó", làm sáng tỏ các đặc điểm mà các con số thô không thể truyền tải. Trong thực tiễn, các thư viện như Matplotlib, Seaborn (Python) hoặc ggplot2 (R) đã trở thành công cụ không thể thiếu, giúp EDA trở nên hiệu quả, trực quan và mang lại giá trị tối đa cho phân tích dữ liệu.

## + Các loại biểu đồ phổ biến (như histogram, scatter plot, boxplot, bar chart) được sử dụng trong các trường hợp nào?

### 1. Histogram

* **Mô tả**: Histogram là biểu đồ cột liền kề, thể hiện phân bố tần suất của một biến định lượng liên tục bằng cách chia dữ liệu thành các khoảng (bins) và đếm số lần xuất hiện trong mỗi khoảng. Nó cung cấp cái nhìn trực quan về hình dạng phân bố dữ liệu (như chuẩn, lệch trái/phải, hoặc đa đỉnh).
* **Trường hợp sử dụng**:
  + **Hiểu phân bố dữ liệu**: Sử dụng để kiểm tra xem một biến có phân bố chuẩn, lệch (skewed), hay đa đỉnh (bimodal/multimodal) không. Ví dụ: phân tích phân bố thu nhập của một nhóm dân cư hoặc thời gian phản hồi của một hệ thống.
  + **Phát hiện đặc điểm bất thường**: Histogram giúp nhận diện các khoảng dữ liệu bất thường, như sự tập trung quá mức ở một vài giá trị hoặc sự hiện diện của giá trị ngoại lai (outliers).
  + **So sánh phân bố giữa các nhóm**: Có thể sử dụng nhiều histogram (hoặc histogram chồng lên nhau) để so sánh phân bố của một biến giữa các nhóm, ví dụ: phân bố điểm thi của học sinh theo giới tính.
  + **Hỗ trợ kiểm tra giả định thống kê**: Trong EDA, histogram giúp đánh giá giả định về phân bố dữ liệu (như tính chuẩn) trước khi áp dụng các mô hình thống kê hoặc machine learning.
  + **Ví dụ**: Histogram của thời gian giao hàng trong một công ty logistics để xem liệu quá trình này có bị trễ thường xuyên hay không.
* **Khi nào sử dụng**: Khi cần phân tích phân bố của một biến định lượng liên tục, đặc biệt trong EDA để hiểu cấu trúc dữ liệu.

### 2. Scatter Plot (Biểu đồ phân tán)

* **Mô tả**: Scatter plot hiển thị các điểm dữ liệu trên hệ tọa độ 2D, với một biến định lượng trên trục x và một biến định lượng khác trên trục y. Mỗi điểm đại diện cho một cặp giá trị của hai biến.
* **Trường hợp sử dụng**:
  + **Khám phá mối quan hệ giữa hai biến**: Dùng để kiểm tra xem hai biến có tương quan tuyến tính, phi tuyến, hay không có tương quan. Ví dụ: mối quan hệ giữa số giờ học và điểm thi của học sinh.
  + **Phát hiện cụm (clusters) và ngoại lai**: Scatter plot giúp nhận diện các nhóm dữ liệu tự nhiên hoặc các điểm bất thường nằm xa phần lớn dữ liệu. Ví dụ: phát hiện các giao dịch bất thường trong dữ liệu tài chính.
  + **Hiểu xu hướng và mẫu hình**: Dùng để quan sát xu hướng dữ liệu theo thời gian hoặc giữa các biến, như doanh thu hàng tháng theo chi phí quảng cáo.
  + **Hỗ trợ xây dựng mô hình**: Trong EDA, scatter plot giúp xác định các mối quan hệ tiềm năng để áp dụng các mô hình như hồi quy tuyến tính hoặc phi tuyến.
  + **Ví dụ**: Scatter plot giữa chi tiêu quảng cáo và doanh thu bán hàng để đánh giá hiệu quả chiến dịch marketing.
* **Khi nào sử dụng**: Khi cần khám phá mối quan hệ hoặc tương quan giữa hai biến định lượng, hoặc để phát hiện các mẫu và ngoại lai.

### 3. Boxplot (Biểu đồ hộp)

* **Mô tả**: Boxplot (hay biểu đồ hộp và râu) tóm tắt phân bố của một biến định lượng thông qua các thành phần: trung vị (median), tứ phân vị (Q1, Q3), khoảng tứ phân vị (IQR), và các râu (whiskers) biểu thị phạm vi dữ liệu. Các điểm ngoài râu được coi là giá trị ngoại lai.
* **Trường hợp sử dụng**:
  + **Tóm tắt phân bố dữ liệu**: Boxplot cung cấp cái nhìn tổng quan về trung tâm (median), độ phân tán (IQR), và tính đối xứng của dữ liệu mà không cần hiển thị toàn bộ phân bố.
  + **Phát hiện giá trị ngoại lai**: Boxplot đặc biệt hữu ích trong EDA để xác định các điểm dữ liệu bất thường (outliers), như giá trị cực đại hoặc cực tiểu bất thường. Ví dụ: phát hiện các giao dịch có giá trị quá cao trong dữ liệu bán hàng.
  + **So sánh phân bố giữa các nhóm**: Dùng để so sánh phân bố của một biến định lượng giữa các nhóm định tính, như lương của nhân viên theo phòng ban hoặc thời gian hoàn thành công việc theo ca làm việc.
  + **Hỗ trợ kiểm tra giả định mô hình**: Boxplot giúp kiểm tra tính đồng nhất phương sai (homoscedasticity) hoặc các đặc điểm dữ liệu trước khi áp dụng các mô hình thống kê.
  + **Ví dụ**: Boxplot của giá nhà theo khu vực để so sánh mức giá trung bình và phát hiện các căn nhà có giá bất thường.
* **Khi nào sử dụng**: Khi cần tóm tắt phân bố, phát hiện ngoại lai, hoặc so sánh phân bố giữa các nhóm trong một biến định lượng.

### 4. Bar Chart (Biểu đồ cột)

* **Mô tả**: Bar chart sử dụng các cột có chiều cao tỷ lệ với giá trị của biến, thường dùng cho dữ liệu định tính (categorical) hoặc dữ liệu định lượng rời rạc. Các cột không liền kề nhau, khác với histogram.
* **Trường hợp sử dụng**:
  + **So sánh giá trị giữa các danh mục**: Bar chart lý tưởng để so sánh giá trị của một biến giữa các nhóm, như doanh thu của các cửa hàng khác nhau hoặc số lượng khách hàng theo khu vực địa lý.
  + **Tóm tắt tần suất hoặc giá trị tổng hợp**: Dùng để hiển thị tần suất (frequency) hoặc giá trị tổng của các danh mục, như số lượng sản phẩm bán ra theo loại sản phẩm.
  + **Hiển thị xu hướng đơn giản**: Khi cần thể hiện sự thay đổi của một biến định tính qua thời gian, như doanh số bán hàng theo tháng.
  + **Truyền đạt thông tin rõ ràng**: Bar chart dễ hiểu, phù hợp để trình bày kết quả cho đối tượng không chuyên, như trong báo cáo kinh doanh hoặc thuyết trình.
  + **Ví dụ**: Bar chart hiển thị số lượng xe bán ra của các hãng xe (Toyota, Honda, Ford) trong một quý.
* **Khi nào sử dụng**: Khi cần so sánh giá trị hoặc tần suất giữa các danh mục định tính hoặc rời rạc, đặc biệt trong các báo cáo hoặc phân tích tổng hợp.

### Hướng dẫn chọn loại biểu đồ

* **Histogram**: Dùng cho phân tích phân bố của một biến định lượng liên tục, tập trung vào hình dạng và đặc điểm phân bố.
* **Scatter Plot**: Dùng để khám phá mối quan hệ giữa hai biến định lượng, xác định tương quan, cụm, hoặc ngoại lai.
* **Boxplot**: Dùng để tóm tắt phân bố, phát hiện ngoại lai, hoặc so sánh phân bố giữa các nhóm.
* **Bar Chart**: Dùng để so sánh giá trị giữa các danh mục định tính hoặc rời rạc, hoặc để trình bày tóm tắt dữ liệu.

**Lưu ý khi sử dụng**

* **Phù hợp với loại dữ liệu**: Đảm bảo chọn biểu đồ phù hợp với loại dữ liệu (định lượng liên tục, định lượng rời rạc, hoặc định tính) để tránh diễn giải sai.
* **Mục tiêu phân tích**: Trong EDA, chọn biểu đồ dựa trên câu hỏi cụ thể, ví dụ: tìm hiểu phân bố (histogram), tương quan (scatter plot), ngoại lai (boxplot), hay so sánh nhóm (bar chart).
* **Tối ưu hóa trình bày**: Sử dụng màu sắc, nhãn và tỷ lệ hợp lý để đảm bảo biểu đồ rõ ràng, không gây nhầm lẫn. Tránh lạm dụng các hiệu ứng 3D hoặc màu sắc phức tạp làm mất tập trung.
* **Công cụ hỗ trợ**: Các thư viện như Matplotlib, Seaborn (Python), ggplot2 (R), hoặc phần mềm như Tableau, Power BI giúp tạo ra các biểu đồ này một cách dễ dàng và chuyên nghiệp.

## + Làm thế nào để chọn loại biểu đồ phù hợp với đặc điểm của dữ liệu (ví dụ: dữ liệu phân loại, dữ liệu số, dữ liệu thời gian)?

Việc chọn loại biểu đồ phù hợp với đặc điểm của dữ liệu là một bước quan trọng trong phân tích và trực quan hóa dữ liệu, đặc biệt trong khám phá dữ liệu (EDA). Sự lựa chọn này phụ thuộc vào **loại dữ liệu** (phân loại, số, thời gian), **mục tiêu phân tích** (so sánh, phân bố, xu hướng, tương quan), và **đối tượng người xem**.

### 1. Hiểu các loại dữ liệu và đặc điểm của chúng

Trước khi chọn biểu đồ, cần xác định rõ loại dữ liệu bạn đang xử lý:

* **Dữ liệu phân loại (Categorical Data)**: Bao gồm các danh mục hoặc nhóm không có thứ tự số học, ví dụ: giới tính (nam, nữ), khu vực (miền Bắc, miền Nam), loại sản phẩm (điện thoại, laptop). Có thể chia thành:
  + **Nominal**: Không có thứ tự (ví dụ: màu sắc, thương hiệu).
  + **Ordinal**: Có thứ tự (ví dụ: trình độ học vấn - trung học, đại học, sau đại học).
* **Dữ liệu số (Numerical Data)**: Là các giá trị số, chia thành:
  + **Liên tục (Continuous)**: Giá trị có thể là bất kỳ số thực nào trong một khoảng, ví dụ: chiều cao, nhiệt độ, doanh thu.
  + **Rời rạc (Discrete)**: Giá trị chỉ nhận các số nguyên hoặc giá trị cụ thể, ví dụ: số lượng sản phẩm bán ra, số người tham gia.
* **Dữ liệu thời gian (Time-Series Data)**: Dữ liệu được thu thập theo thời gian, thường liên quan đến xu hướng hoặc chu kỳ, ví dụ: doanh thu hàng tháng, giá cổ phiếu theo ngày.

### 2. Hướng dẫn chọn biểu đồ theo loại dữ liệu và mục tiêu phân tích

Dưới đây là cách chọn loại biểu đồ phù hợp dựa trên đặc điểm dữ liệu và mục tiêu phân tích, kèm theo các ví dụ cụ thể.

**A. Dữ liệu phân loại (Categorical Data)**

Dữ liệu phân loại thường được sử dụng để so sánh tần suất, tỷ lệ, hoặc giá trị tổng hợp giữa các danh mục.

* **Biểu đồ phù hợp**:
  + **Bar Chart (Biểu đồ cột)**:
    - **Khi nào sử dụng**: So sánh giá trị giữa các danh mục riêng biệt (nominal) hoặc có thứ tự (ordinal). Thích hợp để hiển thị tần suất, tổng hợp, hoặc giá trị trung bình của một biến số theo danh mục.
    - **Ví dụ**: So sánh doanh thu của các cửa hàng (Hà Nội, TP.HCM, Đà Nẵng) hoặc số lượng khách hàng theo loại thành viên (thường, VIP).
    - **Lưu ý**: Sử dụng cột dọc hoặc ngang tùy thuộc vào số lượng danh mục. Nếu có quá nhiều danh mục (>10), hãy cân nhắc nhóm lại hoặc dùng biểu đồ khác như treemap.
  + **Pie Chart (Biểu đồ tròn)**:
    - **Khi nào sử dụng**: Hiển thị tỷ lệ phần trăm hoặc đóng góp của từng danh mục trong tổng thể, đặc biệt khi có ít danh mục (thường <6).
    - **Ví dụ**: Tỷ lệ thị phần của các thương hiệu điện thoại (Apple, Samsung, Xiaomi).
    - **Lưu ý**: Tránh sử dụng nếu có quá nhiều danh mục hoặc tỷ lệ giữa các danh mục quá nhỏ, vì sẽ khó đọc.
  + **Stacked Bar Chart (Biểu đồ cột chồng)**:
    - **Khi nào sử dụng**: So sánh tổng giá trị giữa các danh mục và đồng thời hiển thị cấu trúc bên trong của mỗi danh mục.
    - **Ví dụ**: Doanh thu của các cửa hàng theo loại sản phẩm (điện thoại, phụ kiện) trong từng khu vực.
    - **Lưu ý**: Đảm bảo các danh mục con dễ phân biệt bằng màu sắc hoặc nhãn.
  + **Treemap**:
    - **Khi nào sử dụng**: Khi có nhiều danh mục hoặc danh mục phân cấp (hierarchical), cần hiển thị tỷ lệ đóng góp.
    - **Ví dụ**: Tỷ lệ doanh thu của các loại sản phẩm trong một danh mục lớn (điện tử, thời trang, đồ gia dụng).
* **Mục tiêu phân tích**:
  + So sánh giá trị: Bar chart là lựa chọn hàng đầu.
  + Hiển thị tỷ lệ: Pie chart hoặc treemap.
  + Kết hợp so sánh và cấu trúc: Stacked bar chart.

