NHẬP MÔN THỐNG KÊ HƯỚNG TỚI MÁY HỌC

Phạm Minh Hoàng

Diễn đàn VBA Việt Nam

Website: tuhocvba.net

Ngày 17 tháng 6 năm 2022

LỜI GIỚI THIỆU

Nói tới diễn đàn THVBA, người ta nghĩ tới những bài viết dễ hiểu, vì phương châm của chúng tôi là viết làm sao để người ngu nhất cũng hiểu được. Triết lý này là nền móng căn cơ phát triển diễn đàn cho tới nay. THVBA chuyên về lĩnh vực VBA và đã đóng góp rất nhiều bài viết về VBA được cộng đồng đánh giá rất cao. Nhiều người khi muốn tìm hiểu một lĩnh vực nào đó thường nói đùa rằng, ước gì anh em quản trị viên THVBA cũng mở thêm mảng mà họ quan tâm, họ mong chờ những bài viết với phong cách dễ hiểu.

Lập trình AI hay Khoa học dữ liệu là lĩnh vực khó mặc dù Python đã hỗ trợ chúng ta rất nhiều. Nếu chỉ vận dụng code có sẵn và làm theo thì chẳng có gì để nói, vấn đề ở đây là làm sao hiểu được bản chất để từ đó làm chủ tri thức. Kiến thức toán thống kê được sử dụng nhiều, nhưng phần lớn mọi người giảng cho nhau nghe thì chỉ nói tới khái niệm chung chung, không minh họa bằng những ví dụ cụ thể để mọi người hình dung đúng bản chất. Một số khác mặc nhiên công nhận điều người khác nói mà không hiểu rõ ràng những khái niệm thống kê này.

Theo tôi biết, sách về toán thống kê ở Việt Nam không thiếu, nhưng liên kết nó tới các vấn đề về AI, về khoa học dữ liệu, thì chưa có cuốn sách giáo khoa nào làm tốt. Trước nhu cầu cấp bách đó, tôi ấp ủ tạo nên cuốn sách này, với lối trình bày dễ hiểu, tôi sẽ làm rõ những khái niệm cơ bản về xác suất, đồng thời sử dụng Python làm công cụ kiểm chứng các kết quả tính toán thống kê. Tôi không mong muốn gì hơn đó là cuốn sách thực sự có ích cho các bạn. Và nếu trong tương lai nó trở thành cuốn sách gối đầu giường của các bạn sinh viên theo học ngành khoa học dữ liệu, hay máy học, thì đó là niềm vui đối với tôi.

Nội dung trong cuốn sách này được tôi biên dịch từ Chúc các bạn gặt hái nhiều thành công!

Admin Forum THVBA
nickname: tuhocvba
Phạm Minh Hoàng
Tốt nghiệp ĐH Bách Khoa Hà Nội khóa 2003-2008
Cựu học sinh chuyên toán Chuyên Hùng Vương Phú Thọ khóa 2000-2003

Mục lục

Bài 1

Thống kê mô tả và Suy luận thống kê

1.1 Phạm vi khóa học

Ở khóa học này chúng ta sẽ học các nội dung cơ bản về thống kê học.
Những điều cơ bản của những điều cơ bản. Tôi chưa bao giờ nghiên cứu thống kê! !! Tôi muốn bạn thực hiện khóa học này trước khi đọc một cuốn sách khó về thống kê.
Phạm vi của khóa học này sẽ bắt đầu đi từ thống kê mô tả và xem nhẹ ước tính và thử nghiệm. Sau đó, tôi ước mình có thể kết nối với một khóa học máy học.
Tuy nhiên, tôi nghĩ rằng bạn sẽ có thể đọc các sách thống kê khác nếu bạn học qua khóa học này vì bạn sẽ có thể học một cách vững chắc các ý tưởng cơ bản về thống kê. Đó sẽ là kiến thức cần thiết để nghiên cứu học máy, vì vậy hãy củng cố những kiến thức cơ bản trong khóa học này!

1.2 Sử dụng Python trong khóa học này

Trong khóa học này tôi sẽ sử dụng Python để tiếp cận với thống kê, tuy nhiên dù cho bạn không hiểu gì về code python thì bạn có thể bỏ qua các phần chứa code python.
Tuy nhiên, do mục đích nhắm tới sau này là khoa học dữ liệu (Data Science), do đó dù thế nào đi nữa, nếu bạn có chút kiến thức cơ bản về Python thì vẫn tốt hơn đấy.
Ở khóa học này, các thư viện trong python được sử dụng chủ yếu là NumPy, Pandas, matplotlib, seaborn, và tôi muốn giới thiệu tới hai thư viện mới là SciPy (stats) và scikit-learn.

SciPy là thư viện mở của python trong khoa học, đọc là sai-pai. Nó dựa trên **NumPy**

và có các mô-đun rất hữu ích trong khoa học và kỹ thuật, chẳng hạn như các bài toán thống kê và tối ưu hóa, tích phân và đại số tuyến tính.
Trong khóa học này, chúng ta sẽ sử dụng module stats có trong SciPy để nghiên cứu về thống kê. **scikit-learn** là thư viện mở của python ứng dụng trong máy học. Nó có mặt ở trong hầu hết các thuật toán chính về máy học.
Vì scikit xuất phát từ SciPy Toolkit , nó là một thư viện mở rộng của SciPy, nhưng từ quan điểm của người dùng, bạn có thể coi nó như một thư viện riêng biệt từ SciPy.
scikit-learn là thư viện ứng dụng trong máy học nhưng cũng được sử dụng một chút trong thống kê. Cả SciPy và scikit-learn đều có trong Anaconda.
Trong khóa học này, chúng tôi thực sự sẽ viết code python, nhưng xin lưu ý rằng code trong khóa học này không nhất thiết phải là "tối ưu". (Bởi vì nó không phải là một khóa học “Python”).

1.3 Thống kê là gì?

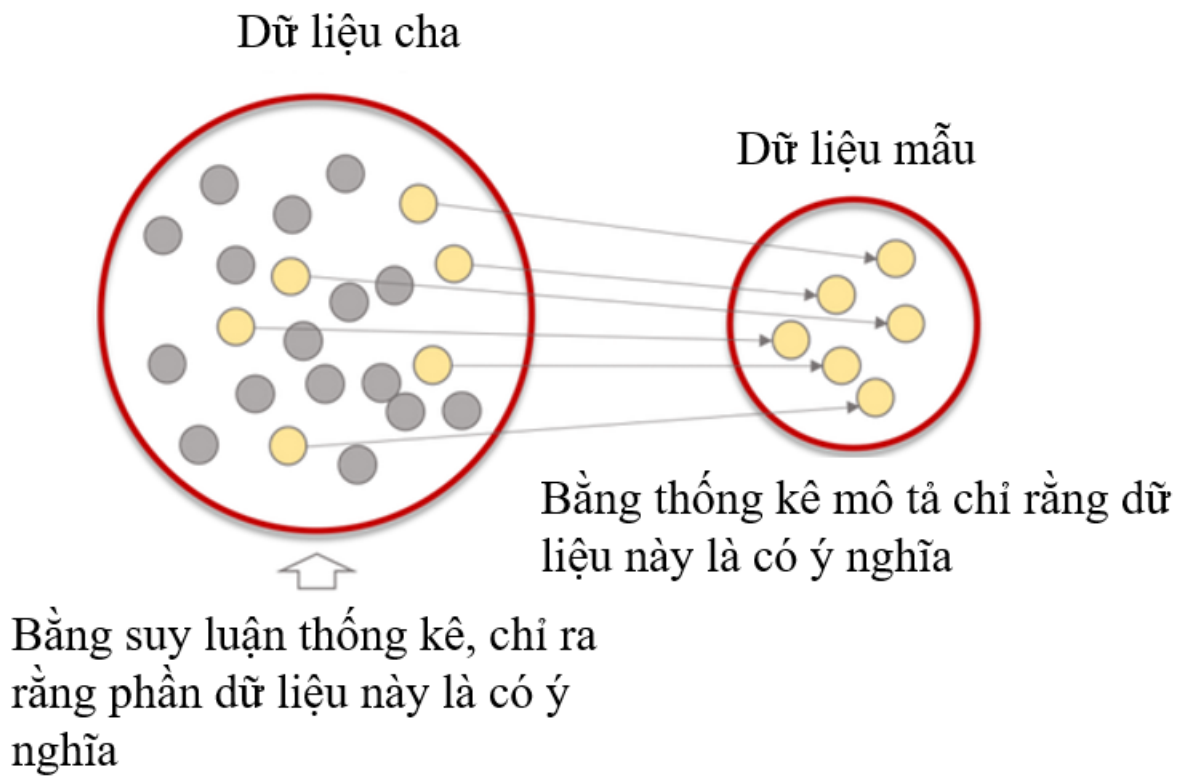
Nào, hãy bắt đầu với câu hỏi, thống kê là gì? Thống kê là bộ môn khoa học nghiên cứu các phương pháp, cách nghĩ về phân tích dữ liệu thống kê. Dữ liệu thống kê là: Khi chúng ta có một vật mà chúng ta muốn quan sát, chúng ta tập hợp các giá trị quan sát được, các giá trị đo được từ đối tượng đó.
Chẳng hạn, nếu chúng ta muốn biết thu nhập năm của một người làm khoa học dữ liệu ở Nhật Bản, thực tế, trước hết chúng ta phải tìm kiếm dữ liệu có tên như thế nào đó liên quan tới thu nhập hàng năm của người làm khoa học dữ liệu. Giá trị mà chúng ta thu thập, tập hợp lại, đó gọi là dữ liệu thống kê.
Trong khóa học này, tôi chia thống kê làm hai nhóm lớn:

- Thống kê mô tả (descriptive statistics).
- Thống kê luận lý (thống kê suy luận) (inferential statistics).

Nghe qua những từ ngữ trên bạn đừng nghĩ là khó nhé, nó không khó đâu.
Tầm quan trọng của việc trực quan hóa dữ liệu bằng cách sử dụng các biểu đồ phân tán cũng đã được thực hiện trong khóa học Python.
Tuy nhiên, thực tế nếu chỉ có biểu đồ phân tán, ta có thể hiểu khuynh hướng của dữ liệu, nhưng nếu chỉ có thế thì nó vẫn có giới hạn, do đó tôi muốn sử dụng kiến thức số học cơ bản để nói dữ liệu này là $\cdots$
Ví dụ, khi nhìn vào thu nhập năm của những người bạn làm về khoa học dữ liệu tại Nhật Bản, có cảm nhận rằng thu nhập này cao hơn người bình thường, nhưng cao hơn bao nhiêu, thì cần có các chỉ số dễ hiểu.
Chẳng hạn nhìn vào mức thu nhập bình quân (trung bình) cũng được, hoặc có thể sử dụng giá trị trung tâm.
Nếu sử dụng các chỉ số như thế, thì dù dữ liệu nhìn thấy là ít (không cần nhiều người) đi chăng nữa, ta vẫn có thể phân tích đúng không nào.

À, bạn có thể nghĩ ồ việc như thế cũng có thể làm được hay sao, vậy thì phải cố học thống kê thôi!

Từ những dữ liệu có ý nghĩa mà bạn thu thập trong thực tế, gọi là dữ liệu mẹ(cha) . Tiếng anh là population, từ dữ liệu quan sát thực tế này ta lấy ra một phần dữ liệu để nghiên cứu, phần dữ liệu này gọi là dữ liệu mẫu, hay dữ liệu tiêu bản, tiếng anh là sample. Suy luận thống kê là sử dụng dữ liệu mẫu để từ đó suy luận ra đặc tính của dữ liệu cha.



Suy luận thống kê có hai kiểu là Ước lượng và Kiểm định. Ước lượng là suy ra tỷ lệ trong dữ liệu cha hoặc trung bình từ một mẫu dữ liệu. (Ví dụ: "Thu nhập trung bình hàng năm của các nhà khoa học dữ liệu Nhật Bản chắc chắn là XX!" Hoặc "Tỷ lệ nam-nữ của các nhà khoa học dữ liệu Nhật Bản chắc chắn là XX!")