**B. Dữ liệu số (Numerical Data)**

Dữ liệu số thường được sử dụng để phân tích phân bố, mối quan hệ, hoặc sự phân tán của dữ liệu.

* **Biểu đồ phù hợp**:
  + **Histogram**:
    - **Khi nào sử dụng**: Phân tích phân bố của một biến định lượng liên tục, để xem dữ liệu có chuẩn, lệch, hay đa đỉnh không.
    - **Ví dụ**: Phân bố tuổi của khách hàng hoặc thời gian hoàn thành đơn hàng.
    - **Lưu ý**: Chọn số lượng bins hợp lý (quá ít bins làm mất chi tiết, quá nhiều bins gây nhiễu).
  + **Boxplot (Biểu đồ hộp)**:
    - **Khi nào sử dụng**: Tóm tắt phân bố của một biến số (trung vị, tứ phân vị, phạm vi) và phát hiện giá trị ngoại lai. Cũng dùng để so sánh phân bố giữa các nhóm danh mục.
    - **Ví dụ**: So sánh giá nhà giữa các khu vực hoặc phân bố điểm thi theo lớp học.
    - **Lưu ý**: Hữu ích khi cần phát hiện outliers hoặc so sánh độ phân tán.
  + **Scatter Plot (Biểu đồ phân tán)**:
    - **Khi nào sử dụng**: Khám phá mối quan hệ giữa hai biến định lượng, như tương quan tuyến tính, phi tuyến, hoặc phát hiện cụm/ngoại lai.
    - **Ví dụ**: Mối quan hệ giữa chi tiêu quảng cáo và doanh thu hoặc chiều cao và cân nặng.
    - **Lưu ý**: Có thể thêm màu sắc hoặc kích thước điểm để biểu thị biến thứ ba (ví dụ: nhóm khách hàng).
  + **Line Plot (Biểu đồ đường)**:
    - **Khi nào sử dụng**: Khi dữ liệu số có yếu tố thời gian hoặc thứ tự liên tục, nhưng không phải dữ liệu thời gian thực sự (xem phần dữ liệu thời gian).
    - **Ví dụ**: Sự thay đổi của một biến số theo các giá trị liên tục, như hiệu suất mô hình theo tham số điều chỉnh.
    - **Lưu ý**: Chỉ dùng khi trục x là liên tục hoặc có thứ tự rõ ràng.
* **Mục tiêu phân tích**:
  + Phân bố: Histogram hoặc boxplot.
  + Tương quan/mối quan hệ: Scatter plot.
  + So sánh giữa các nhóm: Boxplot hoặc scatter plot với màu sắc phân nhóm.

**C. Dữ liệu thời gian (Time-Series Data)**

Dữ liệu thời gian thường được sử dụng để phân tích xu hướng, chu kỳ, hoặc biến động theo thời gian.

* **Biểu đồ phù hợp**:
  + **Line Chart (Biểu đồ đường)**:
    - **Khi nào sử dụng**: Hiển thị xu hướng hoặc sự thay đổi của một biến số theo thời gian, như doanh thu hàng tháng, nhiệt độ hàng ngày.
    - **Ví dụ**: Biểu đồ giá cổ phiếu theo ngày hoặc số lượt truy cập website theo tuần.
    - **Lưu ý**: Đảm bảo trục thời gian được chia tỷ lệ đều (linear scale) trừ khi yêu cầu log scale.
  + **Area Chart**:
    - **Khi nào sử dụng**: Tương tự line chart, nhưng nhấn mạnh vào khối lượng hoặc tổng tích lũy, thường dùng khi có nhiều chuỗi dữ liệu thời gian.
    - **Ví dụ**: Doanh thu tích lũy của các sản phẩm theo thời gian.
    - **Lưu ý**: Tránh chồng quá nhiều khu vực, gây khó đọc.
  + **Stacked Area Chart**:
    - **Khi nào sử dụng**: Hiển thị đóng góp của từng danh mục vào tổng thể theo thời gian.
    - **Ví dụ**: Doanh thu theo loại sản phẩm (điện thoại, laptop) qua các quý.
    - **Lưu ý**: Chỉ dùng khi các danh mục có ý nghĩa khi cộng dồn.
  + **Bar Chart (cho khoảng thời gian rời rạc)**:
    - **Khi nào sử dụng**: Khi thời gian được nhóm thành các khoảng rời rạc (như tháng, quý), và cần so sánh giá trị.
    - **Ví dụ**: Doanh thu hàng quý của một công ty.
    - **Lưu ý**: Thích hợp hơn line chart khi các điểm thời gian không liên tục.
* **Mục tiêu phân tích**:
  + Xu hướng hoặc chu kỳ: Line chart hoặc area chart.
  + So sánh giá trị theo thời gian rời rạc: Bar chart.
  + Cấu trúc đóng góp theo thời gian: Stacked area chart.

### 3. Quy tắc chung khi chọn biểu đồ

Để chọn biểu đồ phù hợp, cần cân nhắc các yếu tố sau:

* **Loại dữ liệu**:
  + **Phân loại (Categorical)**: Bar chart, pie chart, treemap, stacked bar chart.
  + **Số liên tục (Continuous Numerical)**: Histogram, boxplot, scatter plot.
  + **Số rời rạc (Discrete Numerical)**: Bar chart, boxplot.
  + **Thời gian (Time-Series)**: Line chart, area chart, stacked area chart.
* **Mục tiêu phân tích**:
  + **Phân bố**: Histogram, boxplot.
  + **So sánh**: Bar chart, boxplot, stacked bar chart.
  + **Tương quan**: Scatter plot.
  + **Xu hướng**: Line chart, area chart.
  + **Tỷ lệ/đóng góp**: Pie chart, treemap, stacked bar/area chart.
* **Số lượng biến**:
  + Một biến: Histogram, boxplot (cho số); bar chart, pie chart (cho phân loại).
  + Hai biến: Scatter plot (cho số); bar chart hoặc boxplot (cho số và phân loại).
  + Nhiều biến: Heatmap, pair plot, hoặc stacked chart.
* **Đối tượng người xem**:
  + Đối tượng không chuyên: Ưu tiên biểu đồ đơn giản như bar chart, pie chart.
  + Đối tượng chuyên môn: Có thể dùng các biểu đồ phức tạp hơn như boxplot, scatter plot, hoặc heatmap.
* **Kích thước dữ liệu**:
  + Dữ liệu nhỏ: Pie chart, bar chart.
  + Dữ liệu lớn hoặc nhiều danh mục: Treemap, heatmap, hoặc lọc dữ liệu trước khi trực quan hóa.

### 4. Lưu ý khi chọn và thiết kế biểu đồ

* **Đơn giản và rõ ràng**: Tránh lạm dụng màu sắc, hiệu ứng 3D, hoặc quá nhiều thông tin trong một biểu đồ, vì có thể gây nhầm lẫn.
* **Phù hợp với công cụ**: Sử dụng các thư viện như Matplotlib, Seaborn (Python), ggplot2 (R), hoặc phần mềm như Tableau, Power BI để tạo biểu đồ chất lượng cao.
* **Kiểm tra thang đo**: Đảm bảo trục và tỷ lệ phù hợp, đặc biệt với dữ liệu thời gian hoặc dữ liệu số lớn (cân nhắc log scale nếu cần).
* **Thử nghiệm nhiều biểu đồ**: Trong EDA, thử nhiều loại biểu đồ để khám phá dữ liệu từ các góc độ khác nhau.

### 5. Ví dụ minh họa

Giả sử bạn có dữ liệu về doanh thu của một công ty theo khu vực và thời gian:

* **Dữ liệu phân loại**: Doanh thu theo khu vực (Hà Nội, TP.HCM, Đà Nẵng).
  + **Biểu đồ phù hợp**: Bar chart để so sánh doanh thu giữa các khu vực.
* **Dữ liệu số**: Phân bố doanh thu hàng ngày của một khu vực.
  + **Biểu đồ phù hợp**: Histogram để xem phân bố, hoặc boxplot để phát hiện ngoại lai.
* **Dữ liệu thời gian**: Doanh thu hàng tháng trong năm 2024.
  + **Biểu đồ phù hợp**: Line chart để hiển thị xu hướng.
* **Kết hợp số và phân loại**: Doanh thu theo khu vực và loại sản phẩm.
  + **Biểu đồ phù hợp**: Stacked bar chart để so sánh tổng doanh thu và tỷ lệ đóng góp của từng loại sản phẩm.

## + Sự khác biệt giữa các thư viện trực quan hóa trong Python như Matplotlib, Seaborn và Plotly là gì?

Các thư viện trực quan hóa dữ liệu trong Python như **Matplotlib**, **Seaborn**, và **Plotly** đều là những công cụ mạnh mẽ, nhưng chúng có mục đích, tính năng, và trường hợp sử dụng khác nhau.

### 1. Matplotlib

* **Mô tả**: Matplotlib là thư viện trực quan hóa lâu đời và phổ biến nhất trong Python, được phát triển để tạo ra các biểu đồ 2D (và một số 3D) với khả năng tùy chỉnh cao. Nó là nền tảng cho nhiều thư viện khác như Seaborn.
* **Đặc điểm chính**:
  + **Tính linh hoạt**: Hỗ trợ nhiều loại biểu đồ như line plot, scatter plot, bar chart, histogram, boxplot, pie chart, v.v.
  + **Tùy chỉnh chi tiết**: Cho phép kiểm soát gần như mọi khía cạnh của biểu đồ (màu sắc, nhãn, trục, kích thước, v.v.), phù hợp cho các nhà phân tích muốn tinh chỉnh biểu đồ.
  + **Hỗ trợ cơ bản**: Là thư viện cấp thấp, cung cấp nền tảng để xây dựng biểu đồ từ đầu, nhưng thường yêu cầu nhiều dòng code hơn để tạo biểu đồ phức tạp.
  + **Không tương tác**: Biểu đồ Matplotlib thường là tĩnh, phù hợp với các ấn phẩm hoặc báo cáo in ấn.
* **Ưu điểm**:
  + Cộng đồng lớn, tài liệu phong phú, và tích hợp tốt với các thư viện như NumPy, Pandas.
  + Có thể tạo ra các biểu đồ chất lượng cao, phù hợp cho xuất bản khoa học (ví dụ: bài báo học thuật).
  + Hỗ trợ nhiều định dạng xuất file (PNG, PDF, SVG, v.v.).
* **Hạn chế**:
  + Giao diện mặc định không bắt mắt, cần tùy chỉnh nhiều để tạo biểu đồ thẩm mỹ.
  + Yêu cầu nhiều code để tạo biểu đồ phức tạp hoặc đẹp mắt.
  + Không hỗ trợ tương tác trực tiếp (trừ khi kết hợp với các công cụ khác như IPython/Jupyter).
* **Trường hợp sử dụng**:
  + Tạo biểu đồ tĩnh cho báo cáo, bài báo, hoặc tài liệu khoa học.
  + Khi cần tùy chỉnh chi tiết hoặc xây dựng các biểu đồ không chuẩn.
  + Ví dụ: Vẽ histogram phân bố dữ liệu hoặc biểu đồ line để phân tích xu hướng đơn giản.

Ví dụ code:

import matplotlib.pyplot as plt

import numpy as np

x = np.random.randn(1000)

plt.hist(x, bins=30, color='blue', alpha=0.7)

plt.title('Histogram of Random Data')

plt.xlabel('Value')

plt.ylabel('Frequency')

plt.show()

### 2. Seaborn

* **Mô tả**: Seaborn được xây dựng dựa trên Matplotlib, cung cấp giao diện cấp cao hơn để tạo ra các biểu đồ thẩm mỹ và thống kê với ít code hơn. Nó tập trung vào việc đơn giản hóa việc trực quan hóa dữ liệu thống kê, đặc biệt trong EDA.
* **Đặc điểm chính**:
  + **Tập trung vào thống kê**: Tích hợp sẵn các biểu đồ thống kê như heatmap, boxplot, violin plot, pair plot, và regression plot.
  + **Thẩm mỹ mặc định**: Seaborn cung cấp các chủ đề (themes) và bảng màu đẹp mắt, giúp tạo ra biểu đồ chuyên nghiệp mà không cần tùy chỉnh nhiều.
  + **Tích hợp với Pandas**: Hoạt động tốt với DataFrame của Pandas, giúp dễ dàng xử lý dữ liệu lớn.
  + **Hỗ trợ tĩnh**: Giống Matplotlib, Seaborn chủ yếu tạo ra các biểu đồ tĩnh, nhưng dễ sử dụng hơn cho các tác vụ EDA.
* **Ưu điểm**:
  + Code ngắn gọn và dễ sử dụng hơn Matplotlib cho các biểu đồ thống kê phổ biến.
  + Tích hợp sẵn các kiểu biểu đồ phức tạp như heatmap, pair plot, hoặc violin plot, rất hữu ích trong EDA.
  + Giao diện mặc định đẹp, dễ tùy chỉnh chủ đề và màu sắc.
* **Hạn chế**:
  + Ít linh hoạt hơn Matplotlib khi cần tạo biểu đồ không chuẩn hoặc tùy chỉnh sâu.
  + Vẫn là biểu đồ tĩnh, không hỗ trợ tương tác.
  + Phụ thuộc vào Matplotlib, nên đôi khi cần hiểu Matplotlib để tinh chỉnh thêm.
* **Trường hợp sử dụng**:
  + Phân tích dữ liệu trong EDA, đặc biệt khi cần khám phá phân bố, tương quan, hoặc so sánh giữa các nhóm.
  + Tạo các biểu đồ thống kê phức tạp như heatmap cho ma trận tương quan hoặc pair plot để phân tích đa biến.
  + Ví dụ: Vẽ boxplot so sánh phân bố dữ liệu hoặc heatmap để kiểm tra tương quan.