Kiểm định là đưa ra câu trả lời Có / Không cho bất kỳ "câu hỏi" nào dựa trên kết quả khảo sát của mẫu. (Ví dụ: trả lời Có / Không cho "Thu nhập hàng năm của các nhà khoa học dữ liệu Nhật Bản có tăng so với năm ngoái không?")

Nói chung, hầu hết mọi người nghĩ về "thống kê" là thống kê suy luận. Trên thực tế, thống kê ước lượng là phần chính của thống kê. Tuy nhiên, kiến thức về thống kê mô tả là không thể thiếu để ước lượng và kiểm định, vì vậy trong khóa học này, chúng ta sẽ nghiên cứu về thống kê mô tả trước khi đi vào thống kê suy luận.

Ồ từ từ đã tôi đã nhớ hết đâu!!! Bạn không cần phải biết! Bạn không cần phải nhớ bất kỳ từ nào! Hãy làm quen dần dần nhé!
Sau đó, từ lần sau, tôi sẽ giải thích "giá trị đại diện" đầu tiên của thống kê mô tả!

Bài 2

Giá trị đại diện

Ở bài học trước chúng ta đã chia ra gồm có thống kê mô tả và thống kê suy luận. Thống kê mô tả là lĩnh vực phân tích dữ liệu thống kê từ những dữ liệu quan sát được. Tuy nhiên để hiểu được kết quả phân tích dữ liệu, ta cần có chỉ số thể hiện đặc tính của dữ liệu. Ở bài học này sẽ giới thiệu các chỉ số nói lên đặc tính của dữ liệu.
Cũng từ đây, chúng ta sẽ sử dụng Python để tính toán chỉ số của dữ liệu.

2.1 Giá trị đại diện của dữ liệu

Nó có một cái tên khá khó hiểu, nhưng mấu chốt là giá trị được sử dụng để giải thích ý nghĩa của dữ liệu.
Ví dụ, giả sử bạn mua 5 quả táo ở siêu thị. Khi được hỏi một quả táo nặng bao nhiêu, bạn thường trả lời là trọng lượng trung bình của năm quả táo phải không? "Giá trị trung bình" được sử dụng ở đây là giá trị đại diện thể hiện trọng lượng của quả táo đã mua. Nó chỉ được đại diện bởi một con số và được sử dụng để đánh giá tài sản tổng thể.
Bạn vô tình sử dụng "giá trị đại diện" này trong cuộc sống hàng ngày của mình để xác định xem tổng thể những quả táo bạn mua hôm nay có to không, đúng không? Lần này, chúng ta sẽ xem xét kỹ hơn "giá trị trung bình" này.

2.2 Trung bình số học thường được biết đến là gì (trung bình cộng)

Ồ, nói như vậy chắc là có lắm kiểu giá trị trung bình lắm đúng không? Đúng vậy. Và giá trị trung bình cộng (arithmetic mean) chỉ là một trong số nhiều giá trị trung bình. Ví dụ: Tôi có 5 quả táo có trọng lượng như sau $295g$, $300g$, $300g$, $310g$, $311g$. Chúng có khối lượng trung bình là:

$$\frac{(295g + 300g + 300g + 310g + 311g)}{5} = 303.2g$$

Trong python, ta có thể dùng thư viện numpy để tính toán giá trị trung bình cộng như sau:

```
import numpy as np
apple_weights = [295, 300, 300, 310, 311]
np.mean(apple_weights)
```

Kết quả ta được:

```
303.2
```

Giá trị trung bình cộng của một dãy số là tổng các số của dãy số chia cho số số hạng. Nếu bạn viết điều này trong một công thức toán học, nó sẽ như sau:

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

2.3 Sử dụng tỷ suất (trung bình hình học)

Chúng ta hãy xem xét ví dụ sau:
Người ta thống kê được rằng nhân viên một công ty từ khi gia nhập vào một công ty có độ thăng tiến về thu nhập như sau:

-Sau một năm làm việc thì lương tăng so với năm trước là 5%.

-Sau hai năm làm việc thì lương tăng so với năm trước là 10%.

-Sau ba năm làm việc thì lương tăng so với năm trước là 30%.

Giả sử khi mới gia nhập một công ty, lương anh nhân viên là 500 đô la. Vậy bây giờ (sau 3 năm) thì lương của anh ấy là bao nhiêu?

$$500\,(1 + 0.05)\,(1 + 0.1)\,(1 + 0.3) = 750.75$$

Như vậy mức tăng trung bình hàng năm là bao nhiêu % là câu hỏi chúng ta sẽ nghĩ tới. Nếu tính bằng công thức tính giá trị trung bình 5%, 10%, 30% thì chẳng phải mức tăng lương bình quân hàng năm là: $\frac{5\% + 10\% + 30\%}{3} = 15\%$ hay sao?
Nào, bây giờ tôi giả định tỷ suất tăng lương hàng năm là g. Khi đó ta có:

$$500\,(1 + g)\,(1 + g)\,(1 + g) = 500\,(1 + g)^3 = 750.75$$

Nếu ta coi $m_g = (1 + g)$, ta có: $x_1 = (1 + 0.05)$,$x_2 = (1 + 0.01)$,$x_3 = (1 + 0.3)$.
Ta có công thức: ${m_g}^3 = x_1 x_2 x_3 \Rightarrow m_g = \sqrt[3]{x_1 x_2 x_3}$
Giá trị này gọi là trung bình hình học (geometric mean). Nó được sử dụng khi tính tỷ suất trung bình.
Như vậy nếu là trung bình cộng thì ta có công thức: $\frac{x_1 + x_2 + x_3}{3}$
Trong trường hợp tính tỷ suất trung bình thì là: $\sqrt[3]{x_1 x_2 x_3}$
Tổng quát ta có:

$$m_g = \sqrt[n]{x_1 x_2 x_3 \cdots x_n}$$

Ta hãy thử tính giá trị trung bình hình học bằng Python. Ở đây tôi sẽ sử dụng module stats trong SciPy.
Các bạn có thể sử dụng `scipy.stats.gmean()` để tính.

```
from scipy import stats
salary_growth = [1.05, 1.1, 1.3]
salary_growth_mean = stats.gmean(salary_growth)
print(salary_growth_mean)
```

Kết quả:

```
1.1450956868476592
```

Như vậy từ nay ai đó hỏi bạn tỷ suất tăng lương trung bình hàng năm, các bạn hãy sử dụng công thức tính giá trị trung bình hình học.

2.4 Trung bình điều hòa

Ở phần này tôi xin giới thiệu một khái niệm trung bình nữa, đó là trung bình điều hòa, có tên tiếng anh là harmonic mean.
Nó còn có tên gọi khác là nghịch đảo của trung bình nghịch đảo. Nghe tên thì có cảm giác khó hiểu nên sau đây tôi đi vào ví dụ.
Chẳng hạn một người đi từ A tới B với vận tốc là $x_1$km/h.Ở chiều về anh ta đi từ B tới A với vận tốc là $x_2$km/h.
Như vậy vận tốc trung bình là bao nhiêu? Nhiều người sẽ nghĩ rằng đó chẳng phải là $\frac{x_1 + x_2}{2}$km/h hay sao? Tuy nhiên chúng ta hãy xem xét vấn đề dưới góc độ thời gian như sau:
Thời gian anh ta đi quãng đường $2.AB$ là bao lâu nếu coi d (km) là khoảng cách AB, ta có:$\frac{d}{x_1} + \frac{d}{x_2}$ (đơn vị thời gian).
Như vậy vận tốc trung bình mà anh ta đã đi là quãng đường chia cho thời gian:

$$\frac{2d}{\frac{d}{x_1} + \frac{d}{x_2}} = \frac{1}{\frac{1}{2}\left(\frac{1}{x_1} + \frac{1}{x_2}\right)}\text{km/h}$$

Nhìn vào công thức này ta thấy nó có hình thù là nghịch đảo của trung bình cộng của nghịch đảo.
Chẳng hạn chiều đi anh ta đi với vận tốc 20km/h và chiều về đi với vận tốc 60km/h, như thế nếu tính công thức trung bình cộng sẽ là $(20 + 60)/2 = 40$km/h, tuy nhiên không phải vậy, nếu tính bằng công thức nghịch đảo như đã nói ở trên, nó sẽ là nghịch đảo của $(1/20 + 1/60)/2 = 1/30$, tức là vận tốc trung bình là 30km/h.
Một cách tổng quát, nếu ta có n số hạng $x_1, x_2 . \cdots, x_n$, khi đó trung bình điều hòa là:

$$m_h = \frac{1}{\frac{1}{n}\left(\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_n}\right)}$$

Trong Python ta có cách tính trung bình điều hòa bằng cách sử dụng scipy.stats.hmean() như sau:

```
velocities = [20, 60]
velocities_mean = stats.hmean(velocities)
print(velocities_mean)
```

Kết quả:

```
30.0
```

Thành thật mà nói, tôi không nghĩ có nhiều trường hợp nó được sử dụng như giá trị đại diện, nhưng nó xuất hiện vài lần khi học về lý thuyết máy học, do đó tôi đã giới thiệu cho các bạn công thức này.

2.5 Tính chất quan trọng của giá trị trung bình cộng

Đối với giá trị trung bình cộng, nó có rất nhiều tính chất nhưng ở đây tôi giới thiệu hai tính chất quan trọng.

Đầu tiên phải nói tới tính chất, **tổng độ lệch (sai khác) từ các điểm dữ liệu tới giá trị trung bình cộng là** 0.

Ví dụ với 5 quả táo lần lượt có trọng lượng là 295g, 300g, 300g, 310g, 311g có giá trị trung bình là 303.2g. Lần lượt lấy trọng lượng từng quả táo trừ cho giá trị trung bình cộng rồi cộng các kết quả này với nhau ta sẽ được tổng là 0.

```
apple_weights = np.array([295, 300, 300, 310, 311])
apple_w_mean = np.mean(apple_weights)
deviations = apple_weights - apple_w_mean
print(deviations)
print(deviations.sum())
```

Kết quả:

```
[-8.2 -3.2 -3.2  6.8  7.8]
5.684341886080802e-14
```

$e-14$ có nghĩa là 10^{-14},nó có giá trị xấp xỉ là 0.

Tóm lại, giá trị trung bình có nghĩa là nó ở vị trí chính giữa trong toàn bộ dữ liệu, do đó độ lệch từ các điểm dữ liệu tới nó sẽ có giá trị dương và âm, chúng triệt tiêu lẫn nhau và dẫn tới tổng độ sai khác này có kết quả là 0 .

Độ sai khác tới giá trị trung bình được gọi là Thiên Sai (deviation). Thiên trong từ Thiên Vị-Thiên Kiến ý nói thiên lệch về một phía, Sai trong từ Sai Khác. Tôi nghĩ một từ thuần việt là **độ lệch** có lẽ dễ hiểu hơn.

Một tính chất quan trọng khác đó là **tổng các bình phương của độ lệch từ các điểm dữ liệu tới giá trị trung bình là giá trị nhỏ nhất**. Từ một giá trị X bất kỳ, ta có tổng các bình phương của độ sai khác từ các dữ liệu tới giá trị X là $S(X) = \sum_{i=1}^{n}(x_i - X)^2$.

Vậy X nhận giá trị là bao nhiêu thì tổng trên nhận giá trị nhỏ nhất? Trước hết ta hãy tính đạo hàm của biểu thức trên và tìm hiểu xem nó nhận giá trị 0 khi nào:

$$\frac{dS(X)}{dX} = -2\sum_{i=1}^{n}(x_i - X) = 0$$

Tiếp tục biến đổi:

$$0 = \sum_{i=1}^{n} (x_i - X) = (x_1 + x_2 + \cdots + x_n) - nX = n\bar{x} - nX$$

Biến đổi tới đây tôi nghĩ nhiều người đã hiểu. Tuy nhiên nếu tới đây mà vẫn có người không hiểu thì cũng không sao đâu. Tôi nghĩ bạn có thể công nhận điều này mà không cần phải chứng minh.

Những tính chất trên có vẻ như là hiển nhiên nhưng chúng lại rất quan trọng khi sử dụng nó để giải thích lý thuyết thống kê trong tương lai.

2.6 Tổng kết

Trong bài học này, để giải thích dữ liệu bằng các chỉ số dễ hiểu, tôi đã giới thiệu các giá trị đại diện được sử dụng để giải thích dữ liệu, đặc biệt là giá trị trung bình được sử dụng nhiều nhất.
Thông thường, giá trị trung bình được đề cập thường được hiểu đó là giá trị trung bình cộng trừ khi có ghi chú khác. Trong khuôn khổ khóa học này cũng vậy, thuật ngữ "trung bình" đề cập đến đó là trung bình số học-hay còn gọi là trung bình cộng.

- Giá trị đại diện là giá trị đại diện cho dữ liệu. Chỉ một giá trị được sử dụng để đánh giá bản chất của toàn bộ dữ liệu.
- Trung bình là một trong những chỉ số thường được sử dụng làm giá trị đại diện.
- Chú ý rằng giá trị trung bình hình học được sử dụng trong trường hợp giá trị dữ liệu là tỷ suất.
- Để tính giá trị trung bình hình học ta dùng `scipy.stats.gmean()`, và để tính giá trị trung bình điều hòa ta sử dụng `scipy.stats.hmean()` .
- Tổng độ lệch từ các điểm dữ liệu tới giá trị trung bình cộng là 0.
- Giá trị trung bình cộng là giá trị mà tại đó tổng các bình phương của độ lệch từ các điểm dữ liệu tới nó là nhỏ nhất.

Chỉ toàn chữ và công thức, quả thật là khó nhớ, vì vậy hãy cố gắng học nó. Lần tới, tôi sẽ giới thiệu những giá trị đại diện khác.

Bài 3

Giá trị đại diện khác

Tôi đã đề cập đến chỉ số quan trọng nhất, "trung bình", trong "giá trị đại diện", là "giá trị giải thích các đặc điểm của toàn bộ dữ liệu", nhưng vì có các giá trị khác có thể được sử dụng làm giá trị đại diện, tôi sẽ giới thiệu ngắn gọn về chúng.

3.1 Giá trị trung vị, giá trị giữa

Đây là giá trị xuất hiện nhiều lần khi các bạn học Data Science-Khoa học dữ liệu (Trong NumPy hay boxplot của Seaborn). Tôi không nghĩ mình cần phải giải thích về giá trị trung vị **(median)**, hay còn gọi là giá trị giữa, vì nó là một chỉ số thường đường dùng thường xuyên. Khi chúng ta sắp xếp dữ liệu từ bé đến lớn, giá trị này nằm ở chính giữa. (Nếu có một số lượng dữ liệu chẵn thì người ta thường lấy trung bình cộng của cả hai số ở giữa làm giá trị trung vị).

```
import numpy as np
arr = np.array([7, 2, 4])
x = np.median(arr)
y = np.mean(arr)
print(f"Gia tri trung vi la {x}")
print(f"Gia tri trung binh la {y}")
```

Kết quả:

```
Gia tri trung vi la 4.0
Gia tri trung binh la 4.333333333333333
```

Để làm rõ trong trường hợp dữ liệu có số chẵn phần tử:

```
import numpy as np
arr = np.array([7, 2, 4,6])
x = np.median(arr)
print(f"Gia tri trung vi la {x}")
```

Kết quả:

```
Gia tri trung vi la 5.0
```

Như trong ví dụ ở phần trước khi đề cập tới các trái táo, nếu các giá trị dữ liệu gần giống nhau, bạn có thể sử dụng giá trị trung bình làm giá trị đại diện, tuy nhiên có những giá trị ngoại lệ (giá trị lệch khỏi phân phối tổng thể)**(outlier)**, nếu sử dụng giá trị trung

bình làm giá trị đại diện, việc hiểu sai về bản chất dữ liệu có thể xảy ra. Ví dụ khi nói tới thu nhập hàng năm hoặc về tài sản, tiền tệ, có lẽ giá trị trung vị sẽ tốt hơn là giá trị trung bình khi ta phân tích các dữ liệu như vậy. Dù có thể có những giá trị nằm ra ngoài phân phối tổng thể thì giá trị trung vị hầu như không bị ảnh hưởng trong khi đó nếu như có một (hoặc nhiều) giá trị đột biến lớn hay đột biến nhỏ nằm ngoài phân phối tổng thể của dữ liệu sẽ có tác động rất lớn tới giá trị trung bình cộng. Ngoài ra cần lưu ý thêm, lượng phép tính khi tính toán trung vị sẽ tốn nhiều hơn so với tính toán giá trị trung bình cộng. Giá trị trung bình không quá tốn kém khi nói tới chi phí tính toán, ta có thể cộng lần lượt từng giá trị, sau cùng chia cho tổng số các điểm dữ liệu sẽ có được giá trị trung bình. Tuy nhiên đối với giá trị trung vị, đầu tiên chúng ta mất công xử lý sắp xếp dữ liệu từ bé tới lớn, sau đó lấy giá trị ở giữa. Với lượng dữ liệu lớn, bạn sẽ cảm nhận điều này rõ hơn hết. Sau đây ta sẽ sử dụng NumPy để tính toán giá trị trung bình cộng và giá trị trung vị, cũng như so sánh thời gian tính toán bằng `time.time()` .

```
import numpy as np
import time
# Xuat ngau nhien 1 trieu so tu phan bo tieu chuan
randoms = np.random.randn(10**7)
# Thoi gia truoc khi tinh toan(sec)
before_mean = time.time()
# Tinh gia tri trung binh cong
mean = np.mean(randoms)
# Thoi gian sau khi tinh toan gia tri trung binh cong(sec)
after_mean = time.time()
print('mean is {} (time:{:.2f}s)'.format(mean, after_mean-before_mean))
# Thoi gian truoc khi tinh toan(sec)
before_median = time.time()
# Tinh toan gia tri trung vi
median = np.median(randoms)
# Thoi gian sau khi tinh toan gia tri trung vi
after_median = time.time()
print('median is {} (time:{:.2f}s)'.format(median, after_median-before_median))
```

Kết quả:

```
mean is 0.00015996733320892628 (time:0.01s)
median is -4.2001016778634444e-05 (time:0.14s)
```

`time.time()` sẽ trả về đại lượng thời gian có đơn vị là giây khi nó được thực thi. Hãy nhớ lấy nó khi bạn cần tính toán thời gian thực thi một xử lý nào đó trong Python. Ngoài ra `{:.2f}` có nghĩa là sẽ hiển thị giá trị số thập phân tới hai chữ số sau dấu phẩy. Khi tính toán thời gian và sau đó hiển thị thời gian, nó trở nên tiện lợi, hãy nhớ lấy nó. Từ phân phối tiêu chuẩn lấy ngẫu nhiên các giá trị thì tôi nghĩ giá trị trung bình có lẽ sẽ là số xấp xỉ 0. Và giá trị trung vị cũng xấp xỉ là 0. (Phân phối tiêu chuẩn là phân phối chuẩn với giá trị trung bình là 0 và phương sai-mức độ phân tán là 1).

3.2 Giá trị xuất hiện nhiều lần nhất: Tối Tần Trị

Với tư cách là một giá trị đại diện, không phải là giá trị trung bình, cũng không phải là giá trị trung vị, trong phần này tôi giới thiệu một giá trị khác, đó là Tối Tần Trị **(mode)**. Tối tần trị có nghĩa là giá trị được quan sát thấy xuất hiện nhiều lần nhất trong bảng

phân bố dữ liệu. Nếu ta sắp xếp dữ liệu theo số lần xuất hiện của chúng, có thể thấy tối tần trị giống như là một đỉnh núi cao nhất. Trong xác suất thống kê, có thể nói đây là giá trị có mật độ xuất hiện nhiều lần nhất.

Ví dụ nếu dữ liệu thống kê cực kỳ tập trung vào một giá trị nhất định, khi đó có thể sử dụng giá trị tối tần trị như là một giá trị đại diện thì thích hợp hơn giá trị trung bình hoặc trung vị.

Ta có thể sử dụng `scipy.stats.mode()` để tính tối tần trị.

```
from scipy import stats
# Tra ve Toi Tan Tri (mode) va so lan xuat hien cua no
mode, count = stats.mode([6, 2, 4, 5, 1, 3, 5, 3, 4])
# Neu tim thay hai gia tri Toi Tan Tri, no se tra ve gia tri nho hon. Trong vi du nay so
    3 xuat hien 2 lan, so 4 xuat hien 2 lan.
# Vi vay gia tri tra ve la 3, vi 3 < 4
print(mode)
print(count)
```

Kết quả:

```
[3]
[2]
```

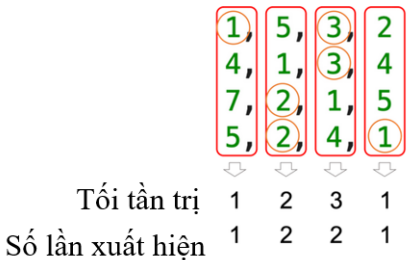
Nếu đầu vào là một mảng **NumPy Array** hai chiều. Khi đó mode sẽ được tính như thế nào?

```
array = np.array([[1, 5, 3, 2],
[4, 1, 3, 4],
[7, 2, 1, 5],
[5, 2, 4, 1]])
mode, count = stats.mode(array, axis=0)
print(mode)
print(count)
```

Kết quả:

```
[[1 2 3 1]]
[[1 2 2 1]]
```

Ở trên ta chỉ định `axis=0` có nghĩa là nó sẽ duyệt từng cột dữ liệu trong ma trận. Trong mỗi cột dữ liệu sẽ tồn tại một giá trị mode. Và ta có bốn cột dữ liệu, cho nên sẽ có bốn giá trị mode. Tương ứng với mỗi mode lại có số lần xuất hiện của giá trị mode đó trong cột dữ liệu chứa nó.



Chế độ mặc định là `axis=0`, tuy nhiên bạn có thể chỉ định `axis=1`, khi đó nó sẽ duyệt theo hàng. Bạn hãy thử thực hành nhé.

3.3 Tổng kết

Ngoài giá trị trung bình, tôi đã giới thiệu thêm hai giá trị đại diện mà các bạn có thể sử dụng đó là trung vị và tối tần trị. Tối Tần Trị: Tối có nghĩa là tối đa, cực hạn. Tần trong từ Tần Suất. Trị trong từ Giá Trị. Có nghĩa là giá trị xuất hiện nhiều lần nhất.

- Giá trị trung vị là giá trị nằm ở trung tâm trong dữ liệu sau khi đã được sắp xếp tuần tự từ nhỏ đến lớn.
- Giá trị trung vị ít bị ảnh hưởng bởi giá trị nhiễu nằm xa vùng phân bố của dữ liệu. Đây là điểm nổi trội so với giá trị trung bình.
- So với giá trị trung bình thì để tính toán giá trị trung vị cần phải sắp xếp lại dữ liệu, vì thế mà phép tính tìm trung vị sẽ mất nhiều thời gian xử lý hơn.
- Để tính toán thời gian xử lý trong `Python` có thể sử dụng `time.time()`
- Tối tần trị là giá trị xuất hiện nhiều lần nhất trong dữ liệu.

Như vậy câu chuyện của chúng ta đã kết thúc, không có gì khó hiểu đúng không mọi người. Tất nhiên các giá trị này sẽ được sử dụng khi giải thích các phân phối khác nhau trong thống kê trong tương lai, vì vậy hãy ghi nhớ chúng.