Ví dụ code:

import seaborn as sns

import pandas as pd

# Giả sử có DataFrame

df = pd.DataFrame({'A': np.random.randn(100), 'B': np.random.rand(100)})

sns.scatterplot(x='A', y='B', data=df)

plt.title('Scatter Plot with Seaborn')

plt.show()

### 3. Plotly

* **Mô tả**: Plotly là thư viện hiện đại, tập trung vào việc tạo các biểu đồ tương tác (interactive) có thể hiển thị trên web hoặc trong Jupyter Notebook. Nó hỗ trợ cả biểu đồ 2D, 3D và các ứng dụng tương tác phức tạp.
* **Đặc điểm chính**:
  + **Tương tác**: Biểu đồ Plotly cho phép người dùng tương tác (phóng to, di chuột để xem giá trị, lọc dữ liệu, v.v.), phù hợp cho các ứng dụng web hoặc dashboard.
  + **Hỗ trợ đa dạng**: Tạo được nhiều loại biểu đồ, từ cơ bản (line, bar, scatter) đến nâng cao (3D plot, bản đồ, sankey diagram).
  + **Tích hợp web**: Có thể xuất biểu đồ sang HTML hoặc tích hợp với các framework như Dash để xây dựng ứng dụng web.
  + **API cấp cao và cấp thấp**: Plotly Express cung cấp cách tiếp cận đơn giản, trong khi Plotly Graph Objects cho phép tùy chỉnh chi tiết.
* **Ưu điểm**:
  + Biểu đồ tương tác, lý tưởng cho việc trình bày trực tuyến hoặc chia sẻ với người dùng không chuyên.
  + Hỗ trợ biểu đồ 3D, bản đồ địa lý, và các biểu đồ phức tạp mà Matplotlib/Seaborn khó thực hiện.
  + Tích hợp tốt với Pandas và các công cụ web như Jupyter, Dash.
* **Hạn chế**:
  + Hiệu suất có thể chậm hơn với dữ liệu rất lớn so với Matplotlib/Seaborn.
  + Yêu cầu kết nối internet cho một số tính năng (trong chế độ online), mặc dù có chế độ offline.
  + Độ phức tạp cao hơn khi tùy chỉnh sâu so với Matplotlib.
* **Trường hợp sử dụng**:
  + Tạo dashboard tương tác hoặc báo cáo trực tuyến.
  + Trực quan hóa dữ liệu phức tạp như bản đồ địa lý, biểu đồ 3D, hoặc dữ liệu thời gian thực.
  + Ví dụ: Vẽ biểu đồ line tương tác cho xu hướng doanh thu hoặc bản đồ hiển thị doanh số theo khu vực.

import plotly.express as px

import pandas as pd

df = pd.DataFrame({'x': range(10), 'y': np.random.randn(10)})

fig = px.line(df, x='x', y='y', title='Interactive Line Plot')

fig.show()

### 4. So sánh tổng quan

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | Matplotlib | Seaborn | Plotly |
| Loại biểu đồ | Tĩnh, 2D (hỗ trợ 3D cơ bản) | Tĩnh, tập trung thống kê | Tương tác, 2D/3D, bản đồ |
| Tính thẩm mỹ | Mặc định đơn giản, cần tùy chỉnh | Đẹp, chủ đề mặc định bắt mắt | Đẹp, hiện đại, tương tác |
| Dễ sử dụng | Cần nhiều code, tùy chỉnh phức tạp | Code ngắn gọn, dễ dùng cho EDA | Dễ với Plotly Express, phức tạp với Graph Objects |
| Tùy chỉnh | Rất linh hoạt, cấp thấp | Trung bình, dựa trên Matplotlib | Linh hoạt, nhưng phức tạp hơn |
| Tương tác | Không (trừ khi dùng công cụ bổ sung) | Không | Có, lý tưởng cho web/dashboard |
| Hiệu suất | Tốt với dữ liệu lớn | Tốt, nhưng phụ thuộc Matplotlib | Chậm hơn với dữ liệu rất lớn |
| Trường hợp sử dụng | Báo cáo tĩnh, xuất bản khoa học | EDA, biểu đồ thống kê | Dashboard, web, biểu đồ phức tạp |

### 5. Khi nào chọn thư viện nào?

* **Matplotlib**:
  + Khi bạn cần tạo biểu đồ tĩnh với tùy chỉnh chi tiết, như cho bài báo khoa học hoặc báo cáo in ấn.
  + Phù hợp khi làm việc với các biểu đồ cơ bản hoặc dữ liệu lớn mà không cần tương tác.
  + Ví dụ: Tạo histogram phân bố hoặc line plot cho phân tích dữ liệu cơ bản.
* **Seaborn**:
  + Khi bạn thực hiện EDA và cần nhanh chóng tạo các biểu đồ thống kê đẹp mắt (heatmap, boxplot, pair plot).
  + Lý tưởng cho việc khám phá dữ liệu với Pandas DataFrame.
  + Ví dụ: Vẽ heatmap để kiểm tra tương quan giữa các biến hoặc violin plot để so sánh phân bố.
* **Plotly**:
  + Khi bạn cần biểu đồ tương tác để trình bày trên web, dashboard, hoặc chia sẻ với người dùng không chuyên.
  + Phù hợp cho các biểu đồ phức tạp như 3D, bản đồ địa lý, hoặc dữ liệu thời gian thực.
  + Ví dụ: Tạo dashboard doanh thu tương tác hoặc biểu đồ 3D cho dữ liệu khoa học.

### 6. Kết hợp các thư viện

Trong thực tế, các thư viện này thường được sử dụng cùng nhau:

* **Matplotlib + Seaborn**: Seaborn sử dụng Matplotlib làm backend, nên bạn có thể dùng Matplotlib để tinh chỉnh thêm các biểu đồ Seaborn.
* **Matplotlib/Seaborn + Plotly**: Dùng Matplotlib/Seaborn để khám phá dữ liệu ban đầu trong EDA, sau đó chuyển sang Plotly để tạo biểu đồ tương tác cho trình bày cuối cùng.
* **Plotly Express + Graph Objects**: Sử dụng Plotly Express cho các biểu đồ nhanh, sau đó chuyển sang Graph Objects nếu cần tùy chỉnh chi tiết.

### 7. Ví dụ minh họa

Giả sử bạn có dữ liệu về doanh thu theo khu vực và muốn trực quan hóa:

* **Matplotlib**: Vẽ bar chart tĩnh để so sánh doanh thu giữa các khu vực, phù hợp cho báo cáo in.
* **Seaborn**: Vẽ boxplot để so sánh phân bố doanh thu theo khu vực, nhấn mạnh vào ngoại lai.
* **Plotly**: Vẽ biểu đồ line tương tác để hiển thị xu hướng doanh thu theo thời gian trên web.

## + Những nguyên tắc thiết kế nào cần tuân thủ để tạo ra một biểu đồ trực quan hóa dễ hiểu và hiệu quả?

Để tạo ra một biểu đồ trực quan hóa dễ hiểu và hiệu quả, cần tuân thủ các **nguyên tắc thiết kế** giúp tối ưu hóa khả năng truyền tải thông tin, giảm nhiễu, và đảm bảo người xem (bao gồm cả người không chuyên) có thể dễ dàng hiểu được câu chuyện mà dữ liệu muốn kể. Dưới đây là phân tích chuyên sâu về các nguyên tắc thiết kế quan trọng, được trình bày một cách có hệ thống và áp dụng trong bối cảnh phân tích dữ liệu, đặc biệt là khám phá dữ liệu (EDA).

### 1. Tối giản và rõ ràng (Simplicity and Clarity)

* **Nguyên tắc**: Loại bỏ các yếu tố không cần thiết để tập trung vào thông tin cốt lõi. Biểu đồ nên đơn giản, tránh gây nhầm lẫn hoặc quá tải thông tin.
* **Cách thực hiện**:
  + Chỉ bao gồm các yếu tố trực tiếp liên quan đến mục tiêu phân tích (ví dụ: nhãn trục, tiêu đề, dữ liệu chính).
  + Tránh sử dụng hiệu ứng 3D, màu sắc sặc sỡ không cần thiết, hoặc các chi tiết trang trí (như nền phức tạp, đường lưới dày).
  + Sử dụng font chữ dễ đọc (như Arial, Helvetica) và kích thước chữ đủ lớn (thường từ 10pt trở lên cho nhãn, 12-14pt cho tiêu đề).
* **Ví dụ**: Khi vẽ bar chart so sánh doanh thu giữa các khu vực, không thêm các hiệu ứng đổ bóng hoặc màu gradient phức tạp; chỉ dùng màu đơn giản và nhãn rõ ràng.
* **Tại sao quan trọng**: Tối giản giúp người xem tập trung vào dữ liệu thay vì bị phân tâm bởi các yếu tố không liên quan.

### 2. Chọn loại biểu đồ phù hợp (Appropriate Chart Type)

* **Nguyên tắc**: Chọn loại biểu đồ phù hợp với loại dữ liệu và mục tiêu phân tích để truyền tải thông tin một cách chính xác và trực quan.
* **Cách thực hiện**:
  + **Dữ liệu phân loại (categorical)**: Sử dụng bar chart, pie chart, hoặc treemap để so sánh hoặc hiển thị tỷ lệ.
  + **Dữ liệu số liên tục (numerical continuous)**: Sử dụng histogram, boxplot để phân tích phân bố, hoặc scatter plot để khám phá tương quan.
  + **Dữ liệu thời gian (time-series)**: Sử dụng line chart hoặc area chart để hiển thị xu hướng.
  + Xem xét số lượng biến và mục tiêu (phân bố, so sánh, xu hướng, tương quan) như đã đề cập trong câu trả lời trước.
* **Ví dụ**: Để phân tích phân bố doanh thu hàng ngày, sử dụng histogram thay vì pie chart, vì pie chart không phù hợp cho dữ liệu liên tục.
* **Tại sao quan trọng**: Loại biểu đồ sai có thể dẫn đến diễn giải sai hoặc làm mất đi ý nghĩa của dữ liệu.

### 3. Đảm bảo tính chính xác và trung thực (Accuracy and Integrity)

* **Nguyên tắc**: Biểu đồ phải phản ánh chính xác dữ liệu, không được bóp méo hoặc gây hiểu lầm.
* **Cách thực hiện**:
  + Sử dụng tỷ lệ trục phù hợp, tránh cắt trục y (ví dụ: bắt đầu từ giá trị khác 0 trong bar chart) trừ khi có lý do rõ ràng và được chú thích.
  + Không phóng đại hoặc thu nhỏ các hiệu ứng trực quan (như kích thước cột, điểm, hoặc vùng) để tránh đánh lừa người xem.
  + Ghi rõ nguồn dữ liệu và đơn vị đo (nếu có) để đảm bảo tính minh bạch.
* **Ví dụ**: Trong một bar chart so sánh doanh thu, đảm bảo trục y bắt đầu từ 0 để không phóng đại sự khác biệt giữa các cột.
* **Tại sao quan trọng**: Sự thiếu trung thực trong biểu đồ có thể dẫn đến các quyết định sai lầm, đặc biệt trong kinh doanh hoặc nghiên cứu.

### 4. Sử dụng màu sắc hiệu quả (Effective Use of Color)

* **Nguyên tắc**: Sử dụng màu sắc để làm nổi bật thông tin chính, phân biệt các yếu tố, và đảm bảo dễ nhìn, kể cả với người mù màu hoặc trong môi trường in ấn.
* **Cách thực hiện**:
  + **Giới hạn số lượng màu**: Sử dụng bảng màu đơn giản (thường 3-5 màu) để tránh gây rối mắt.
  + **Tương phản rõ ràng**: Chọn màu có độ tương phản cao để phân biệt các danh mục hoặc nhóm (ví dụ: xanh dương và cam thay vì xanh dương và xanh lam).
  + **Dùng màu có ý nghĩa**: Gán màu theo ngữ cảnh, ví dụ: đỏ cho giá trị tiêu cực, xanh lá cho tích cực.
  + **Hỗ trợ người mù màu**: Sử dụng các bảng màu thân thiện với người mù màu (như ColorBrewer) hoặc thêm texture/pattern để hỗ trợ.
  + **Kiểm tra in ấn**: Đảm bảo biểu đồ vẫn rõ ràng khi in đen trắng.
* **Ví dụ**: Trong heatmap biểu thị tương quan, sử dụng bảng màu từ xanh (tương quan âm) đến đỏ (tương quan dương) để dễ hiểu.
* **Tại sao quan trọng**: Màu sắc không phù hợp có thể làm giảm khả năng đọc hoặc gây nhầm lẫn.

### 5. Nhãn và chú thích rõ ràng (Clear Labeling and Annotations)

* **Nguyên tắc**: Cung cấp nhãn, tiêu đề, và chú thích đầy đủ để người xem hiểu ngay lập tức ý nghĩa của biểu đồ mà không cần giải thích thêm.
* **Cách thực hiện**:
  + Thêm tiêu đề ngắn gọn, mô tả rõ mục đích của biểu đồ (ví dụ: “Doanh Thu Theo Khu Vực Năm 2024”).
  + Nhãn trục x và y phải rõ ràng, bao gồm đơn vị đo (nếu có), ví dụ: “Doanh Thu (triệu VND)”.
  + Sử dụng chú thích (legend) khi cần phân biệt nhiều nhóm dữ liệu, đặt ở vị trí không che khuất dữ liệu.
  + Thêm annotation (chú thích trực tiếp) cho các điểm nổi bật, như giá trị ngoại lai hoặc đỉnh xu hướng.
* **Ví dụ**: Trong line chart xu hướng doanh thu, thêm nhãn “Đỉnh Doanh Thu” tại điểm cao nhất và ghi rõ giá trị.
* **Tại sao quan trọng**: Nhãn và chú thích giúp người xem không phải đoán ý nghĩa của biểu đồ, đặc biệt trong các báo cáo hoặc trình bày.