Bài 4

Mức độ phân tán (Sử dụng phạm vi và vị trí phần tư)

Nếu sử dụng giá trị đại diện thì giá trị như thế nào sẽ mang tính đặc trưng của toàn bộ dữ liệu? Đây hẳn nhiên là điều mà chúng ta muốn biết. Tuy nhiên nếu chỉ nói về giá trị đại diện thì cũng có trường hợp chúng ta không thể biết hết các tính chất của toàn bộ dữ liệu. Quay trở lại bài toán về trọng lượng các quả táo. Do trọng lượng của chúng có độ chênh lệch không nhiều, do đó chỉ với giá trị trung bình cộng đã truyền đạt đầy đủ thông tin về trọng lượng của những trái táo trong vụ thu hoạch này. Thế nhưng giả sử chúng ta có năm con chó với những trọng lượng khác nhau. Nếu chỉ đánh giá qua trọng lượng trung bình thì thật khó phải không nào. Trong trường hợp có những con chó to, con chó nhỏ, thì chỉ với trọng lượng trung bình chúng ta không thể đánh giá được hết tính chất của toàn bộ dữ liệu. Những trường hợp như thế, khi mà giá trị của các dữ liệu có sự chênh lệch nhiều, chúng ta cần một chỉ số mà qua đó có thể đánh giá được độ phân bố của dữ liệu, cùng với các giá trị đại diện qua đó đánh giá đúng đắn các tính chất của toàn bộ dữ liệu. Và trong phần này chúng ta sẽ nói tới độ phân tán để biểu thị mức độ phân tán của dữ liệu.

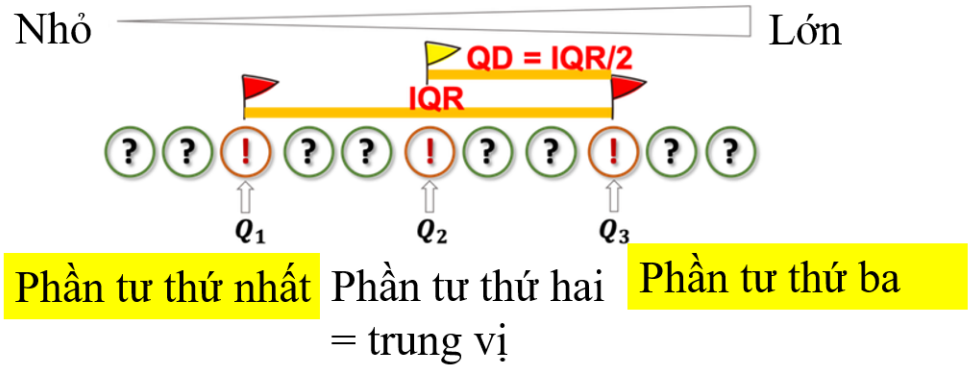
4.1 Phạm vi của giá trị

Trong các con chó mà bạn nuôi, chúng có trọng lượng lần lượt là 10kg, 13kg, 17kg, 20kg, 29kg. Khi đó trọng lượng trung bình là 17.8kg nhưng nếu chỉ dựa vào chỉ số này mà nói các con chó có trọng lượng khoảng 17.8kg thì chưa thỏa đáng, vì từ con chó nhỏ nhất là 10kg tới con chó lớn nhất là 29kg là một khoảng cách khá lớn, mức độ chênh lệch giữa con nhỏ nhất và lớn nhất là 19kg, do đó nếu chỉ nói về trọng lượng trung bình thì thật khó hình dung thực tế ra sao.
Thay vì trả lời con chó tôi nuôi có trọng lượng trung bình là 17.8kg, trả lời một cách đầy đủ rằng: Con chó tôi nuôi có trọng lượng trung bình là 17.8kg và trong phạm vi từ 10kg đến 29kg, khi đó người nghe sẽ hình dung ra được mức độ phân tán của dữ liệu. Độ phân tán của dữ liệu có thể biểu thị bằng Giá Trị Lớn Nhất - Giá Trị Nhỏ Nhất. Giá trị này được gọi là phạm vi (range).

Giá trị trung bình là 17.8kg và có phạm vi là 19kg, lúc này người nghe có thể hình dung được thực tế ra sao. Tuy nhiên nếu như chỉ thế mà không suy xét các tình huống có những giá trị gây nhiễu, tức các trường hợp ngoại lệ, các thông tin không có ý nghĩa bị lẫn vào trong các dữ liệu của chúng ta, thì chúng ta sẽ không giải quyết triệt để từ các vấn đề có trong thực tế.
Do đó tiếp theo tôi sẽ giới thiệu một chỉ số thường được sử dụng, đó là vị trí phần tư.

4.2 Phạm vi sử dụng phần tư và độ lệch phần tư

Các giá trị cực trị như giá trị tối thiểu và tối đa dễ bị ảnh hưởng bởi các giá trị ngoại lệ, do đó, chiến lược là sử dụng các giá trị gần giá trị trung bình hơn một chút để xác định mức độ phân tán.
Dữ liệu sau khi được sắp xếp theo chiều tăng dần, ta chia chúng thành bốn phần bằng nhau dựa vào ba cột mốc Q_1, Q_2, Q_3. Chúng được gọi là vị trí phần tư (quartile).
Giá trị tại vị trí chia dữ liệu thành bốn phần bằng nhau khi dữ liệu được sắp xếp (đây được gọi là phần tư) được sử dụng để thu được độ lệch phần tư (thiên sai phần tư).

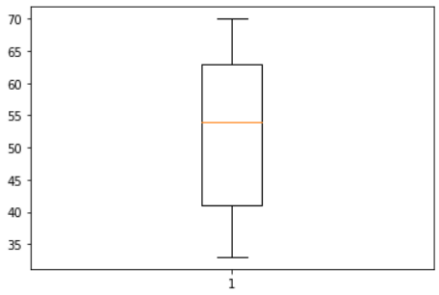


Phương pháp chia bốn phần bằng nhau rất đơn giản. Đầu tiên tìm tới giá trị trung vị, nó chia dữ liệu làm hai nửa trước và nửa sau.
Với mỗi nửa này ta lại tìm trung vị của chúng, vậy là xong. Nếu dữ liệu có lẻ số hạng thì thông thường sẽ lấy giá trị trung bình cộng hai số ở giữa.
Mỗi một phần tư sẽ có các điểm Q_1, Q_2, Q_3 gọi là phần tư thứ nhất, phần tư thứ hai, phần tư thứ ba. Chúng ta lưu ý phần tư thứ hai chính là giá trị trung vị.
Vậy thì $Q_3 - Q_1$ sẽ được coi là phạm vi phần tư (interquartile range: IQR), một nửa giá trị này là $\dfrac{Q_3 - Q_1}{2}$ được gọi là thiên sai phần tư hay độ lệch phần tư (QD: quartile deviation). Nhìn vào biểu đồ trên có thể thấy QD bằng $Q_3 - Q_2$, hãy cẩn thận vì điều đó không chính xác.
IQR, QD là một chỉ số được sử dụng để biểu thị độ phân tán của dữ liệu. Nó không chịu ảnh hưởng như phạm vi (range) cho dù có giá trị gây nhiễu-hay còn gọi là giá trị ngoại lệ xuất hiện trong dữ liệu của chúng ta.
Ta có thể sử dụng `scipy.stats.iqr()` để tính IQR.

```
from scipy import stats
import matplotlib.pyplot as plt
%matplotlib inline
data = [33, 35, 36, 39, 43, 49, 51, 54, 54, 56, 62, 64, 64, 69, 70]
iqr = stats.iqr(data)
qd = iqr/2
print('IQR: {}'.format(iqr))
print('QD: {}'.format(qd))
# Chieu cao cua box la IQR
plt.boxplot(data)
plt.show()
```

Kết quả:

```
IQR: 22.0
QD: 11.0
```



4.3 Điểm hạn chế của phạm vi và phạm vi phần tư

Bạn thấy thế nào? Chỉ số này có lẽ là khó hiểu đúng không?
Phạm vi có nhược điểm mang tính quyết định là dễ bị ảnh hưởng bởi các yếu tố ngoại lệ, các dữ liệu gây nhiễu sẽ làm cho phạm vi không phản ánh chính xác bản chất của dữ liệu, nhưng phạm vi phần tư và độ lệch giữa các phần tư mặc dù được hiểu là nơi mà dữ liệu phân tán tập trung nhất ở đó cũng không đủ để mô tả mức độ phân tán dữ liệu vì không phải tất cả dữ liệu đều được sử dụng ở đây. Tuy nhiên bản thân nó cũng chưa đủ tốt và có thể bạn cũng chưa cảm thấy thích nó hoàn toàn.
Do đó chúng ta có một chỉ số dễ hiểu hơn có thể sử dụng cho toàn bộ dữ liệu, đó là trung bình thiên sai, hay là trung bình độ sai khác. Ở phần tiếp theo tôi sẽ thuyết minh về phân tán và độ lệch chuẩn.
Nói về độ phân tán có thể nói rằng khái niệm phân tán và độ lệch chuẩn xuất hiện áp đảo, tôi nghĩ nhiều người có thể đã nghe về nó. Đây là một nội dung vô cùng quan trọng, tôi xin phép sẽ trình bày ở phần sau.

4.4 Tổng kết

Trong bài học này, tôi đã giới thiệu cho các bạn về phạm vi phần tư và độ lệch phần tư (thiên sai phần tư) , phạm vi từ giá trị cực đại tới giá trị cực tiểu để biểu thị mức độ dàn trải (phân tán) của dữ liệu.

- Phạm vi (range) = Giá Trị Cực Đại - Giá Trị Cực Tiểu: Dễ bị ảnh hưởng bởi giá trị ngoại lệ (dữ liệu gây nhiễu).
- Phần tư (quartile): Giá trị tại vị trí chia dữ liệu thành bốn phần bằng nhau khi mà dữ liệu đã được sắp xếp từ nhỏ tới lớn.
- Phần tư thứ hai là giá trị trung vị.
- Phạm vi phần tư $(IQR) = Q_3 - Q_1$. Ít bị ảnh hưởng bởi giá trị ngoại lệ.
- Độ lệch phần tư (thiên sai phần tư) (QD) : là một nửa của IQR.
- Những khái niệm nói trên không biểu diễn toàn bộ dữ liệu, chưa đầy đủ.

Phần tư là một nội dung vô cùng quan trọng trong thống kê, hãy ghi nhớ các khái niệm này. Trong khóa học dữ liệu, sử dụng phần tư để đánh giá mức độ phân tán dữ liệu là rất thường xuyên. Tuy nhiên, từ quan điểm của độ phân tán, thì phạm vi phần tư và thiên lệch phần tư chưa thích hợp để đánh giá mức độ dàn trải của dữ liệu.
Phần tiếp theo sẽ giới thiệu một chỉ số quan trọng trong các chỉ số của mức độ phân tán, đó là **phân bố** và **độ lệch chuẩn**. Phân tán và độ lệch chuẩn là nội dung thường thấy trong thống kê học và trong máy học. Vì quan trọng như vậy nên các bạn hãy gắng học nhé.

Bài 5

Nhất định hiểu về phân tán và độ lệch chuẩn

Nói về độ phân tán, ở bài trước chúng ta đã biết tới các khái niệm phạm vi, độ lệch phần tư QD và phạm vi phần tư IQR. Đó là những thứ rất cơ bản nhưng nó có khuyết điểm là không được sử dụng trong tính toán về chỉ số của toàn bộ dữ liệu.
Do đó trong bài học này, để bổ sung khiếm khuyết đó tôi sẽ giới thiệu thêm về phân tán và độ lệch chuẩn, nó là một trong những lý thuyết quan trọng nhất trong thống kê học.

- Độ lệch trung bình
- Phân tán
- Độ lệch chuẩn

Từng khái niệm sẽ được nhắc tới. Sau đó chúng ta sẽ sử dụng Python để tính toán. Ngoài ra sẽ còn nói tới phương sai bất thiên(phân tán bất thiên).

5.1 Độ lệch trung bình

Vì phạm vi cũng như IQR, QD không được sử dụng trong tính toán chỉ số của toàn bộ dữ liệu, do đó chúng ta đã có câu chuyện đó là chưa có một giá trị biểu thị đầy đủ về tính phân tán của toàn bộ dữ liệu. Vậy để nói tới độ phân tán của toàn bộ dữ liệu, thì đơn giản nhất là cách nào đây?

"Mỗi giá trị cách giá trị trung bình bao xa" được gọi là độ lệch và câu chuyện rằng việc cộng các độ lệch thường dẫn đến kết quả bằng 0 đã được đề cập trong bài học 2.

Điều này là đương nhiên vì khi ta thực hiện phép tính độ sai khác từ các điểm dữ liệu tới giá trị trung bình $(x_i - \bar{x})$ khi thì có giá trị dương, khi thì có giá trị âm, cuối cùng chúng triệt tiêu lẫn nhau và kết quả sẽ là 0. Như vậy, nếu tôi chỉ quan tâm khoảng cách là bao xa mà không quan tâm giá trị âm hay dương, tức là lấy giá trị tuyệt đối của độ lệch từ

các điểm dữ liệu tới các giá trị trung bình thì có được không?

Công thức $|x_i - \bar{x}|$ có vẻ tốt. Nếu lấy các giá trị này cộng lại và chia cho số điểm dữ liệu để lấy giá trị trung bình, chỉ số này có thể được sử dụng để biểu diễn mức độ phân tán. Nói tóm lại, ta có thể sử dụng trung bình cộng của giá trị tuyệt đối thiên sai từ các điểm dữ liệu tới giá trị trung bình cộng để biểu thị mức độ phân tán dữ liệu. Giá trị này được gọi là độ lệch trung bình (thiên sai trung bình) **(mean deviation)** hay độ lệch trung bình tuyệt đối **(mean absolute deviation)**. Nó còn được biết đến với tên MD.
Ta có công thức:

$$MD = \frac{1}{n}\left(|x_1 - \bar{x}| + |x_2 - \bar{x}| + \cdots + |x_n - \bar{x}|\right) = \frac{1}{n}\sum_{i=1}^{n}|x_i - \bar{x}|$$

Điều này thật dễ hiểu khi đại diện cho sự thay đổi trong dữ liệu, phải không? Nếu mỗi dữ liệu nằm rải rác, giá trị tuyệt đối của độ lệch của mỗi giá trị đương nhiên sẽ lớn, do đó MD sẽ lớn, và nếu nó nhỏ, MD sẽ nhỏ.
Khi toàn bộ dữ liệu giống nhau thì độ phân tán MD sẽ là 0. Khi đó đương nhiên có thể nói rằng độ phân tán là 0.
Như vậy chúng ta đã giải quyết xong vấn đề về độ phân tán hay độ phân tán của dữ liệu. Tuy nhiên còn một vấn đề nữa đó là trong công thức trên sử dụng dấu giá trị tuyệt đối. Ồ, dùng dấu giá trị tuyệt đối thì có sao đâu?

Không chỉ trong MD, nói chung trong thống kê học người ta có khuynh hướng tránh dùng giá trị tuyệt đối. Tại sao? Bởi vì việc tính toán giá trị khi âm hay khi dương làm cho phép toán thay đổi, điều đó thật là phiền hà.

Vậy thì làm thế nào để tránh được rắc rối này? Đó chính là thực hiện lũy thừa bậc 2 (bình phương), khi đó giá trị âm sẽ trở thành giá trị dương mà không cần sử dụng tới dấu giá trị tuyệt đối.

5.2 Phân tán và độ lệch chuẩn

Phân tán (variance) -các tài liệu khác gọi nó là phương sai, là bình phương độ lệch (thiên sai).
Phân tán $= \frac{1}{n}\left((x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + \cdots + (x_n - \bar{x})^2\right)$
Thật vui khi dấu giá trị tuyệt đối đã biến mất. Tuy nhiên, nếu nó được bình phương, nó sẽ lệch khỏi thang của giá trị ban đầu. (Ví dụ, giả sử trọng lượng trung bình là 10kg và độ lệch là 2kg. Nếu bạn bình phương nó, nó sẽ là 4kg, điều này làm cho nó khó giải thích giá trị, phải không?)
Sẽ rất tuyệt nếu lấy căn bậc hai của phân tán (phương sai) để phù hợp với tỷ lệ như đã nói ở trên, phải không? Căn bậc hai của phương sai. Đó là độ lệch chuẩn**(standard deviation)**! Độ lệch chuẩn thường là từ viết tắt s của độ lệch chuẩn. (Nói chung, sử dụng

σ (sigma) cho độ lệch chuẩn của dữ liệu tổng thể và s cho độ lệch chuẩn của mẫu dữ liệu nhỏ hơn.)

$$s = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Độ lệch chuẩn và phân tán (còn gọi là phương sai) là cơ sở lý luận rất quan trọng trong thống kê học, các bạn hãy nắm vững về nó.

5.3 Sử dụng Python để tính toán phân tán và độ lệch chuẩn

Ta có thể sử dụng `.std()` của thư viện NumPy để tính độ lệch chuẩn. Ngoài ra để tính độ phân tán, ta có thể sử dụng hàm `.var()` trong thư viện NumPy. Nào, đầu tiên ta sẽ tính độ phân tán từng bước như sau:

```
import numpy as np
def get_variance(samples):
    # Tinh gia tri trung binh
    mean = np.mean(samples)
    # Tinh do lech (thien sai)
    deviations = samples - mean
    # Luy thua bac 2 doi voi thien sai
    square_deviations = deviations * deviations
    # Tinh tong cac luy thua bac 2 cua thien sai
    sum_square_deviations = np.sum(square_deviations)
    # Lay tong luy thua bac 2 cua thien sai chia cho so luong du lieu(do phan tan)
    variance = sum_square_deviations/len(samples)
    return variance
```

Trông có vẻ dài nhỉ, nhưng không hề khó chút nào đúng không. Hãy xác nhận từng dòng code thử xem sao. Nếu bạn cảm thấy lo lắng, hãy sử dụng JupyterLab để chạy từng dòng code và xác nhận kết quả.
Nào bây giờ ta hãy xác nhận kết quả:

```
samples = [10, 10, 11, 14, 15, 15, 16, 18, 18, 19, 20]
# Tinh do phan tan bang ham tu tao:
print(get_variance(samples))
# Tinh do phan tan bang ham trong NumPy:
print(np.var(samples))
```

Kết quả:

```
11.537190082644628
11.537190082644628
```

Như vậy kết quả là giống nhau, bây giờ ta hãy cùng nhau tính toán độ lệch chuẩn.

```
#Tinh do lech chuan thong qua do phan tan
print(np.sqrt(get_variance(samples)))
# Tinh do lech chuan thong qua ham co san trong NumPy
print(np.std(samples))
```

Kết quả:

```
3.3966439440489826
3.3966439440489826
```

Như vậy kết quả là giống nhau đúng như những gì chúng ta đã dự đoán từ trước. Ngoài NumPy, người ta cũng có thể tính độ lệch chuẩn và độ phân tán thông qua stats của SciPy hay Pandas.
Đầu tiên hãy thử với `scipy.stats`. Đối với SciPy, độ phân tán và độ lệch chuẩn lần lượt được tính bởi các hàm `scipy.stats.tvar()` và `scipy.stats.tstd()`. Chữ "t" xuất hiện trong tên của các hàm có ý nghĩa là **t**rimmed. Bạn có thể chỉ định một phạm vi giá trị để sử dụng cho phép tính để bạn có thể xử lý các giá trị ngoại lệ-các giá trị gây nhiễu (giống với hình ảnh cắt bỏ các phần thừa ở hai đầu dữ liệu). Lần này tôi sẽ sử dụng nó như nó vốn có.

```
from scipy import stats
# Tinh do phan tan
print(stats.tvar(samples))
# Tinh do lech chuan
print(stats.tstd(samples))
```

Kết quả:

```
12.690909090909091
3.562430222602134
```

Ồ, sao kỳ vậy, kết quả khác với ban nãy khi ta tính toán bằng NumPy. Với NumPy độ phân tán là 11.5 và độ lệch chuẩn là 3.4 trong khi đó với SciPy thì độ phân tán là 12.7 và độ lệch chuẩn là 3.6, chúng có giá trị cao hơn một chút so với kết quả của NumPy.

5.4 Phân tán và Phân tán bất thiên

Cái giá trị phân tán được tính toán bởi hàm `tvar()` và `tstd` trong Module stats của thư viện SciPy thực ra là giá trị phân tán bất thiên. Phân tán bất thiên có vẻ là một từ ngữ khó hiểu, tôi sẽ giải thích ở bài sau. Công thức tính phân tán bất thiên khác phân tán trước đó chúng ta đã biết đó là phép tính không chia cho n mà chia cho $n-1$.

$$\text{Phân tán bất thiên} = \frac{1}{n-1}\left((x_1-\bar{x})^2+(x_2-\bar{x})^2+\cdots+(x_n-\bar{x})^2\right) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2$$

Sao kỳ vậy? Dây có lẽ là phản ứng thường thấy từ mọi người khi nhìn vào công thức trên. Tuy nhiên không hiểu thì cũng không sao đâu. Tôi nghĩ rằng đây là thời điểm khó khăn cho mọi người từ bỏ công thức trước đây để làm quen với công thức mới này. Do đó tôi sẽ giải thích trong bài học tiếp theo. `scipy.stats.tvar()` và `scipy.stats.tstd()` có kết quả lớn hơn `np.var()` và `np.std()` bởi vì chúng không phải chia cho n mà là chia cho $n-1$. Với Pandas cũng vậy, chúng cũng tính toán cho kết quả là phân tán bất thiên. Hãy tham khảo code dưới đây.

```
import pandas as pd
samples = [10, 10, 11, 14, 15, 15, 16, 18, 18, 19, 20]
df = pd.DataFrame({'sample':samples})
```

```
print(df['sample'].var())
print(df['sample'].std())
```

```
12.690909090909093
3.5624302226021345
```

Kết quả này giống với khi chúng ta sử dụng `scipy.stats`. Tại sao `scipy.stats` và `Pandas` lại chia cho $n-1$ để tính phân tán bất thiên, trong khi đó `NumPy` lại chia cho n để tính độ phân tán? Tại sao lại có hai loại phân tán? Chúng ta sẽ làm rõ điều này ở bài học tiếp theo.

5.5 Tổng kết

Trong bài học này, với tư cách là độ phân tán (phân bố), tôi đã giới thiệu trung bình thiên sai (độ lệch trung bình), phân tán, độ lệch chuẩn (thiên sai tiêu chuẩn) cho các bạn. Không giống như IQR và QD sử dụng phạm vi và vi trí phần tư, chúng có nguyên tắc cũng như đặc điểm là sử dụng tất cả dữ liệu để tính toán.
Đặc biệt, phân tán (phương sai) và độ lệch chuẩn là một trong những mục quan trọng nhất trong lý thuyết thống kê, vì vậy hãy lưu ý ghi nhớ chúng nhé.

- Độ lệch trung bình (MD) : Là giá trị trung bình của giá trị tuyệt đối độ lệch ($|x_i - \bar{x}|$). Xử lý dấu giá trị tuyệt đối là một rắc rối cần giải quyết.
- Phân tán còn gọi là phương sai (s^2) : Trung bình lũy thừa bậc 2 của độ lệch $\left((x_i - \bar{x})^2\right)$.
- Độ lệc chuẩn (s): Căn bậc hai của phân tán.
- `np.var()` và `np.std()` tính độ phân tán và độ lệch chuẩn.
- `scipy.stats.tvar()` và `scipy.stats.tstd()` có thể tính toán độ phân tán và độ lệch chuẩn nhưng kết quả tính toán là phân tán bất thiên.
- Phân tán bất thiên sử dụng công thức phân tán nhưng không chia cho n mà chia cho $n-1$.

Bài học này là tương đối dài nhưng đây là nội dung vô cùng quan trọng. Tôi nghĩ các bạn nên theo dõi câu chuyện từ: Phạm vi $\longrightarrow$ IQR/QD $\longrightarrow$ MD $\longrightarrow$ Phân tán $\longrightarrow$ Độ lệch chuẩn.

Bài 6

Phân tán bất thiên là gì? Tại sao phân tán từ dữ liệu tiêu bản lại nhỏ hơn phân tán từ dữ liệu cha?