### 6. Tối ưu hóa tỷ lệ và bố cục (Proportionality and Layout)

* **Nguyên tắc**: Đảm bảo tỷ lệ và bố cục của biểu đồ cân đối, tránh làm méo mó dữ liệu hoặc gây khó khăn trong việc so sánh.
* **Cách thực hiện**:
  + Duy trì tỷ lệ khung hình (aspect ratio) phù hợp, ví dụ: không làm bar chart quá hẹp khiến các cột khó phân biệt.
  + Đảm bảo các yếu tố trực quan (cột, điểm, vùng) có kích thước tỷ lệ với giá trị dữ liệu.
  + Sắp xếp các danh mục hoặc điểm dữ liệu theo thứ tự hợp lý (ví dụ: theo thời gian, giá trị tăng dần/giảm dần).
  + Nếu có nhiều biểu đồ, sử dụng lưới (grid) để sắp xếp gọn gàng, như trong pair plot hoặc dashboard.
* **Ví dụ**: Trong bar chart, sắp xếp các cột theo thứ tự doanh thu giảm dần để dễ so sánh.
* **Tại sao quan trọng**: Tỷ lệ và bố cục không hợp lý có thể làm sai lệch nhận thức về dữ liệu hoặc gây khó khăn khi đọc.

### 7. Phù hợp với đối tượng người xem (Audience-Centric Design)

* **Nguyên tắc**: Thiết kế biểu đồ dựa trên nhu cầu và trình độ của đối tượng người xem, từ chuyên gia đến người không chuyên.
* **Cách thực hiện**:
  + Đối với **người không chuyên**: Sử dụng các biểu đồ đơn giản (bar chart, pie chart), ngôn ngữ dễ hiểu, và tránh thuật ngữ phức tạp.
  + Đối với **chuyên gia**: Có thể sử dụng các biểu đồ phức tạp hơn (boxplot, heatmap) và cung cấp thêm thông tin chi tiết (như giá trị thống kê).
  + Xem xét bối cảnh sử dụng: Biểu đồ cho báo cáo in cần độ phân giải cao, trong khi biểu đồ web cần tương tác (Plotly).
* **Ví dụ**: Trong báo cáo cho ban lãnh đạo, sử dụng bar chart đơn giản với màu sắc nổi bật thay vì pair plot phức tạp.
* **Tại sao quan trọng**: Một biểu đồ phù hợp với đối tượng sẽ tăng hiệu quả truyền đạt và tránh gây hiểu lầm.

### 8. Hỗ trợ tương tác (nếu cần) (Interactivity)

* **Nguyên tắc**: Với các biểu đồ trực tuyến hoặc dashboard, thêm tính năng tương tác để tăng khả năng khám phá dữ liệu.
* **Cách thực hiện**:
  + Sử dụng các thư viện như Plotly để tạo biểu đồ tương tác (phóng to, di chuột xem giá trị, lọc dữ liệu).
  + Thêm bộ lọc hoặc nút điều khiển để người dùng tự chọn dữ liệu (ví dụ: chọn khu vực trong dashboard doanh thu).
  + Đảm bảo tính tương tác không làm phức tạp hóa trải nghiệm người dùng.
* **Ví dụ**: Trong biểu đồ Plotly, thêm tooltip hiển thị giá trị doanh thu khi di chuột qua các điểm trên line chart.
* **Tại sao quan trọng**: Tương tác giúp người dùng khám phá dữ liệu sâu hơn, đặc biệt trong các ứng dụng web hoặc phân tích tương tác.

### 9. Kiểm tra khả năng tiếp cận (Accessibility)

* **Nguyên tắc**: Đảm bảo biểu đồ dễ tiếp cận với mọi người, kể cả những người có hạn chế về thị giác hoặc sử dụng thiết bị khác nhau.
* **Cách thực hiện**:
  + Sử dụng bảng màu thân thiện với người mù màu (ví dụ: từ ColorBrewer).
  + Đảm bảo độ tương phản cao giữa văn bản và nền.
  + Cung cấp phiên bản văn bản hoặc mô tả (alt text) cho biểu đồ khi sử dụng trên web.
  + Kiểm tra biểu đồ ở các định dạng (màn hình nhỏ, in đen trắng).
* **Ví dụ**: Sử dụng pattern (sọc, chấm) thay vì chỉ dựa vào màu để phân biệt các cột trong bar chart.
* **Tại sao quan trọng**: Tăng khả năng tiếp cận giúp biểu đồ hữu ích cho nhiều đối tượng hơn.

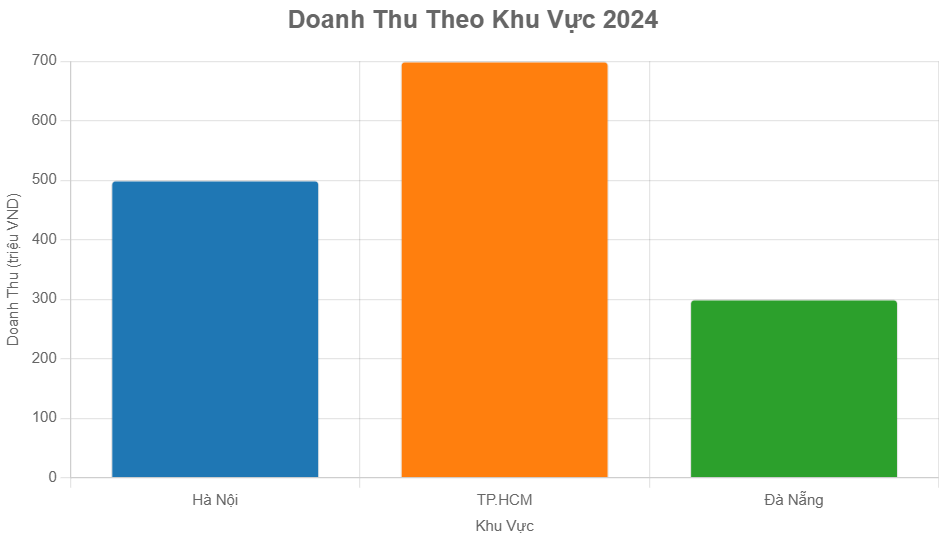
### 10. Kiểm tra và lặp lại (Test and Iterate)

* **Nguyên tắc**: Kiểm tra biểu đồ với người dùng thực tế và cải thiện dựa trên phản hồi để đảm bảo hiệu quả.
* **Cách thực hiện**:
  + Hỏi ý kiến người xem về tính dễ hiểu và thẩm mỹ của biểu đồ.
  + Kiểm tra biểu đồ trên các thiết bị khác nhau (máy tính, điện thoại, in ấn).
  + Thử nghiệm nhiều phiên bản (ví dụ: thay đổi màu sắc, loại biểu đồ) để tìm ra thiết kế tốt nhất.
* **Ví dụ**: Nếu người xem khó hiểu pie chart do quá nhiều danh mục, chuyển sang bar chart hoặc treemap.
* **Tại sao quan trọng**: Phản hồi từ người dùng giúp cải thiện biểu đồ, đảm bảo nó đạt được mục tiêu truyền tải thông tin.

Ví dụ minh hoạ:

Giả sử bạn cần trực quan hóa doanh thu của một công ty theo khu vực trong năm 2024:

* **Loại biểu đồ**: Chọn bar chart để so sánh doanh thu giữa các khu vực (Hà Nội, TP.HCM, Đà Nẵng).
* **Tối giản**: Chỉ hiển thị các cột, nhãn trục, và tiêu đề; không thêm hiệu ứng 3D.
* **Màu sắc**: Sử dụng màu tương phản (xanh dương, cam, xanh lá) và kiểm tra thân thiện với người mù màu.
* **Nhãn**: Thêm nhãn “Doanh Thu (triệu VND)” trên trục y, tiêu đề “Doanh Thu Theo Khu Vực 2024”.
* **Tính chính xác**: Đảm bảo trục y bắt đầu từ 0 để phản ánh đúng sự khác biệt.
* **Tương tác (nếu dùng Plotly)**: Thêm tooltip hiển thị giá trị doanh thu khi di chuột qua cột.



**Kết luận**

Tuân thủ các nguyên tắc trên giúp tạo ra biểu đồ không chỉ đẹp mà còn truyền tải thông tin một cách hiệu quả, dễ hiểu, và phù hợp với mục tiêu phân tích. Trong EDA, những nguyên tắc này đặc biệt quan trọng để khám phá dữ liệu một cách trực quan và hỗ trợ quá trình đưa ra quyết định.

## + Làm thế nào để tạo một biểu đồ đơn giản như histogram hoặc bar chart bằng Matplotlib? Bạn có thể chia sẻ đoạn code mẫu không?

### 1. Tạo Histogram bằng Matplotlib

**Histogram** được sử dụng để hiển thị phân bố của một biến định lượng liên tục, giúp khám phá hình dạng phân bố (như chuẩn, lệch, hoặc đa đỉnh) trong quá trình phân tích dữ liệu (EDA).

**Bước thực hiện**

1. **Chuẩn bị dữ liệu**: Sử dụng NumPy hoặc Pandas để tạo hoặc nhập dữ liệu.
2. **Tạo biểu đồ**: Sử dụng hàm plt.hist() để vẽ histogram.
3. **Tùy chỉnh**: Thêm nhãn, tiêu đề, và điều chỉnh các tham số như số lượng bins, màu sắc.
4. **Hiển thị**: Gọi plt.show() để hiển thị biểu đồ.

**Code mẫu**

python

import matplotlib.pyplot as plt

import numpy as np

*# Tạo dữ liệu mẫu: 1000 giá trị ngẫu nhiên theo phân bố chuẩn*

np.random.seed(42) *# Đảm bảo kết quả tái lập*

data = np.random.randn(1000) *# Dữ liệu ngẫu nhiên*

*# Tạo histogram*

plt.hist(data, bins=30, color='#1f77b4', alpha=0.7, edgecolor='black')

*# Tùy chỉnh biểu đồ*

plt.title('Histogram của Dữ Liệu Ngẫu Nhiên', fontsize=14, pad=10)

plt.xlabel('Giá trị', fontsize=12)

plt.ylabel('Tần suất', fontsize=12)

plt.grid(True, alpha=0.3) *# Thêm lưới nhẹ để dễ đọc*

*# Hiển thị biểu đồ*

plt.show()

**Giải thích**

* **Dữ liệu**: np.random.randn(1000) tạo 1000 giá trị ngẫu nhiên theo phân bố chuẩn.
* **Tham số bins**: Số lượng bins (30) xác định số cột trong histogram. Điều chỉnh số bins để cân bằng giữa chi tiết và độ mượt mà.
* **Màu sắc**: color='#1f77b4' (xanh dương) và alpha=0.7 (độ trong suốt) giúp biểu đồ dễ nhìn. edgecolor='black' làm nổi bật các cột.
* **Nhãn và tiêu đề**: Được thêm để đảm bảo biểu đồ rõ ràng, tuân thủ nguyên tắc nhãn rõ ràng.
* **Lưới**: grid=True với độ trong suốt nhẹ (alpha=0.3) hỗ trợ đọc tần suất mà không gây nhiễu.

**Trường hợp sử dụng**

* Phân tích phân bố của dữ liệu liên tục, như điểm thi, thu nhập, hoặc thời gian giao hàng.
* Kiểm tra xem dữ liệu có phân bố chuẩn hay lệch trong EDA.

### 2. Tạo Bar Chart bằng Matplotlib

**Bar chart** được sử dụng để so sánh giá trị giữa các danh mục (categorical data), như doanh thu theo khu vực hoặc số lượng sản phẩm theo loại.

**Bước thực hiện**

1. **Chuẩn bị dữ liệu**: Tạo danh sách các danh mục (labels) và giá trị tương ứng.
2. **Tạo biểu đồ**: Sử dụng hàm plt.bar() để vẽ bar chart.
3. **Tùy chỉnh**: Thêm nhãn trục, tiêu đề, và điều chỉnh màu sắc, kích thước cột.
4. **Hiển thị**: Gọi plt.show() để hiển thị.

**Code mẫu**

python

import matplotlib.pyplot as plt

*# Dữ liệu mẫu: Doanh thu theo khu vực*

categories = ['Hà Nội', 'TP.HCM', 'Đà Nẵng', 'Cần Thơ']

values = [500, 700, 300, 400] *# Doanh thu (triệu VND)*

*# Tạo bar chart*

plt.bar(categories, values, color='#ff7f0e', edgecolor='black', width=0.4)

*# Tùy chỉnh biểu đồ*

plt.title('Doanh Thu Theo Khu Vực Năm 2024', fontsize=14, pad=10)

plt.xlabel('Khu Vực', fontsize=12)

plt.ylabel('Doanh Thu (triệu VND)', fontsize=12)

plt.grid(True, axis='y', alpha=0.3) *# Lưới theo trục y*

*# Hiển thị biểu đồ*

plt.show()

**Giải thích**

* **Dữ liệu**: categories là danh sách các khu vực, values là doanh thu tương ứng.
* **Tham số width**: width=0.4 điều chỉnh độ rộng cột để tránh quá sát nhau.
* **Màu sắc**: color='#ff7f0e' (màu cam) và edgecolor='black' tạo độ tương phản và rõ ràng.
* **Nhãn và tiêu đề**: Được thêm để mô tả rõ ràng mục đích biểu đồ.
* **Lưới**: Chỉ thêm lưới trên trục y (axis='y') để hỗ trợ so sánh giá trị mà không làm rối biểu đồ.

**Trường hợp sử dụng**

* So sánh giá trị giữa các danh mục, như doanh thu, số lượng khách hàng, hoặc sản phẩm bán ra.
* Trình bày kết quả tổng hợp trong báo cáo hoặc phân tích EDA.

### 3. Các nguyên tắc thiết kế được áp dụng

Cả hai đoạn code trên đều tuân thủ các nguyên tắc thiết kế trực quan hóa:

* **Tối giản**: Chỉ sử dụng các yếu tố cần thiết (nhãn, tiêu đề, lưới nhẹ).
* **Chọn loại biểu đồ phù hợp**: Histogram cho dữ liệu số liên tục, bar chart cho dữ liệu phân loại.
* **Màu sắc hiệu quả**: Sử dụng màu đơn giản, tương phản cao, và thân thiện với người xem.
* **Nhãn rõ ràng**: Tiêu đề và nhãn trục mô tả chính xác nội dung.
* **Tính chính xác**: Trục y bắt đầu từ 0 trong bar chart để tránh phóng đại; số bins hợp lý trong histogram.