Trong bài trước với tư cách là chỉ số biểu hiện mức độ phân tán hay phân bố, ta đã nói về những chỉ số rất quan trọng như phân tán s^2 và độ lệch chuẩn s. Chúng ta có thể sử dụng `NumPy` hay `scipy.stats` hay `Pandas` để tính giá trị phân tán và độ lệch chuẩn. Tuy nhiên chúng ta cũng nhận ra rằng kết quả tính toán của `scipy.stats` và `Pandas` cho kết quả khác với `NumPy`. Điều này được giải thích đó là vì `scipy.stats` và `Pandas` đã đưa ra kết quả tính toán phân tán là phân tán bất thiên.
Phân tán bất thiên có nghĩa là sử dụng công thức phân tán nhưng không chia cho n mà chia cho $n - 1$.
Trong bài học lần này chúng ta sẽ làm rõ rút cuộc phân tán bất thiên là gì?

- Phân tán bất thiên là chỉ số được sử dụng để ước tính độ phân tán của dữ liệu tiêu bản (dữ liệu mẫu) từ dữ liệu cha.

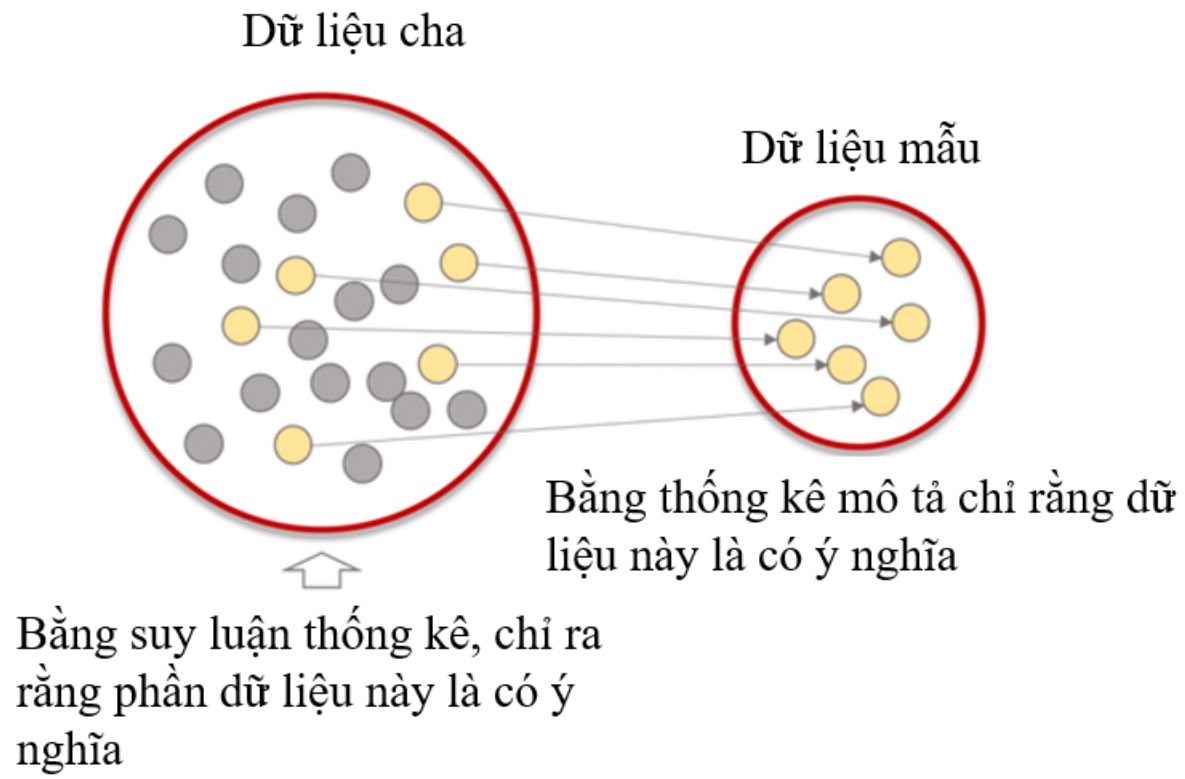
- Như lý giải ở trên thì lý do mà `NumPy` không sử dụng phân tán bất thiên là vì hóa ra `NumPy` chỉ đang tính toán chỉ số mô tả về phân tán của dữ liệu được lấy làm đối số.

- Mặt khác, `scipy.stats` và `Pandas` được cho là được sử dụng trong thống kê và khoa học dữ liệu, vì vậy chúng trả về độ phân tán bất thiên.

- Phân tán bất thiên dễ sử dụng hơn phân tán thông thường trong việc xây dựng lý thuyết thống kê, do đó, có nhiều công cụ và thư viện trả về phân tán bất thiên theo mặc định.

6.1 Ước lượng phân tán của dữ liệu cha như thế nào thì tốt?

Phần chính của thống kê học là tìm ra đặc tính của dữ liệu cha bằng cách sử dụng mẫu dữ liệu có giới hạn mà ta gọi là dữ liệu tiêu bản. Tóm lại đây là suy luận thống kê.



Từ các bài học trước cho đến lúc này những gì tôi đã diễn giải cho các bạn đó là thống kê mô tả. Thống kê mô tả đó là mô tả đặc tính của dữ liệu từ dữ liệu vốn có trong tay. Hãy tách biệt một cách rõ ràng sự khác biệt này khi các bạn học thống kê.
Chúng ta sẽ xem xét chi tiết ở phần sau, chúng ta sẽ sử dụng chỉ số của dữ liệu tiêu bản để suy ra chỉ số dân số. Ví dụ, chúng ta biết rằng giá trị trung bình của một dữ liệu tiêu bản có thể được sử dụng để suy ra giá trị trung bình của cả một tập hợp (dữ liệu cha). Chi tiết sẽ được giải thích lại ngay trong khóa học này.
Nói cách khác, ngay cả khi bạn nói "trung bình" thì nó sẽ có nghĩa khác nhau trong những ngữ cản thống kê mô tả và trong ngữ cảnh thống kê suy luận.
Theo cách này, giá trị của dữ liệu tiêu bản được sử dụng để ước tính giá trị đặc trưng của dữ liệu cha, và phép thống kê được sử dụng để ước tính đó được gọi là ước lượng. Bạn không cần thiết phải nhớ những từ ngữ này. Trong công việc không có nhiều trường hợp mà bạn cần phải phân biệt rõ ràng điều này. Đặt tên trong khi học thống kê là rất hữu ích, vì vậy trong cuốn sách này tôi sẽ giới thiệu các từ này và tôi sẽ sử dụng chúng.
Tóm lại, giá trị trung bình được tính từ dữ liệu tiêu bản (dữ liệu mẫu) được sử dụng bằng công cụ ước lượng để từ đó ước tính giá trị trung bình của dữ liệu cha. Vậy khi ước tính độ phân tán của dữ liệu cha có thể sử dụng ước lượng hay không? Có thể sử dụng ước

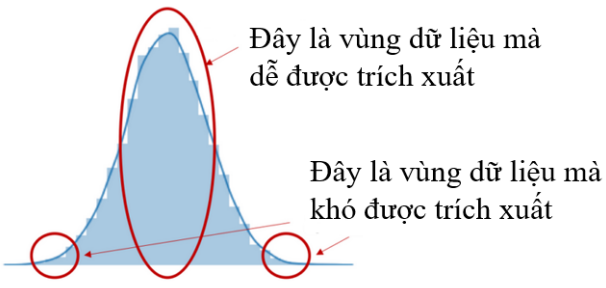
lượng độ phân tán của dữ liệu tiêu bản hay không?
Trong thực tế, độ phân tán của dữ liệu tiêu bản nhỏ hơn một chút so với độ phân tán của dữ liệu cha. Đầu tiên bạn hãy lý giải điều này bằng cảm giác cũng được.

6.2 [Lý giải bằng hình ảnh]Tại sao độ phân tán của dữ liệu tiêu bản lại nhỏ hơn độ phân tán của dữ liệu cha?

Cho tới bây giờ, đây hẳn là câu hỏi các bạn vẫn còn đang băn khoăn. Nhưng nếu suy nghĩ kỹ một chút ta sẽ thấy điều này là đương nhiên.
Khi mà chúng ta lấy ngẫu nhiên từ dữ liệu cha để hình thành nên một dữ liệu mẫu, có thể nói rằng dữ liệu mẫu (dữ liệu tiêu bản) là một phần của dữ liệu cha. Vì chúng ta không có lấy giá trị nào nằm ngoài phạm vi của dữ liệu cha, vì vậy độ phân tán của dữ liệu mẫu rõ ràng có thể nhỏ hơn độ phân tán của dữ liệu cha, đây là điều đương nhiên.
Sau đây chúng ta hãy cùng thực hiện đoạn code dưới đây:

```
population = np.array([1, 5, 10, 11, 14, 15, 15, 16, 18, 18, 19, 20, 25, 30])
# Trich xuat du lieu:
samples = np.random.choice(population, size=3)
print('population variance is {}'.format(np.var(population)))
print('sample variance is {}'.format(np.var(samples)))
print('samples: {}'.format(samples))
```

Bằng `np.var()` ta sẽ tính độ phân bố của `population` với tư cách là dữ liệu cha ($[1, 5, 10, 11, 14, 15, 15, 16, 18, 18, 19, 20, 25, 30]$) và tính độ phân bố của `sample` với tư cách là dữ liệu mẫu hay dữ liệu tiêu bản.
Bạn hãy thực thi code trên nhiều lần, trong nhiều trường hợp chẳng phải độ phân tán của dữ liệu mẫu nhỏ hơn độ phân tán của dữ liệu cha là 51.39 hay sao?
Trừ khi bạn lấy sự kết hợp ở các đầu mút, chẳng hạn [1,5,30] hoặc [1,25,30], còn bình thường thì độ phân tán của dữ liệu mẫu có xu hướng thấp hơn độ phân tán của dữ liệu cha. Do lấy ngẫu nhiên, dữ liệu mẫu được trích xuất từ dữ liệu cha, vì vậy phân bố của dữ liệu mẫu thường có giá trị lệch so với phân bố của dữ liệu cha. Ngược lại xác xuất mà một giá trị nằm ngoài phân bố thường có khả năng thấp, và tất nhiên độ phân tán dữ liệu mẫu có khuynh hướng nhỏ hơn độ phân tán của dữ liệu cha $\left(\frac{n-1}{n}\right)$.



Nhân tiện hình này có thể được vẽ như sau:

```
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
sns.distplot(np.random.randn(int(1e4)), bins=30)
plt.axis('off')
```

`seaborn` là thư viện của Python được sử dụng trong khoa học dữ liệu. Về `displot` nó được giới thiệu trong khóa học nhập môn Python cho Data science do diễn đàn tuhocvba.net tổ chức, bạn hãy tham gia đăng ký học nhé. Tôi nghĩ điều này đã cho các bạn thấy rằng phương sai (độ phân tán) của dữ liệu mẫu hay còn gọi là dữ liệu tiêu bản có xu hướng nhỏ hơn phương sai của dữ liệu cha. Số lượng dữ liệu của dữ liệu mẫu (n) càng nhỏ thì xác suất các giá trị lệch khỏi phân bố được đưa vào dữ liệu mẫu càng ít, vì vậy bạn có thể thấy một hình ảnh rõ ràng rằng độ phân tán cũng sẽ trở nên nhỏ hơn.

6.3 [Lý giải bằng Số Học]Tại sao độ phân tán của dữ liệu tiêu bản lại nhỏ hơn độ phân tán của dữ liệu cha?

Nào, bây giờ chúng ta hãy thử giải thích bằng kiến thức toán. Vì các giá trị đặc trưng của dữ liệu cha thường được biểu thị bằng các chữ cái Hy Lạp, nên trong khóa học này, giá trị trung binh của dữ liệu cha sẽ được ký hiệu là μ. Và độ phân tán sẽ được ký hiệu là σ^2.
Độ phân tán của dữ liệu cha la σ^2 và của dữ liệu mẫu là s^2 và chúng được áp dụng công thức như dưới đây, nếu như bạn không hiểu chúng, xin hãy xem lại bài học trước.

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$$

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Độ phân tán của dữ liệu cha là σ^2 được tính thông qua giá trị trung bình của dữ liệu cha là μ, trong khi đó độ phân tán của dữ liệu mẫu là s^2 được tính thông qua giá trị trung bình của dữ liệu mẫu là $\bar{x}$.
Tuy nhiên, khi tính độ phân tán của dữ liệu mẫu chúng ta sử dụng giá trị trung bình, giá trị trung bình này là $\bar{x}$, nếu thay vào đó ta sử dụng giá trị trung bình của dữ liệu cha thì có được không?

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$$

Nếu như chúng ta tính toán độ phân tán của dữ liệu mẫu dùng cho việc ước lượng độ phân tán của dữ liệu cha, như thế thì nếu sử dụng giá trị trung bình của dữ liệu cha μ thì kết quả có vẻ như sẽ tiệm cận tới độ phân tán của dữ liệu cha là σ^2.
Tuy nhiên, giá trị trung bình của dữ liệu cha là μ thông thường chúng ta không biết, do

đó dù muốn sử dụng cũng không có cách nào mà sử dụng nó. Đó là lý do chúng ta phải ước lượng giá trị trung bình trung bình $\bar{x}$ của dữ liệu mẫu thay cho giá trị trung bình của dữ liệu cha là μ.
Như đã nói trong các bài học trước, giá trị trung bình cộng là giá trị mà tổng bình phương độ sai khác từ các điểm dữ liệu tới nó là nhỏ nhất.
Chẳng hạn ta có giá trị X bất kỳ. Tổng bình phương độ sai khác từ các điểm dữ liệu trong dữ liệu mẫu tới X là:

$$\sum_{i=1}^{n} (x_i - X)^2$$

Dữ liệu mẫu có giá trị trung bình cộng là $\bar{x}$. Và tổng trên đạt giá tị nhỏ nhất khi $X = \bar{x}$. Độ lệch chuẩn của dữ liệu mẫu chính là giá trị nhỏ nhất của tổng trên.

$$s^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

Nếu như X là một giá trị khác thì tổng trên sẽ có giá trị lớn hơn. Như vậy:

$$\frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2 \leq \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2$$

Đến đây chúng ta đã làm sáng tỏ vấn đề. Lưu ý rằng giá trị trung bình của dữ liệu cha μ thường là giá trị mà chúng ta không biết, và vì thế chúng ta thường ước lượng bằng cách sử dụng giá trị trung bình của một mẫu dữ liệu nhỏ hơn là $\bar{x}$, và vì vậy mà độ phân tán khi tính toán mà sử dụng giá trị trung bình $\bar{x}$ sẽ cho ra giá trị độ phân tán nhỏ hơn một chút.

6.4 Phân tán bất thiên (Phương sai không chệch) có thể được sử dụng làm công cụ ước tính cho phương sai tổng thể (phân tán của dữ liệu cha)

Đến bây giờ có lẽ các bạn đều đã hiểu, thông thường khi tính toán độ phân tán của dữ liệu mẫu s^2 sẽ nhỏ hơn độ phân tán của dữ liệu cha là σ^2.
Độ phân tán của dữ liệu mẫu s^2 không thể sử dụng để ước lượng giá trị độ phân tán của dữ liệu cha σ^2. Phân tán bất thiên được sử dụng làm công cụ, nó có thể nhỏ hơn hay lớn hơn một chút so với độ phân tán của dữ liệu mẫu s^2. Tóm lại phân tán bất thiên là giá trị ước lượng của độ phân tán dữ liệu cha σ^2.
Vì vậy, chúng ta không sử dụng `scipy.stats` hoặc `Pandas` để tính độ phân tán một cách nhanh chóng, thay vào đó chúng ta sử dụng lý luận của thống kê học để tính độ phân tán bất thiên.
Trong lý luận của thống kê học, độ phân tán bất thiên dễ xử lý hơn độ phân tán thông thường, vì vậy mà nhiều công cụ thống kê trả về giá trị độ phân tán bất thiên một cách mặc định khi tính toán liên quan tới độ phân tán.

Thành thật mà nói, tôi không nghĩ rằng có ai đó thực sự quan tâm đến việc sử dụng độ phân tán thông thường hay độ phân tán bất thiên trong thống kê mô tả như một công cụ ước tính. Tuy nhiên để chắc chắn, bạn nên xem công cụ hoặc thư viện nào đang tính toán độ phân tán, nếu n lớn thì sự khác biệt là rất nhỏ, vì vậy tôi không nghĩ chúng ta cần phải lo lắng về điều đó, đặc biệt là với các mẫu dữ liệu lớn.

6.5 Tại sao lại là $n-1$, bất thiên là gì?

Chúng ta đã lý giải được rằng độ phân tán của dữ liệu mẫu có xu hướng nhỏ hơn độ phân tán của dữ liệu cha. Tuy nhiên có một câu hỏi đặt ra, tại sao lại chia cho $n-1$? Tại sao không chia cho $n-2$ hoặc $n-3$? Đúng vậy, đây là một thắc mắc rất tự nhiên. Để giải thích việc này trước hết chúng ta cần làm rõ "Tính Bất Thiên". Chúng ta nói về độ phân tán bất thiên, vậy thì "bất thiên" có nghĩa là gì? Để giải thích điều này sẽ rất dài và tốn một chút thời gian, do đó tôi xin phép được trình bày trong bài học tới. Tôi nghĩ rằng với người nhập môn, các bạn hãy cố gắng nhớ những điều dễ hiểu trước đã. Không cần phải cố gắng nhớ quá nhiều kiến thức một lúc, nó sẽ khiến các bạn cảm thấy quá tải.

6.6 Tổng kết

Trong bài học lần này tôi đã giải thích cho các bạn về độ phân tán bất thiên. Vậy rút cục độ phân tán bất thiên là cái gì? Tại sao khi tính toán độ phân tán của dữ liệu mẫu ta không sử dụng cách tính thông thường? Chúng ta đã lý giải được rằng nó có giá trị nhỏ hơn độ phân tán của dữ liệu cha.
Đây được coi là một nội dung vô cùng quan trọng. Nếu như các bạn vẫn còn chưa hiểu, xin hãy đọc lại nội dung bài học này thêm một lần nữa. Và hãy thử tra cứu google để tìm hiểu thêm. Sau đây tôi xin tóm tắt nội dung bài học này:

- Độ phân tán bất thiên là đại lượng được để ước lượng độ phân tán của dữ liệu cha.
- Khi tính toán độ phân tán của dữ liệu mẫu theo cách thông thường, nó sẽ có khuynh hướng có giá trị nhỏ hơn độ phân tán của dữ liệu cha.
- Lý do mà độ phân tán của dữ liệu mẫu nhỏ hơn dữ liệu cha đó là vì khi tính toán độ phân tán ta không sử dụng được giá trị trung bình của dữ liệu cha mà sử dụng giá trị trung bình của dữ liệu mẫu để tính toán.

Chúng ta vẫn còn một khúc mắc về lý do chia cho $n-1$ mà tôi chưa giải thích nhưng tôi nghĩ các bạn đã hình dung ra được độ phân tán bất thiên là gì.
Trong bài học tới tôi sẽ trình bày về tính bất thiên cũng như lý do chia cho $n-1$.

Bài 7

Lý do độ phân tán bất thiên được tính bằng phép chia cho $n-1$. Tính bất thiên nghĩa là gì?

Ở bài học trước chúng ta đã nói về câu chuyện độ phân tán bất thiên, nó là giá trị ước lượng độ phân tán của dữ liệu cha. Nếu độ phân tán của dữ liệu mẫu là s^2 và độ phân tán của dữ liệu cha là σ^2 thì độ phân tán của dữ liệu mẫu s^2 có khuynh hướng nhỏ hơn σ^2.
Do đó trong bài học này chúng ta sẽ làm rõ tính bất thiên rút cuộc là gì? Và tại sao không phải là n mà lại là chia cho $n-1$. Tôi nghĩ bất cứ ai khi bắt đầu học thống kê sẽ cùng có thắc mắc này, do đó việc làm rõ khúc mắc này là cần thiết.
Nhìn riêng về mặt ngôn ngữ ta có thể thấy:

- Độ phân tán bất thiên là một công cụ để ước tính bất thiên (không thiên vị) của độ phân tán tổng thể (độ phân tán của dữ liệu cha), nên từ "bất thiên" được sử dụng.
- Giá trị ước lượng nếu tính trung bình sẽ cho giá trị trùng với tham số của dữ liệu mẫu, khi đó công cụ ước tính này được cho là không thiên sai (không chệch) và một công cụ ước tính như thế được gọi là công cụ ước tính bất thiên.
- Tuy nhiên tại sao không phải là $n-2$ hay $n-3$ mà lại là $n-1$ thì đây là điều cần phải chứng minh làm rõ.

Sẽ có nhiều bạn băn khoăn, nãy giờ nói gì tôi cũng không hiểu nữa. Được rồi, chúng ta sẽ bắt tay vào ngay bây giờ đây, các bạn hãy kiên nhẫn một chút.

7.1 Tính bất thiên là gì?

Bạn nghĩ rằng bạn đã tính toán độ phân tán của dữ liệu cha và của dữ liệu mẫu. Tuy nhiên vì độ phân tán của dữ liệu mẫu thường có khuynh hướng nhỏ hơn độ phân tán của dữ liệu cha, chúng ta hiểu rằng độ phân tán của dữ liệu mẫu có khuynh hướng nhỏ hơn độ phân tán bất thiên, là giá trị mà chúng ta sử dụng để ước lượng độ phân tán của dữ

liệu cha, nên có thể nói rằng độ phân tán của dữ liệu mẫu không bằng với độ phân tán của dữ liệu cha, nó có một độ lệch sai khác nhất định. Cũng đúng thôi, vì dữ liệu mẫu là dữ liệu mà chúng ta trích xuất từ dữ liệu cha, tùy thuộc sự trích xuất khác nhau mà độ phân tán sẽ có giá trị khác nhau, do đó dù thế nào đi nữa, giá trị độ phân tán mà chúng ta tính được từ dữ liệu mẫu là một giá trị mang tính ước lượng.

Một ví dụ đơn giản, chúng ta thử suy nghĩ về việc ước lượng giá trị trung bình của dữ liệu cha là μ. Rõ ràng vì dữ liệu cha là một khối dữ liệu lớn cho nên giá trị trung bình thực tế là không thể tính được, do đó mới nói μ chỉ có tính ước lượng dựa vào dữ liệu mẫu có giá trị trung bình là $\bar{x}$.

Nào, các bạn hãy thử nhìn vào code dưới đây, đây là đoạn code đã sử dụng trước đây, lần này tôi biến đổi một chút. Từ dữ liệu cha, tôi trích xuất ngẫu nhiên 5 giá trị, sau đó tính giá trị trung bình cho dữ liệu mẫu.

```
import numpy as np
population = np.array([1, 5, 10, 11, 14, 15, 15, 16, 18, 18, 19, 20, 25, 30])
# Trich xuat ngau nhien
samples = np.random.choice(population, size=5)
print('population mean is {}'.format(np.mean(population)))
print('sample mean is {}'.format(np.mean(samples)))
print('samples: {}'.format(samples))
```

Nào bây giờ bạn hãy thực thi chạy code trên nhiều lần, và bạn sẽ thấy rằng giá trị trung bình của dữ liệu mẫu không trùng khớp với giá trị trung bình của dữ liệu cha. Tùy thuộc vào mẫu dữ liệu được trích xuất mà giá trị trung bình sẽ khác nhau, điều này là đương nhiên đúng không nào.