### 4. Mẹo để cải thiện biểu đồ

* **Tùy chỉnh thêm**:
  + Thêm chú thích (annotation) để làm nổi bật giá trị quan trọng, ví dụ: giá trị cao nhất trong bar chart.
  + Sử dụng bảng màu từ Matplotlib hoặc Seaborn để tăng tính thẩm mỹ.
* **Kết hợp Pandas**: Nếu dữ liệu ở dạng DataFrame, sử dụng df.plot(kind='bar') hoặc df.hist() để đơn giản hóa.
* **Kiểm tra trên nhiều nền tảng**: Đảm bảo biểu đồ rõ ràng khi xuất ra file (PNG, PDF) hoặc hiển thị trên web.
* **Tương tác (nếu cần)**: Nếu muốn biểu đồ tương tác, cân nhắc chuyển sang Plotly (như đã thảo luận trong câu hỏi trước).

### 5. Mở rộng: Tùy chỉnh sâu hơn

Nếu bạn muốn tùy chỉnh thêm, dưới đây là một ví dụ mở rộng cho bar chart với chú thích giá trị trên cột:

python

import matplotlib.pyplot as plt

*# Dữ liệu mẫu*

categories = ['Hà Nội', 'TP.HCM', 'Đà Nẵng', 'Cần Thơ']

values = [500, 700, 300, 400]

*# Tạo bar chart*

bars = plt.bar(categories, values, color='#ff7f0e', edgecolor='black', width=0.4)

*# Thêm giá trị trên đầu mỗi cột*

for bar in bars:

yval = bar.get\_height()

plt.text(bar.get\_x() + bar.get\_width()/2, yval + 10, f'{yval}', ha='center', fontsize=10)

*# Tùy chỉnh biểu đồ*

plt.title('Doanh Thu Theo Khu Vực Năm 2024', fontsize=14, pad=10)

plt.xlabel('Khu Vực', fontsize=12)

plt.ylabel('Doanh Thu (triệu VND)', fontsize=12)

plt.grid(True, axis='y', alpha=0.3)

*# Hiển thị biểu đồ*

plt.show()

**Giải thích bổ sung**

* **Chú thích giá trị**: Hàm plt.text() thêm giá trị trên đầu mỗi cột, với ha='center' để căn giữa và yval + 10 để đặt nhãn phía trên cột.
* **Tính thẩm mỹ**: Khoảng cách +10 đảm bảo nhãn không chồng lấn cột.

### 6. Khi nào sử dụng Histogram và Bar Chart

* **Histogram**: Dùng khi cần phân tích phân bố của dữ liệu liên tục (như tuổi, thu nhập). Không phù hợp cho dữ liệu phân loại.
* **Bar Chart**: Dùng khi so sánh giá trị giữa các danh mục (như khu vực, loại sản phẩm). Không phù hợp cho dữ liệu liên tục.

### 7. Kết luận

Matplotlib là công cụ mạnh mẽ và linh hoạt để tạo histogram và bar chart, đặc biệt trong EDA. Code mẫu trên cung cấp cách tiếp cận đơn giản nhưng đầy đủ, dễ dàng tùy chỉnh theo nhu cầu. Nếu bạn có dữ liệu cụ thể (ví dụ: tập dữ liệu doanh thu hoặc điểm số) và muốn tạo biểu đồ với các thông số khác (màu sắc, kích thước, kiểu biểu đồ), hãy cung cấp chi tiết để tôi tạo code phù hợp hơn!

Nếu bạn muốn tạo biểu đồ tương tự bằng **Chart.js** (như trong các câu hỏi trước), tôi cũng có thể chuyển đổi code trên thành định dạng Chart.js. Ví dụ, bar chart tương tự như trên trong Chart.js:

## + Làm thế nào để xuất biểu đồ từ Python ra các định dạng như PNG, PDF hoặc HTML để sử dụng trong báo cáo?

Xuất biểu đồ từ Python ra các định dạng như **PNG**, **PDF**, hoặc **HTML** là một bước quan trọng để sử dụng trong báo cáo, bài thuyết trình, hoặc chia sẻ trên web. Các thư viện trực quan hóa phổ biến như **Matplotlib**, **Seaborn**, và **Plotly** đều hỗ trợ xuất biểu đồ sang nhiều định dạng.

### 1. Xuất biểu đồ bằng Matplotlib/Seaborn

**Matplotlib** (và **Seaborn**, vì Seaborn dựa trên Matplotlib) hỗ trợ xuất biểu đồ sang các định dạng như PNG, PDF, SVG, và nhiều định dạng khác thông qua hàm plt.savefig(). Đây là cách phổ biến để tạo biểu đồ tĩnh cho báo cáo hoặc tài liệu in ấn.

**Bước thực hiện**

1. **Tạo biểu đồ**: Sử dụng Matplotlib/Seaborn để vẽ biểu đồ (ví dụ: histogram, bar chart).
2. **Tùy chỉnh chất lượng**: Điều chỉnh độ phân giải (DPI) và kích thước biểu đồ để phù hợp với mục đích sử dụng.
3. **Xuất file**: Gọi plt.savefig() với định dạng mong muốn trước khi gọi plt.show() (vì plt.show() có thể xóa bộ nhớ đệm của biểu đồ).
4. **Đóng biểu đồ (nếu cần)**: Sử dụng plt.close() để tránh hiển thị biểu đồ trên màn hình nếu chỉ cần xuất file.

**Code mẫu: Xuất Histogram sang PNG và PDF**

python

import matplotlib.pyplot as plt

import numpy as np

*# Tạo dữ liệu mẫu*

np.random.seed(42)

data = np.random.randn(1000)

*# Tạo histogram*

plt.figure(figsize=(8, 6)) *# Kích thước biểu đồ (inch)*

plt.hist(data, bins=30, color='#1f77b4', alpha=0.7, edgecolor='black')

plt.title('Histogram của Dữ Liệu Ngẫu Nhiên', fontsize=14)

plt.xlabel('Giá trị', fontsize=12)

plt.ylabel('Tần suất', fontsize=12)

plt.grid(True, alpha=0.3)

*# Xuất sang PNG*

plt.savefig('histogram.png', dpi=300, bbox\_inches='tight')

*# Xuất sang PDF*

plt.savefig('histogram.pdf', bbox\_inches='tight')

*# Hiển thị biểu đồ (tùy chọn)*

plt.show()

*# Đóng biểu đồ*

plt.close()

**Giải thích**

* **Định dạng file**:
  + histogram.png: PNG là định dạng phổ biến cho báo cáo hoặc web, hỗ trợ hình ảnh bitmap chất lượng cao.
  + histogram.pdf: PDF là định dạng vector, lý tưởng cho in ấn hoặc tài liệu chất lượng cao.
* **Tham số dpi**: Độ phân giải (dots per inch), ví dụ dpi=300 đảm bảo chất lượng cao cho in ấn. Để sử dụng trên web, có thể giảm xuống dpi=100 để giảm kích thước file.
* **Tham số bbox\_inches='tight'**: Tự động điều chỉnh lề để đảm bảo các nhãn và tiêu đề không bị cắt.
* **Gọi trước plt.show()**: Nếu gọi plt.show() trước, biểu đồ có thể bị xóa khỏi bộ nhớ, dẫn đến file xuất rỗng.
* **Kích thước biểu đồ**: figsize=(8, 6) đặt kích thước biểu đồ (rộng x cao, tính bằng inch), có thể điều chỉnh tùy mục đích.

**Code mẫu: Xuất Bar Chart sang PNG**

python

import matplotlib.pyplot as plt

*# Dữ liệu mẫu*

categories = ['Hà Nội', 'TP.HCM', 'Đà Nẵng', 'Cần Thơ']

values = [500, 700, 300, 400]

*# Tạo bar chart*

plt.figure(figsize=(8, 6))

plt.bar(categories, values, color='#ff7f0e', edgecolor='black', width=0.4)

plt.title('Doanh Thu Theo Khu Vực Năm 2024', fontsize=14)

plt.xlabel('Khu Vực', fontsize=12)

plt.ylabel('Doanh Thu (triệu VND)', fontsize=12)

plt.grid(True, axis='y', alpha=0.3)

*# Xuất sang PNG*

plt.savefig('barchart.png', dpi=200, bbox\_inches='tight')

*# Xuất sang PDF*

plt.savefig('barchart.pdf', bbox\_inches='tight')

*# Hiển thị biểu đồ*

plt.show()

plt.close()

**Các định dạng khác**

* **SVG**: plt.savefig('chart.svg', bbox\_inches='tight') – Định dạng vector, phù hợp cho chỉnh sửa trong các công cụ như Adobe Illustrator.
* **JPG**: plt.savefig('chart.jpg', dpi=150, quality=95) – Chất lượng thấp hơn PNG, phù hợp cho web khi cần kích thước file nhỏ.

**Lưu ý**

* **Chất lượng in ấn**: Sử dụng DPI cao (200-300) cho PNG/PDF khi in. SVG và PDF là vector nên không phụ thuộc DPI.
* **Đường dẫn file**: Đảm bảo đường dẫn lưu file hợp lệ (ví dụ: 'output/chart.png' nếu lưu vào thư mục con).
* **Seaborn**: Vì Seaborn dựa trên Matplotlib, cách xuất tương tự, chỉ cần đảm bảo gọi plt.savefig().

### 2. Xuất biểu đồ tương tác bằng Plotly

**Plotly** là lựa chọn lý tưởng để tạo biểu đồ tương tác và xuất sang **HTML** hoặc các định dạng tĩnh như PNG, PDF. Plotly hỗ trợ xuất biểu đồ tương tác sang HTML để sử dụng trên web hoặc trong dashboard, và cũng có thể xuất sang định dạng tĩnh thông qua công cụ như kaleido.

**Bước thực hiện**

1. **Tạo biểu đồ**: Sử dụng Plotly Express hoặc Plotly Graph Objects để vẽ biểu đồ.
2. **Xuất sang HTML**: Sử dụng fig.write\_html() để tạo file HTML chứa biểu đồ tương tác.
3. **Xuất sang PNG/PDF**: Sử dụng fig.write\_image() với engine kaleido (cần cài đặt).
4. **Tùy chỉnh**: Điều chỉnh kích thước và tỷ lệ để phù hợp với báo cáo.

**Yêu cầu cài đặt**

Cần cài đặt Plotly và Kaleido:

bash

pip install plotly kaleido

**Code mẫu: Xuất Bar Chart sang PNG, PDF, và HTML**

python

import plotly.express as px

import pandas as pd

*# Dữ liệu mẫu*

df = pd.DataFrame({

'Khu Vực': ['Hà Nội', 'TP.HCM', 'Đà Nẵng', 'Cần Thơ'],

'Doanh Thu': [500, 700, 300, 400]

})

*# Tạo bar chart*

fig = px.bar(df, x='Khu Vực', y='Doanh Thu',

title='Doanh Thu Theo Khu Vực Năm 2024',

color\_discrete\_sequence=['#ff7f0e'])

*# Tùy chỉnh biểu đồ*

fig.update\_layout(

xaxis\_title='Khu Vực',

yaxis\_title='Doanh Thu (triệu VND)',

showlegend=False,

width=800, height=600

)

*# Xuất sang HTML*

fig.write\_html('barchart.html')

*# Xuất sang PNG*

fig.write\_image('barchart.png', engine='kaleido', scale=2)

*# Xuất sang PDF*

fig.write\_image('barchart.pdf', engine='kaleido', scale=2)

*# Hiển thị biểu đồ*

fig.show()

**Giải thích**

* **HTML**: File barchart.html chứa biểu đồ tương tác, có thể mở trong trình duyệt và hỗ trợ phóng to, di chuột để xem giá trị.
* **PNG/PDF**: write\_image() yêu cầu kaleido để xuất biểu đồ tĩnh. Tham số scale=2 tăng độ phân giải (tương đương DPI cao).
* **Kích thước**: width=800, height=600 đặt kích thước biểu đồ (tính bằng pixel).
* **Tùy chỉnh**: color\_discrete\_sequence và các tham số layout đảm bảo biểu đồ tuân thủ nguyên tắc màu sắc và nhãn rõ ràng.

**Code mẫu: Xuất Histogram sang HTML**

python

import plotly.express as px

import numpy as np

*# Tạo dữ liệu mẫu*

np.random.seed(42)

data = np.random.randn(1000)

*# Tạo histogram*

fig = px.histogram(x=data, nbins=30, title='Histogram của Dữ Liệu Ngẫu Nhiên',

color\_discrete\_sequence=['#1f77b4'])

*# Tùy chỉnh*

fig.update\_layout(

xaxis\_title='Giá trị',

yaxis\_title='Tần suất',

width=800, height=600

)

*# Xuất sang HTML*

fig.write\_html('histogram.html')

*# Xuất sang PNG*

fig.write\_image('histogram.png', engine='kaleido', scale=2)

*# Hiển thị biểu đồ*

fig.show()

**Lưu ý**

* **Cài đặt Kaleido**: Nếu gặp lỗi khi xuất PNG/PDF, kiểm tra xem Kaleido đã được cài đặt đúng (pip install kaleido).
* **HTML tương tác**: File HTML rất hữu ích cho báo cáo web, dashboard, hoặc chia sẻ trực tuyến, nhưng cần trình duyệt để xem.
* **Hiệu suất**: Xuất PNG/PDF với Plotly có thể chậm hơn Matplotlib với dữ liệu lớn.

### 3. So sánh Matplotlib và Plotly khi xuất biểu đồ

|  |  |  |
| --- | --- | --- |
| Tiêu chí | Matplotlib | Plotly |
| Định dạng xuất | PNG, PDF, SVG, JPG, v.v. | PNG, PDF, HTML, JPG |
| Tương tác | Không (chỉ tĩnh) | Có (HTML), tĩnh (PNG/PDF) |
| Chất lượng | Cao, tùy chỉnh DPI | Cao, tùy chỉnh scale |
| Dễ sử dụng | Đơn giản với plt.savefig() | Dễ với write\_html/write\_image |
| Hiệu suất | Nhanh với dữ liệu lớn | Chậm hơn với dữ liệu lớn |
| Trường hợp sử dụng | Báo cáo in, bài báo khoa học | Web, dashboard, báo cáo tương tác |

### 4. Các nguyên tắc thiết kế khi xuất biểu đồ

Khi xuất biểu đồ, cần đảm bảo tuân thủ các nguyên tắc thiết kế trực quan hóa:

* **Chất lượng cao**: Sử dụng DPI cao (200-300) hoặc scale cao (2-3) cho PNG/PDF để đảm bảo rõ nét khi in.
* **Nhãn rõ ràng**: Đảm bảo tiêu đề, nhãn trục, và chú thích hiển thị đầy đủ, không bị cắt (sử dụng bbox\_inches='tight' trong Matplotlib).
* **Màu sắc**: Chọn màu tương phản cao, thân thiện với người mù màu, và kiểm tra xem biểu đồ có rõ ràng khi in đen trắng không.
* **Kích thước**: Điều chỉnh kích thước biểu đồ (figsize hoặc width/height) phù hợp với báo cáo (ví dụ: A4, slide PowerPoint).
* **Tính trung thực**: Đảm bảo trục và tỷ lệ không làm méo mó dữ liệu.

### 5. Mẹo nâng cao

* **Tự động hóa**: Sử dụng vòng lặp để xuất nhiều biểu đồ (ví dụ: một biểu đồ cho mỗi khu vực).
* **Kiểm tra trước khi xuất**: Xem trước biểu đồ bằng plt.show() hoặc fig.show() để đảm bảo đúng thiết kế.
* **Tổ chức file**: Lưu file vào thư mục cụ thể (ví dụ: 'output/histogram.png') để dễ quản lý.
* **Kết hợp với báo cáo**: Nhúng file PNG/PDF vào báo cáo Word/PowerPoint hoặc file HTML vào trang web.

### 6. Kết luận

* **Matplotlib** lý tưởng cho các biểu đồ tĩnh chất lượng cao trong báo cáo in ấn (PNG, PDF).
* **Plotly** phù hợp cho các báo cáo tương tác (HTML) hoặc khi cần kết hợp tĩnh và tương tác.

# PHẦN 3: PHÂN TÍCH ĐƠN BIẾN VÀ HAI BIẾN

## + Phân tích đơn biến (univariate analysis) là gì? Nó khác gì với phân tích hai biến (bivariate analysis) trong khám phá dữ liệu?

### Phân tích hai biến (Bivariate Analysis) là gì?

**Phân tích đơn biến** (Univariate Analysis) là một phần của khám phá dữ liệu (Exploratory Data Analysis - EDA), tập trung vào việc phân tích và mô tả một biến duy nhất trong tập dữ liệu, mà không xem xét mối quan hệ với các biến khác. Mục tiêu chính là hiểu rõ đặc điểm nội tại của biến đó, bao gồm phân bố, xu hướng trung tâm, độ phân tán và các đặc trưng khác. Đây là bước đầu tiên trong EDA, giúp phát hiện vấn đề dữ liệu (như giá trị ngoại lai hoặc dữ liệu thiếu) và cung cấp cái nhìn tổng quan về dữ liệu.

### Các thành phần chính của phân tích đơn biến

1. **Thống kê mô tả**:
   * **Xu hướng trung tâm**: Mean (trung bình), median (trung vị), mode (giá trị phổ biến nhất).
   * **Độ phân tán**: Standard deviation (độ lệch chuẩn), variance (phương sai), range (khoảng giá trị), IQR (khoảng tứ phân vị).
   * **Phân vị**: Tứ phân vị (Q1, Q3), percentiles để hiểu phân bố.
2. **Trực quan hóa**:
   * **Histogram**: Hiển thị phân bố tần suất của biến định lượng liên tục.
   * **Boxplot**: Tóm tắt phân bố, phát hiện ngoại lai.
   * **Bar chart**: Với dữ liệu phân loại (categorical), hiển thị tần suất từng danh mục.

**Ví dụ**

Giả sử bạn có dữ liệu về tuổi của 100 khách hàng: [25, 30, 35, ..., 60]. Phân tích đơn biến có thể bao gồm:

* Mean: 40 tuổi.
* Median: 38 tuổi.
* Standard deviation: 10 tuổi (cho thấy độ phân tán).
* Histogram: Hiển thị phân bố tuổi tập trung quanh 30-50 tuổi.
* Code mẫu bằng Matplotlib để vẽ histogram:

python

import matplotlib.pyplot as plt

import numpy as np

*# Dữ liệu mẫu*

ages = np.random.normal(40, 10, 100) *# Phân bố chuẩn với mean=40, std=10*

*# Vẽ histogram*

plt.hist(ages, bins=10, color='#1f77b4', edgecolor='black')

plt.title('Phân Bố Tuổi Khách Hàng')

plt.xlabel('Tuổi')

plt.ylabel('Tần Suất')

plt.savefig('univariate\_histogram.png', dpi=300, bbox\_inches='tight')

plt.show()

**Ứng dụng**

* Kiểm tra chất lượng dữ liệu (phát hiện outliers qua boxplot).
* Tóm tắt dữ liệu cho báo cáo ban đầu.
* Xác định loại phân bố (chuẩn, lệch) để chọn phương pháp phân tích tiếp theo.

### Phân tích hai biến (Bivariate Analysis) là gì?

**Phân tích hai biến** (Bivariate Analysis) là bước tiếp theo trong EDA, tập trung vào việc khám phá mối quan hệ giữa **hai biến** trong tập dữ liệu. Nó giúp phát hiện tương quan, sự phụ thuộc, hoặc xu hướng giữa hai biến, từ đó tạo giả thuyết cho các phân tích sâu hơn (như hồi quy).

### Các thành phần chính của phân tích hai biến

1. **Thống kê mô tả**:
   * **Tương quan (Correlation)**: Hệ số tương quan Pearson (cho biến định lượng), Spearman (cho dữ liệu không tuyến tính), hoặc chi-square (cho dữ liệu phân loại).
   * **Phân tích chéo (Cross-tabulation)**: Với dữ liệu phân loại, tính tần suất hoặc tỷ lệ giữa hai biến.
2. **Trực quan hóa**:
   * **Scatter plot**: Hiển thị mối quan hệ giữa hai biến định lượng (ví dụ: tuổi và thu nhập).
   * **Boxplot nhóm**: So sánh phân bố của một biến định lượng theo nhóm của biến phân loại.
   * **Heatmap**: Hiển thị ma trận tương quan nếu có nhiều cặp biến.

**Ví dụ**

Tiếp tục với dữ liệu tuổi, giả sử bạn có thêm biến thu nhập (triệu VND): [(25, 20), (30, 25), (35, 30), ...]. Phân tích hai biến có thể bao gồm:

* Hệ số tương quan Pearson: 0.75 (tương quan dương mạnh giữa tuổi và thu nhập).
* Scatter plot: Hiển thị xu hướng tăng dần của thu nhập theo tuổi.
* Code mẫu bằng Matplotlib để vẽ scatter plot:

python

import matplotlib.pyplot as plt

import numpy as np

*# Dữ liệu mẫu*

ages = np.random.normal(40, 10, 100)

incomes = 0.5 \* ages + np.random.normal(0, 5, 100) *# Tương quan dương*

*# Vẽ scatter plot*

plt.scatter(ages, incomes, color='#ff7f0e', alpha=0.7)

plt.title('Mối Quan Hệ Giữa Tuổi và Thu Nhập')

plt.xlabel('Tuổi')

plt.ylabel('Thu Nhập (triệu VND)')

plt.savefig('bivariate\_scatter.png', dpi=300, bbox\_inches='tight')

plt.show()

**Ứng dụng**

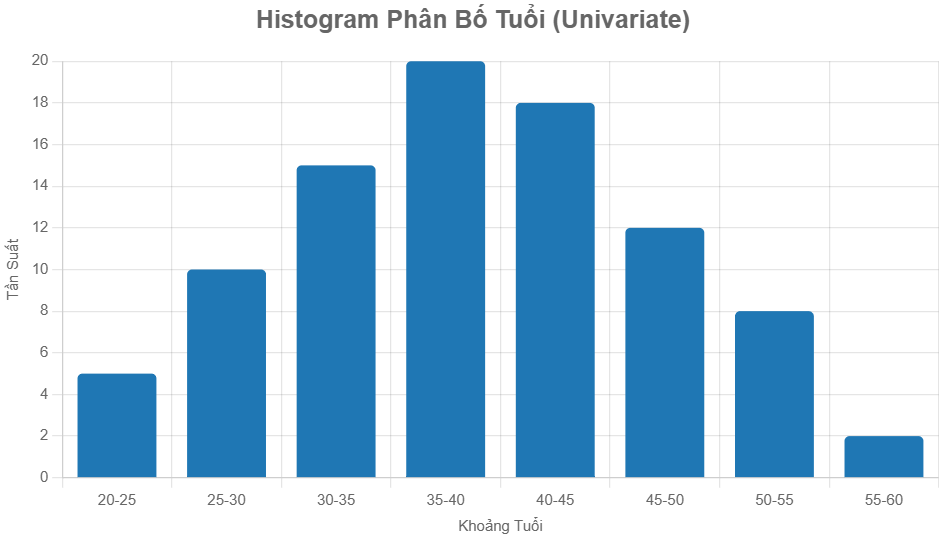
* Phát hiện mối quan hệ tiềm ẩn (ví dụ: thu nhập tăng theo tuổi).
* Hỗ trợ kiểm tra giả thuyết (như kiểm định chi-square cho dữ liệu phân loại).
* Xây dựng nền tảng cho mô hình hóa (như hồi quy).

### Sự khác biệt giữa Phân tích đơn biến và Phân tích hai biến trong EDA

|  |  |  |
| --- | --- | --- |
| Tiêu chí | Phân tích đơn biến | Phân tích hai biến |
| Số lượng biến | Một biến duy nhất. | Hai biến. |
| Mục tiêu | Mô tả đặc điểm nội tại (phân bố, trung tâm). | Khám phá mối quan hệ, tương quan giữa hai biến. |
| Công cụ thống kê | Mean, median, std, IQR. | Correlation, chi-square, cross-tabulation. |
| Trực quan hóa chính | Histogram, boxplot, bar chart. | Scatter plot, boxplot nhóm, heatmap. |
| Ví dụ câu hỏi | “Phân bố tuổi khách hàng như thế nào?” | “Tuổi có ảnh hưởng đến thu nhập không?” |
| Vai trò trong EDA | Bước đầu: Hiểu từng biến riêng lẻ | Bước sau: Tìm mối liên hệ giữa các biến |
| Rủi ro nếu bỏ qua | Không hiểu cấu trúc dữ liệu cơ bản | Bỏ lỡ tương quan, dẫn đến mô hình sai lệch |

### Minh họa bằng biểu đồ

Để minh họa sự khác biệt, dưới đây là biểu đồ histogram (đơn biến) và scatter plot (hai biến) cho dữ liệu tuổi và thu nhập:



### Kết luận

Phân tích đơn biến là nền tảng để hiểu từng biến riêng lẻ, trong khi phân tích hai biến mở rộng để khám phá mối quan hệ, giúp EDA trở nên toàn diện hơn. Trong thực tế, chúng thường được thực hiện tuần tự: bắt đầu từ đơn biến để làm sạch dữ liệu, sau đó chuyển sang hai biến để tạo insight.

## + Các thước đo thống kê nào thường được sử dụng trong phân tích đơn biến (ví dụ: trung bình, trung vị, mode, độ lệch chuẩn)?

**Phân tích đơn biến** (Univariate Analysis) trong khám phá dữ liệu (EDA) tập trung vào việc mô tả một biến duy nhất, giúp hiểu phân bố và đặc điểm cơ bản của nó. Các thước đo thống kê chính được dùng để tóm tắt dữ liệu một cách đơn giản, chia thành các nhóm: xu hướng trung tâm, độ phân tán, phân vị, và tần suất (đặc biệt với dữ liệu phân loại). Những thước đo này giúp phát hiện vấn đề như ngoại lai và hỗ trợ quyết định phân tích tiếp theo.

Dưới đây là các thước đo phổ biến nhất, kèm ví dụ đơn giản với dữ liệu tuổi của 100 khách hàng (phân bố quanh 40 tuổi).

### 1. Thước đo xu hướng trung tâm (Measures of Central Tendency)

Những thước đo này chỉ ra giá trị "đại diện" nhất của biến.

* **Trung bình (Mean)**: Giá trị trung bình của tất cả dữ liệu. Ví dụ: Tuổi trung bình ≈ 40 tuổi.
* **Trung vị (Median)**: Giá trị ở giữa khi sắp xếp dữ liệu. Ví dụ: Tuổi trung vị ≈ 39.5 tuổi (ít bị ảnh hưởng bởi ngoại lai).
* **Mode (Giá trị phổ biến nhất)**: Giá trị xuất hiện nhiều nhất. Ví dụ: Tuổi phổ biến nhất ≈ 38 tuổi.

### 2. Thước đo độ phân tán (Measures of Dispersion)

Những thước đo này cho thấy dữ liệu lan tỏa như thế nào.

* **Khoảng giá trị (Range)**: Chênh lệch giữa giá trị lớn nhất và nhỏ nhất. Ví dụ: 60 - 20 = 40 tuổi.
* **Độ lệch chuẩn (Standard Deviation)**: Mức độ dao động quanh trung bình. Ví dụ: ≈ 10 tuổi (dữ liệu thường dao động ±10 tuổi quanh 40).
* **Phương sai (Variance)**: Đo lường biến thiên tổng thể (thường dùng cùng độ lệch chuẩn). Ví dụ: ≈ 100 (tuổi²).

### 3. Thước đo phân vị (Measures of Position)

Những thước đo này xác định vị trí của dữ liệu trong phân bố.

* **Tứ phân vị (Quartiles)**: Chia dữ liệu thành 4 phần: Q1 (25%), Q2 (median, 50%), Q3 (75%). Ví dụ: Q1 ≈ 32 tuổi, Q3 ≈ 48 tuổi.
* **Khoảng tứ phân vị (IQR)**: Khoảng giữa Q3 và Q1. Ví dụ: ≈ 16 tuổi (giúp phát hiện ngoại lai).

### 4. Thước đo tần suất (Frequency Measures) – Với dữ liệu phân loại

* **Tần suất tuyệt đối**: Số lần xuất hiện của từng danh mục. Ví dụ: Nam: 60, Nữ: 40.
* **Tần suất tương đối**: Tỷ lệ phần trăm. Ví dụ: Nam: 60%, Nữ: 40%.

### Bảng tóm tắt các thước đo phổ biến

|  |  |  |  |
| --- | --- | --- | --- |
| Nhóm thước đo | Thước đo | Loại dữ liệu phù hợp | Ví dụ ứng dụng |
| Xu hướng trung tâm | Mean, Median, Mode | Định lượng/Định tính | Tuổi trung bình của khách hàng |
| Độ phân tán | Range, Std, Variance | Định lượng | Độ lan tỏa điểm số thi |
| Phân vị | Quartiles, IQR | Định lượng | Phân bố lương theo phần trăm |
| Tần suất | Frequency, Relative Freq. | Định loại | Tỷ lệ khách theo khu vực |

### Ví dụ tính toán đơn giản bằng Python

Sử dụng Pandas để tính nhanh (chạy trong Jupyter hoặc script):

python

import pandas as pd

import numpy as np

*# Dữ liệu mẫu*

ages = np.random.normal(40, 10, 100)

df = pd.DataFrame({'Tuoi': ages})

*# Tính các thước đo*

print(df.describe()) *# Mean, std, quartiles*

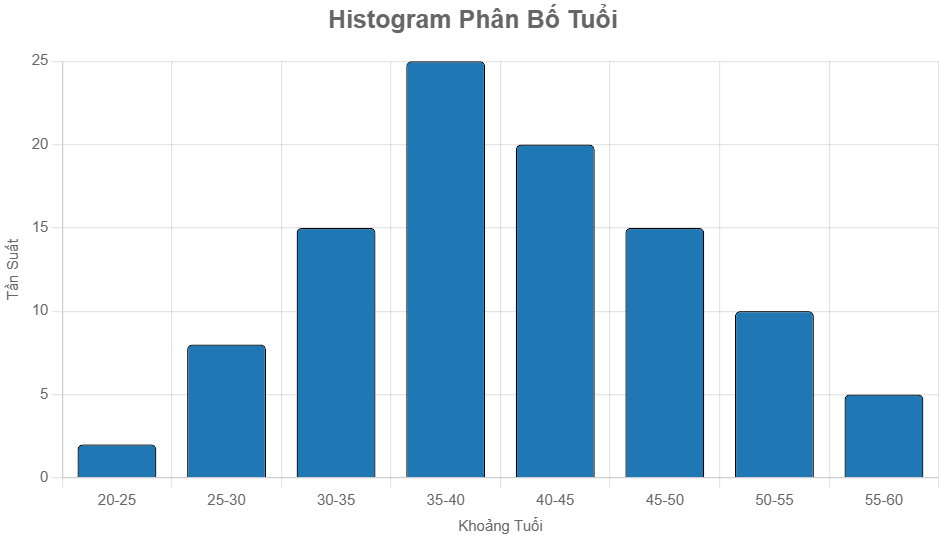
print(f"Median: {df['Tuoi'].median():.2f}")

print(f"Mode: {df['Tuoi'].mode()[0]:.2f}")

**Kết quả mẫu**: Mean ≈ 39.71, Std ≈ 9.88, IQR ≈ 13.03.

### Minh họa trực quan

Histogram là cách tốt để xem phân bố. Dưới đây là ví dụ cho dữ liệu tuổi:



### Lưu ý khi sử dụng

* Chọn thước đo phù hợp: Mean và std cho dữ liệu chuẩn; median và IQR cho dữ liệu lệch hoặc có ngoại lai.
* Kết hợp với biểu đồ: Histogram cho phân bố, boxplot cho quartiles.
* Trong EDA: Những thước đo này giúp làm sạch dữ liệu trước khi phân tích sâu hơn.

## + Trong phân tích hai biến, làm thế nào để xác định mối quan hệ giữa hai biến (ví dụ: tương quan, nhân quả)?

**Phân tích hai biến** (Bivariate Analysis) trong khám phá dữ liệu (EDA) nhằm khám phá mối liên hệ giữa hai biến, giúp phát hiện xu hướng, tương quan hoặc sự phụ thuộc. Mối quan hệ có thể là **tương quan** (correlation – mức độ liên hệ tuyến tính hoặc không tuyến tính) hoặc **nhân quả** (causation – một biến gây ra thay đổi ở biến kia). Tuy nhiên, tương quan không đồng nghĩa với nhân quả ("correlation does not imply causation"), vì có thể có biến nhiễu (confounding variables) hoặc mối quan hệ giả tạo. Dưới đây là cách xác định từng loại, với các bước thực hiện cụ thể.

### 1. Xác định tương quan (Correlation)

Tương quan đo lường mức độ hai biến thay đổi cùng nhau, thường dùng cho dữ liệu định lượng. Mức độ: từ -1 (tương quan âm hoàn hảo) đến +1 (tương quan dương hoàn hảo), 0 là không tương quan.

**Các bước xác định**:

* **Bước 1: Trực quan hóa dữ liệu**:
  + Sử dụng **scatter plot** để quan sát xu hướng (tuyến tính, phi tuyến, hoặc không có).
  + Với dữ liệu phân loại và định lượng: **boxplot nhóm** để so sánh phân bố.
  + Với hai biến phân loại: **bảng chéo (cross-tabulation)** hoặc **bar chart chồng** để xem tần suất chung.
* **Bước 2: Tính hệ số tương quan**:
  + **Pearson correlation**: Cho dữ liệu định lượng tuyến tính (giả định phân bố chuẩn).
  + **Spearman correlation**: Cho dữ liệu không tuyến tính hoặc thứ hạng (ordinal).
  + **Chi-square test**: Cho hai biến phân loại, kiểm tra sự độc lập.
  + Kiểm tra ý nghĩa thống kê qua **p-value** (thường < 0.05 để bác bỏ không tương quan).
* **Bước 3: Diễn giải**:
  + |Hệ số| > 0.7: Tương quan mạnh.
  + 0.3 < |Hệ số| < 0.7: Trung bình.
  + |Hệ số| < 0.3: Yếu.

**Ví dụ**: Với dữ liệu tuổi (biến X) và thu nhập (biến Y) của 100 khách hàng, scatter plot cho thấy xu hướng tăng dần. Tính toán cho thấy hệ số Pearson ≈ 0.636 (tương quan dương trung bình), p-value ≈ 0.000 (ý nghĩa thống kê cao).

### 2. Xác định nhân quả (Causation)

Nhân quả khó xác định hơn, vì cần chứng minh một biến "gây ra" thay đổi ở biến kia, không chỉ liên hệ. Phân tích hai biến chỉ gợi ý, không chứng minh nhân quả – cần thêm dữ liệu hoặc thí nghiệm.

**Các bước xác định**:

* **Bước 1: Kiểm tra tương quan trước**: Nếu không tương quan, không thể có nhân quả.
* **Bước 2: Kiểm soát biến nhiễu**:
  + Sử dụng **hồi quy tuyến tính** (regression): Xem biến X dự đoán Y tốt đến đâu (R-squared), kiểm tra hệ số hồi quy có ý nghĩa không.
  + **Kiểm định nhân quả cơ bản**: Granger causality test (cho dữ liệu thời gian) hoặc t-test để so sánh nhóm.
* **Bước 3: Thiết kế nghiên cứu**:
  + Thí nghiệm ngẫu nhiên (RCT): Phân nhóm ngẫu nhiên để kiểm soát nhiễu.
  + Phân tích đa biến: Bao gồm biến nhiễu để xem mối quan hệ có còn không.
  + Quy tắc Bradford Hill: Kiểm tra tính nhất quán, liều lượng, và thời gian (ví dụ: X xảy ra trước Y).
* **Diễn giải**: Nếu hồi quy cho thấy hệ số dương và ý nghĩa sau khi kiểm soát nhiễu, có thể suy luận nhân quả tạm thời. Ví dụ: Tuổi cao hơn dẫn đến thu nhập cao hơn? Có thể do kinh nghiệm tích lũy, nhưng cần kiểm soát giáo dục hoặc ngành nghề.

**Lưu ý**: Trong EDA, chỉ dừng ở gợi ý nhân quả; phân tích nhân quả cần dữ liệu lớn và mô hình chuyên sâu.

## + Sự khác biệt giữa tương quan (correlation) và hiệp biến (covariance) trong phân tích hai biến là gì?

Trong **phân tích hai biến** (Bivariate Analysis) của khám phá dữ liệu (EDA), **hiệp biến (covariance)** và **tương quan (correlation)** là hai thước đo thống kê phổ biến để khám phá mối liên hệ giữa hai biến định lượng. Cả hai đều giúp xác định xem hai biến có thay đổi cùng hướng (dương), ngược hướng (âm), hay không liên hệ (gần 0). Tuy nhiên, chúng khác nhau về cách tính, diễn giải và ứng dụng. Hiệp biến là nền tảng cho tương quan, nhưng tương quan được "chuẩn hóa" để dễ sử dụng hơn.

### 1. Hiệp biến (Covariance)

* **Định nghĩa**: Hiệp biến đo lường mức độ và hướng mà hai biến thay đổi cùng nhau so với giá trị trung bình của chúng. Nếu một biến tăng khi biến kia tăng, hiệp biến dương; ngược lại là âm.
* **Công thức cơ bản**:

trong đó Xˉ và Yˉ là trung bình của X và Y.

* **Đặc điểm**:
  + Giá trị có thể là bất kỳ số thực nào (dương, âm, 0).
  + Phụ thuộc vào đơn vị đo (ví dụ: nếu đo thu nhập bằng VND thay vì USD, giá trị thay đổi).
  + Không dễ diễn giải mức độ mạnh/yếu vì không giới hạn.
* **Ứng dụng**: Dùng trong xây dựng mô hình (như hồi quy) để xem hướng liên hệ, nhưng ít dùng trực tiếp do khó so sánh.

### 2. Tương quan (Correlation)

* **Định nghĩa**: Tương quan là hiệp biến được chuẩn hóa bằng độ lệch chuẩn của hai biến, thường là hệ số tương quan Pearson. Nó đo lường mức độ liên hệ tuyến tính giữa hai biến, với giá trị từ -1 (tương quan âm hoàn hảo) đến +1 (tương quan dương hoàn hảo).
* **Công thức cơ bản**:

trong đó σX và σY​ là độ lệch chuẩn.

* **Đặc điểm**:
  + Giá trị luôn nằm trong khoảng [-1, 1], dễ diễn giải (ví dụ: r=0.8 là mạnh dương).
  + Không phụ thuộc vào đơn vị đo, dễ so sánh giữa các cặp biến khác nhau.
  + Chỉ đo lường liên hệ tuyến tính; nếu mối quan hệ phi tuyến, cần Spearman hoặc Kendall.
* **Ứng dụng**: Phổ biến trong EDA để nhanh chóng xác định tương quan mạnh/yếu, hỗ trợ chọn biến cho mô hình.

### Bảng so sánh Tương quan và Hiệp biến

|  |  |  |
| --- | --- | --- |
| Tiêu chí | Hiệp biến (Covariance) | Tương quan (Correlation) |
| Mục tiêu | Đo hướng và mức độ thay đổi cùng nhau. | Đo mức độ liên hệ tuyến tính chuẩn hóa. |
| Phạm vi giá trị | Bất kỳ số thực (có thể rất lớn hoặc nhỏ). | [-1, 1] (0: không liên hệ, ±1: hoàn hảo). |
| Phụ thuộc đơn vị | Có (thay đổi nếu thay đơn vị đo). | Không (dễ so sánh giữa các biến). |
| Dễ diễn giải | Khó (không chuẩn hóa). | Dễ (mạnh) |
| Công thức | Dựa trên tích lệch trung bình. | Hiệp biến chia cho tích độ lệch chuẩn. |
| Ứng dụng chính | Xây dựng mô hình (hồi quy). | EDA, phát hiện tương quan nhanh. |
| Ví dụ | Cov(tuổi, thu nhập) = 50 (dương, cùng tăng). | r = 0.6 (tương quan trung bình dương). |

### Ví dụ minh họa

Giả sử dữ liệu tuổi (X) và thu nhập (Y, triệu VND) của 8 khách hàng:

* Tuổi: [25, 30, 35, 40, 45, 50, 55, 60]
* Thu nhập: [20, 25, 30, 35, 40, 45, 50, 55]
* Hiệp biến ≈ 50 (dương, cho thấy khi tuổi tăng, thu nhập tăng).
* Tương quan Pearson ≈ 1.0 (hoàn hảo dương, vì dữ liệu tuyến tính).

Nếu thay đơn vị thu nhập thành USD (giả sử 1 VND = 0.00004 USD), hiệp biến sẽ rất nhỏ (≈ 0.002), nhưng tương quan vẫn ≈ 1.0 – chứng tỏ tương quan không phụ thuộc đơn vị.

### Code mẫu bằng Python (tính toán đơn giản)

Sử dụng NumPy và SciPy để tính:

python

import numpy as np

from scipy.stats import pearsonr

*# Dữ liệu mẫu*

tuoi = np.array([25, 30, 35, 40, 45, 50, 55, 60])

thu\_nhap = np.array([20, 25, 30, 35, 40, 45, 50, 55])

*# Hiệp biến*

cov = np.cov(tuoi, thu\_nhap)[0, 1]

print(f"Hiệp biến: {cov:.2f}")

*# Tương quan*

corr, p\_value = pearsonr(tuoi, thu\_nhap)

print(f"Tương quan: {corr:.2f}, p-value: {p\_value:.2f}")

**Kết quả mẫu**: Hiệp biến: 50.00; Tương quan: 1.00, p-value: 0.00 (ý nghĩa thống kê cao).

### Kết luận

Hiệp biến cung cấp thông tin thô về hướng liên hệ nhưng khó diễn giải do không chuẩn hóa, trong khi tương quan là phiên bản "dễ dùng" hơn, lý tưởng cho EDA để nhanh chóng xác định mức độ liên hệ. Trong phân tích hai biến, hãy bắt đầu bằng tương quan để sàng lọc, sau đó dùng hiệp biến nếu cần chi tiết cho mô hình.

## + Khi nào nên sử dụng biểu đồ trực quan hóa trong phân tích đơn biến so với phân tích hai biến?

Trong khám phá dữ liệu (EDA), biểu đồ trực quan hóa là công cụ mạnh mẽ để làm rõ các đặc điểm dữ liệu, nhưng việc chọn loại biểu đồ phụ thuộc vào mục tiêu phân tích. **Phân tích đơn biến** tập trung vào một biến duy nhất để hiểu phân bố và đặc trưng nội tại, trong khi **phân tích hai biến** khám phá mối quan hệ giữa hai biến. Sử dụng biểu đồ đúng lúc giúp tránh nhầm lẫn và tăng hiệu quả phân tích. Dưới đây là hướng dẫn cụ thể về khi nào nên sử dụng từng loại.

### 1. Sử dụng biểu đồ trong phân tích đơn biến

Sử dụng khi bạn cần mô tả **một biến riêng lẻ**, như kiểm tra phân bố, phát hiện ngoại lai, hoặc tóm tắt tần suất. Đây là bước đầu tiên trong EDA, giúp làm sạch dữ liệu trước khi đi sâu hơn.

* **Khi nào nên sử dụng**:
  + Để xem **phân bố dữ liệu** (normal, skewed, multimodal) hoặc **độ phân tán** (range, IQR).
  + Với dữ liệu định lượng: Kiểm tra tính chuẩn hoặc ngoại lai.
  + Với dữ liệu phân loại: Hiển thị tần suất danh mục.
  + Khi dữ liệu lớn, khó đọc qua số liệu thô.
* **Biểu đồ phổ biến**:
  + **Histogram**: Cho biến định lượng liên tục (xem phân bố tần suất).
  + **Boxplot**: Tóm tắt phân bố, quartiles, và ngoại lai.
  + **Bar chart**: Cho biến phân loại (tần suất từng nhóm).
* **Ví dụ**: Phân tích tuổi khách hàng – dùng histogram để thấy phân bố tập trung quanh 30-40 tuổi.

### 2. Sử dụng biểu đồ trong phân tích hai biến

Sử dụng khi bạn cần khám phá **mối quan hệ giữa hai biến**, như tương quan, xu hướng, hoặc sự khác biệt giữa nhóm. Đây là bước sau đơn biến, giúp tạo giả thuyết cho mô hình hóa.

* **Khi nào nên sử dụng**:
  + Để kiểm tra **tương quan** (dương/âm, mạnh/yếu) hoặc **sự phụ thuộc**.
  + Với hai biến định lượng: Xem xu hướng tuyến tính/phi tuyến.
  + Với biến định lượng và phân loại: So sánh phân bố giữa nhóm.
  + Khi cần phát hiện cụm (clusters) hoặc ngoại lai đa biến.
* **Biểu đồ phổ biến**:
  + **Scatter plot**: Cho hai biến định lượng (xem tương quan).
  + **Boxplot nhóm**: So sánh phân bố của biến Y theo nhóm X (phân loại).
  + **Heatmap**: Hiển thị ma trận tương quan nếu có nhiều cặp biến.
* **Ví dụ**: Phân tích tuổi và thu nhập – dùng scatter plot để thấy thu nhập tăng theo tuổi.

## + Đoạn code mẫu để tạo biểu đồ scatter plot hoặc heatmap để phân tích mối quan hệ giữa hai biến?

Trong phân tích hai biến (Bivariate Analysis) của khám phá dữ liệu (EDA), scatter plot giúp trực quan hóa mối quan hệ giữa hai biến định lượng (như xu hướng tuyến tính hoặc cụm dữ liệu), trong khi heatmap (thường dùng cho ma trận tương quan) giúp hiển thị mức độ tương quan giữa nhiều cặp biến một cách tổng quan.

### 1. Scatter Plot (Biểu đồ phân tán)

Sử dụng để xem xu hướng giữa hai biến định lượng. Ví dụ: Mối quan hệ giữa tuổi và thu nhập.

python

import matplotlib.pyplot as plt

import numpy as np

*# Dữ liệu mẫu: 100 điểm dữ liệu với tương quan dương*

np.random.seed(42) *# Đảm bảo kết quả tái lập*

ages = np.random.normal(40, 10, 100) *# Tuổi*

incomes = 0.5 \* ages + np.random.normal(0, 5, 100) *# Thu nhập (tương quan với tuổi)*

*# Tạo scatter plot*

plt.figure(figsize=(8, 6)) *# Kích thước biểu đồ*

plt.scatter(ages, incomes, alpha=0.7, color='#ff7f0e', s=50) *# alpha: độ trong suốt, s: kích thước điểm*

plt.title('Scatter Plot: Mối Quan Hệ Giữa Tuổi và Thu Nhập', fontsize=14)

plt.xlabel('Tuổi', fontsize=12)

plt.ylabel('Thu Nhập (triệu VND)', fontsize=12)

plt.grid(True, alpha=0.3) *# Lưới nhẹ để dễ đọc*

plt.savefig('scatter\_plot.png', dpi=300, bbox\_inches='tight') *# Xuất file (tùy chọn)*

plt.show()

**Giải thích**:

* **Dữ liệu**: Tạo hai mảng ngẫu nhiên với tương quan dương (thu nhập tăng theo tuổi).
* **Tùy chỉnh**: alpha=0.7 làm điểm trong suốt để tránh chồng chéo; s=50 điều chỉnh kích thước điểm.
* **Ứng dụng**: Nếu các điểm tạo đường thẳng dương, tương quan mạnh; nếu phân tán ngẫu nhiên, không liên hệ.
* **Kết quả**: Biểu đồ cho thấy xu hướng tăng dần, gợi ý tương quan dương.

### 2. Heatmap (Bản đồ nhiệt – cho ma trận tương quan)

Sử dụng để hiển thị tương quan giữa hai (hoặc nhiều) biến dưới dạng màu sắc. Ví dụ: Ma trận tương quan giữa tuổi và thu nhập.

python

import seaborn as sns

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

*# Dữ liệu mẫu (tương tự scatter plot)*

np.random.seed(42)

ages = np.random.normal(40, 10, 100)

incomes = 0.5 \* ages + np.random.normal(0, 5, 100)

df = pd.DataFrame({'Tuoi': ages, 'Thu\_Nhap': incomes}) *# Chuyển thành DataFrame*

*# Tính ma trận tương quan*

corr = df.corr() *# Hệ số Pearson*

*# Tạo heatmap*

plt.figure(figsize=(6, 4))

sns.heatmap(corr, annot=True, cmap='coolwarm', center=0, square=True, linewidths=0.5)

plt.title('Heatmap: Ma Trận Tương Quan Giữa Các Biến', fontsize=14)

plt.savefig('heatmap\_corr.png', dpi=300, bbox\_inches='tight') *# Xuất file (tùy chọn)*

plt.show()

**Giải thích**:

* **Dữ liệu**: Sử dụng Pandas để tính corr() (ma trận tương quan Pearson).
* **Tùy chỉnh**: annot=True hiển thị giá trị số; cmap='coolwarm' dùng màu xanh-đỏ (âm-dương); center=0 cân bằng màu quanh 0.
* **Ứng dụng**: Màu đỏ cho tương quan dương mạnh (ví dụ: 0.64 giữa tuổi và thu nhập); xanh cho âm. Lý tưởng khi có nhiều biến.
* **Kết quả**: Ô tương quan giữa "Tuoi" và "Thu\_Nhap" ≈ 0.64 (trung bình dương), xác nhận mối quan hệ.

### Lưu ý khi sử dụng

* **Cài đặt thư viện**: Chạy pip install matplotlib seaborn pandas numpy nếu chưa có.
* **Mở rộng**: Với dữ liệu thực (CSV), dùng pd.read\_csv('file.csv') thay dữ liệu mẫu. Thêm plt.tight\_layout() để điều chỉnh bố cục.
* **Tuân thủ nguyên tắc thiết kế**: Giữ biểu đồ đơn giản, màu tương phản, nhãn rõ ràng để dễ hiểu mối quan hệ.
* **Khi nào dùng**: Scatter plot cho khám phá trực quan nhanh; heatmap cho tổng quan tương quan khi có >2 biến.

## + Làm thế nào để trực quan hóa mối quan hệ giữa một biến số và một biến phân loại bằng biểu đồ boxplot hoặc violin plot trong Python?

Trong **phân tích hai biến** (bivariate analysis) của khám phá dữ liệu (EDA), khi bạn có một **biến số (numerical variable)** (như thu nhập, tuổi) và một **biến phân loại (categorical variable)** (như khu vực, giới tính), **boxplot** và **violin plot** là hai biểu đồ phổ biến để so sánh phân bố của biến số theo từng nhóm phân loại.

* **Boxplot**: Tóm tắt phân bố qua trung vị, tứ phân vị (quartiles), và ngoại lai (outliers). Phù hợp khi cần xem nhanh sự khác biệt giữa các nhóm (ví dụ: thu nhập trung bình theo khu vực).
* **Violin plot**: Kết hợp boxplot với histogram xoay 90 độ, hiển thị mật độ phân bố chi tiết hơn (density). Phù hợp khi cần xem hình dạng phân bố (skewed, multimodal) theo nhóm.

Cả hai đều dễ tạo bằng **Seaborn** (dựa trên Matplotlib), tích hợp tốt với Pandas. Dưới đây là hướng dẫn và code mẫu sử dụng dữ liệu giả định: thu nhập (numerical) theo khu vực (categorical: Hà Nội, TP.HCM, Đà Nẵng).

**Yêu cầu cài đặt**

Chạy lệnh sau nếu chưa có: pip install matplotlib seaborn pandas numpy.

### 1. Boxplot (Sử dụng Seaborn)

Boxplot giúp so sánh trung vị và độ phân tán giữa các nhóm.

python

import seaborn as sns

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

*# Dữ liệu mẫu: Thu nhập theo khu vực*

np.random.seed(42)

data = {

'Khu\_Vuc': np.random.choice(['Ha\_Noi', 'TP\_HCM', 'Da\_Nang'], 150),

'Thu\_Nhap': np.concatenate([

np.random.normal(50, 10, 50), *# Hà Nội: mean=50 triệu*

np.random.normal(70, 15, 50), *# TP.HCM: mean=70 triệu*

np.random.normal(40, 8, 50) *# Đà Nẵng: mean=40 triệu*

])

}

df = pd.DataFrame(data)

*# Tạo boxplot*

plt.figure(figsize=(8, 6))

sns.boxplot(data=df, x='Khu\_Vuc', y='Thu\_Nhap', palette='Set2')

plt.title('Boxplot: Thu Nhập Theo Khu Vực', fontsize=14)

plt.xlabel('Khu Vực', fontsize=12)

plt.ylabel('Thu Nhập (triệu VND)', fontsize=12)

plt.grid(True, alpha=0.3)

plt.savefig('boxplot\_thu\_nhap.png', dpi=300, bbox\_inches='tight') *# Xuất file (tùy chọn)*

plt.show()

**Giải thích**:

* **Dữ liệu**: Tạo DataFrame với 150 mẫu, thu nhập phân bố khác nhau theo khu vực.
* **Seaborn boxplot**: x='Khu\_Vuc' (phân loại), y='Thu\_Nhap' (số). palette='Set2' cho màu sắc.
* **Kết quả**: Các hộp cho thấy TP.HCM có trung vị thu nhập cao nhất, với ngoại lai ở đuôi.
* **Khi nào dùng**: Để phát hiện sự khác biệt nhanh giữa nhóm, đặc biệt với ngoại lai.

### 2. Violin Plot (Sử dụng Seaborn)

Violin plot hiển thị mật độ phân bố, giúp thấy hình dạng (ví dụ: lệch phải) ngoài boxplot cơ bản.

python

import seaborn as sns

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

*# Dữ liệu mẫu (giống boxplot)*

np.random.seed(42)

data = {

'Khu\_Vuc': np.random.choice(['Ha\_Noi', 'TP\_HCM', 'Da\_Nang'], 150),

'Thu\_Nhap': np.concatenate([

np.random.normal(50, 10, 50),

np.random.normal(70, 15, 50),

np.random.normal(40, 8, 50)

])

}

df = pd.DataFrame(data)

*# Tạo violin plot*

plt.figure(figsize=(8, 6))

sns.violinplot(data=df, x='Khu\_Vuc', y='Thu\_Nhap', palette='Set2', inner='box') *# inner='box' thêm boxplot bên trong*

plt.title('Violin Plot: Thu Nhập Theo Khu Vực', fontsize=14)

plt.xlabel('Khu Vực', fontsize=12)

plt.ylabel('Thu Nhập (triệu VND)', fontsize=12)

plt.grid(True, alpha=0.3)

plt.savefig('violin\_plot\_thu\_nhap.png', dpi=300, bbox\_inches='tight') *# Xuất file (tùy chọn)*

plt.show()

### Giải thích:

* **Dữ liệu**: Giống boxplot.
* **Seaborn violinplot**: inner='box' thêm hộp tóm tắt bên trong violin (tùy chọn). palette='Set2' cho màu.
* **Kết quả**: Violin rộng hơn ở TP.HCM cho thấy phân bố rộng, mật độ cao quanh 70 triệu.
* **Khi nào dùng**: Khi cần chi tiết phân bố (mật độ, đa đỉnh), đặc biệt với dữ liệu lớn.

**Lưu ý khi sử dụng**

* **Dữ liệu thực tế**: Thay dữ liệu mẫu bằng df = pd.read\_csv('your\_file.csv') và điều chỉnh cột.
* **Tùy chỉnh**: Thêm sns.swarmplot để vẽ điểm dữ liệu nếu số lượng nhỏ; dùng plt.xticks(rotation=45) nếu nhãn dài.
* **So sánh**: Boxplot đơn giản hơn cho báo cáo; violin plot chi tiết hơn cho EDA sâu.
* **Nguyên tắc thiết kế**: Giữ màu tương phản, nhãn rõ ràng, và tránh quá tải thông tin để dễ đọc sự khác biệt giữa nhóm.
* **Thư viện thay thế**: Nếu dùng Matplotlib thuần, có thể dùng plt.boxplot nhưng Seaborn dễ hơn cho dữ liệu nhóm.