Nếu thử với vòng lặp với một vạn lần, chúng ta có thể tính giá trị trung bình của một vạn dữ liệu mẫu, giá trị trung bình của một vạn mẫu này sẽ như thế nào?

```
import numpy as np
population = np.array([1, 5, 10, 11, 14, 15, 15, 16, 18, 18, 19, 20, 25, 30])
# Trich xuat mau du lieu ngau nhien
##Gia tri trung binh cua du lieu cha
print('population mean is {}'.format(np.mean(population)))
## Nap gia tri ngau nhien cua cac mau du lieu vao list nay
sample_mean_list = []
count = 10000
## Thuc hien trich xuat mau du lieu 10000 lan
for i in range(count):
###Trich xuat mau du lieu ngau nhien
samples = np.random.choice(population, size=5)
###Nap gia tri trung binh cua mau du lieu vao list
sample_mean_list.append(np.mean(samples))
print('sample_mean_list mean is {}'.format(np.mean(sample_mean_list)))
```

Kết quả là:

```
population mean is 15.5
sample_mean_list mean is 15.5386
```

Sau khi trích xuất dữ liệu mẫu tới một vạn lần và tính giá trị trung bình của một vạn mẫu này được một vạn giá trị trung bình. Sau đó lại lấy giá trị trung bình của một vạn giá trị trung bình nói trên, ta được một giá trị rất gần với giá trị trung bình của dữ liệu cha. Nếu tăng lên là 10 vạn lần, 100 vạn lần trích xuất dữ liệu mẫu, thì giá trị này càng tiến gần tới giá trị trung bình của dữ liệu mẫu. Bạn có thể thay giá trị cho biến số `count`

trong code trên.
Như vậy giá trị trung bình của dữ liệu mẫu khi thì lớn hơn, khi thì nhỏ hơn giá trị trung bình của dữ liệu cha, nhưng nếu như chúng ta lặp vô hạn lần thì giá trị trung bình của dữ liệu mẫu $\bar{x}$ sẽ xấp xỉ bằng giá trị trung bình của dữ liệu cha μ.
Giống như nói ở trên, nếu có một giá trị ước lượng mà khi lấy giá trị trung bình các giá trị này thì nó trùng với tham số của dữ liệu cha, ta nói giá trị ước lượng này là bất thiên (unbiased), nó có nghĩa là không chệch, hay không thiên vị. Công cụ ước lượng như thế gọi là công cụ ước lượng bất thiên (unbiased estimator: ước lượng không thiên vị).
Tóm lại giá trị trung bình $\bar{x}$ của dữ liệu mẫu là công cụ ước lượng bất thiên cho giá trị trung bình μ của dữ liệu cha.
Đến đây các bạn cũng đã hiểu, cho dù chúng ta có lấy vô hạn số dữ liệu mẫu để tính giá trị trung bình độ phân tán s^2 của các dữ liệu mẫu đó thì giá trị này cũng không trùng khớp với độ phân tán của dữ liệu cha σ^2.
Nếu lấy dữ liệu mẫu vô hạn lần rồi tính giá trị trung bình của **độ phân tán bất thiên** thì nó sẽ xấp xỉ bằng với độ phân tán của dữ liệu cha σ^2. Vì vậy, độ phân tán bất thiên ở đây được định nghĩa là chia cho $n-1$.
Bạn hãy thử sửa code trên, thử tính một vạn lần độ phân tán bất thiên của dữ liệu mẫu rồi lấy giá trị trung bình của chúng, sau đó so sánh với độ phân tán của dữ liệu cha xem sao.

```
import numpy as np
from scipy import stats
population = np.array([1, 5, 10, 11, 14, 15, 15, 16, 18, 18, 19, 20, 25, 30])
# Trich xuat mau du lieu ngau nhien
##Gia tri trung binh cua du lieu cha
print('Do phan tan cua population la {}'.format(np.var(population)))
## Nap gia tri ngau nhien cua cac mau du lieu vao list nay
sample_mean_list = []
count = 10000
## Thuc hien trich xuat mau du lieu 10000 lan
for i in range(count):
###Trich xuat mau du lieu ngau nhien
samples = np.random.choice(population, size=5)
###Nap do phan tan bat thien cua mau du lieu vao list
sample_mean_list.append(stats.tvar(samples))
#Tinh gia tri trung binh cua 10000 gia tri do phan tan bat thien
print('sample_mean_list co do phan tan la {}'.format(np.mean(sample_mean_list)))
```

7.2 Cách nghĩ về giá trị kỳ vọng

Trước khi giải thích tại sao lại chia cho $n-1$, ta cần hiểu về giá trị kỳ vọng. Vậy giá trị kỳ vọng được hiểu như thế nào?

Giá trị kỳ vọng

Khi chúng ta mua vé số với số tiền 1000 yên, chúng ta kỳ vọng sẽ nhận được 400 yên. Điều này là vì thông thường những người mua vé số thì trung bình sẽ được nhận lại 400 yên. Tất nhiên cũng có một bộ phận trúng thưởng lớn và nhận về số tiền lớn nhưng nếu tính trung bình những người đã mua vé số thì số tiền trúng thưởng trung bình là 400 yên nếu bạn bỏ ra số tiền 1000 yên để mua vé số.

Nói tóm lại đây là giá trị trung bình dựa trên suy luận logic. Dù nói là trung bình thì nếu đã có khái niệm về xá suất, cái giá trị trung bình này được gọi là giá trị kỳ vọng **(expected value)**.

Giá trị biến động thay đổi do xác suất gọi là biến số xác suất. Ví dụ số tiền trúng xổ số, giá trị khi tung xúc sắc ... được gọi là biến số xác suất, ta ký hiệu là x, khi đó giá trị kỳ vọng được biểu diễn là $E(x)$.

Ví dụ một con xúc sắc có các điểm là 1,2,3,4,5,6, vậy giá trị kỳ vọng khi ta tung xúc sắc sẽ là:

$$E(x) = \frac{1+2+3+4+5+6}{6} = 3.5$$

Có thể thấy, giá trị trung bình của dữ liệu mẫu, hay độ phân tán của dữ liệu mẫu, chúng đều là các biến số xác suất, có giá trị thay đổi. Bởi vì khi ta lấy ngẫu nhiên dữ liệu mẫu từ dữ liệu cha, và tính toán giá trị trung bình thì đương nhiên giá trị trung bình này sẽ biến động thay đổi tùy thuộc vào mẫu dữ liệu, vì vậy có thể hiểu giá trị trung bình này là một biến xác suất.

Giá trị kỳ vọng $E(\bar{x})$ của giá trị trung bình $\bar{x}$ của dữ liệu mẫu là giá trị mà ta mong muốn nó trùng khớp với giá trị trung bình μ của dữ liệu cha. Bởi vì giá trị trung bình của dữ liệu mẫu là giá trị ước lượng bất thiên của giá trị trung bình của dữ liệu cha.

$$E(\bar{x}) = \mu$$

Ta có chứng minh đơn giản như sau:

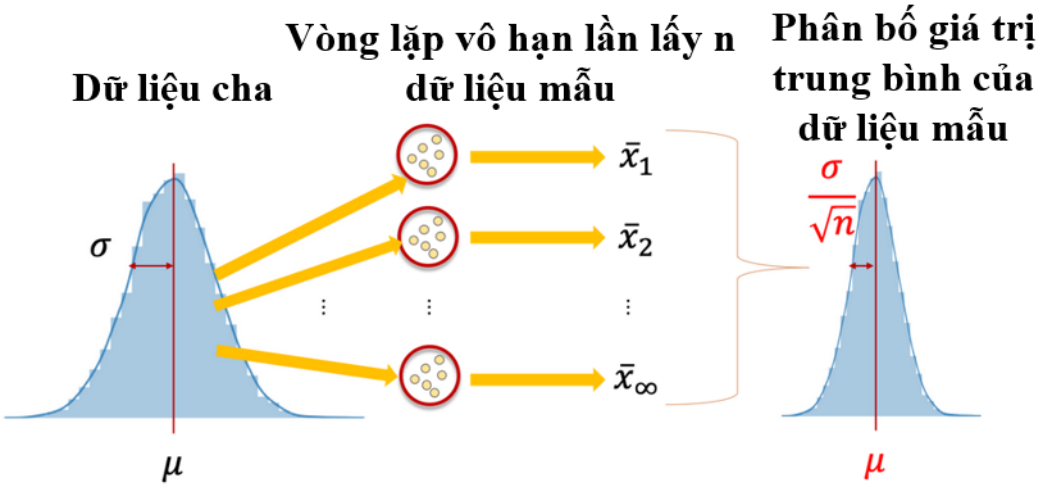
$$\begin{aligned} E(\bar{x}) &= E\left(\left[\frac{1}{n}(x_1 + x_2 + \cdots + x_n)\right]\right) \\ &= \frac{1}{n}E([(x_1 + x_2 + \cdots +_n)]) \\ &= \frac{1}{n}(E(x_1) + E(x_x) + \cdots + E(x_n)) \\ &= \frac{1}{n}n\mu = \mu. \end{aligned}$$

Dữ liệu có giá trị là x được lấy ngẫu nhiên từ dữ liệu cha thì đương nhiên giá trị kỳ vọng của x là $E(x)$ sẽ bằng với μ, vì vậy mà $E(x_1)$ hay $E(x_2)$ ...tất cả đều bằng μ.

Ngoài ra, dữ liệu mẫu có giá trị trung bình là $\bar{x}$, giá trị kỳ vọng của độ phân tán của nó là $E\left(\left[\frac{1}{n}\sum(\bar{x}-\mu)^2\right]\right)$ bằng $\frac{\sigma^2}{n}$.

Đây là phần rất quan trọng nên các bạn hãy ghi nhớ nhé. Khi $n = 1$ thì chúng ta chỉ lấy một dữ liệu mẫu, giá trị kỳ vọng chính là giá trị trung bình của dữ liệu mẫu. Độ phân tán khi đó bằng với độ phân tán của dữ liệu cha σ^2. Ngược lại khi n vô cùng lớn, mỗi lần lấy dữ liệu mẫu thì giá trị trung bình của dữ liệu mẫu lại gần với giá trị trung bình của dữ liệu cha μ nên độ phân tán trở nên nhỏ là điều chúng ta đã lý giải được.

Nhìn vào biểu đồ dưới đây sẽ dễ hiểu hơn. Ở đây không mô tả độ phân tán mà mô tả độ lệch chuẩn.



7.3 Lý do độ phân tán bất thiên chia cho $n-1$

Thành thực mà nói thì cái chứng minh này không quá quan trọng, tức là bạn không biết thì cũng không sao. Thông thường trong công việc làm về khoa học dữ liệu thì tính cần thiết phải hiểu chứng minh này là không có, tức là không biết chứng minh này thì cũng không sao đâu.
Do đó với người bình thường thì các bạn có thể lướt nhanh tới phần tổng kết của bài học này. Còn với người có hứng thú tìm hiểu, chúng ta sẽ tiếp tục cùng nhau với những suy luận như dưới đây.
Có thể nói rằng độ phân tán của dữ liệu cha σ^2 là giá trị kỳ vọng của giá trị trung bình của phép tính tổng bình phương độ sai khác từ các điểm dữ liệu trong dữ liệu cha tới giá trị trung bình của nó là μ.

$$\sigma^2 = E\left(\left[\frac{1}{n}\sum(x_i-\mu)^2\right]\right)$$

Dữ liệu mẫu có giá trị trung bình là $\bar{x}$ nên ta sẽ đưa nó vào biểu thức bên trái, biến đổi như sau:

$$
\begin{aligned}
\sigma^2 &= E\left[\frac{1}{n}\sum(x_i-\mu)^2\right] \\
&= E\left[\frac{1}{n}\sum\left(x_i \boxed{-\bar{x}+\bar{x}} -\mu\right)^2\right] \\
&= E\left[\frac{1}{n}\sum(x_i-\bar{x})^2 + \sum 2(x_i-\bar{x})(\bar{x}-\mu) + \sum(\bar{x}-\mu)^2\right] \\
&= E\left[\frac{1}{n}\sum(x_i-\bar{x})^2 + 2(\bar{x}-\mu)\underbrace{\sum(x_i-\bar{x})}_{\text{Tổng độ sai khác từ các điểm dữ liệu tới giá trị trung bình là 0}} + \sum(\bar{x}-\mu)^2\right] \\
&= E\left[\frac{1}{n}\sum(x_i-\bar{x})^2\right] + \underbrace{E\left[\frac{1}{n}\sum(\bar{x}-\mu)^2\right]}_{\text{Độ phân tán của giá trị trung bình của dữ liệu mẫu là } \frac{\sigma^2}{n}} \\
&= E\left[\frac{1}{n}\sum(x_i-\bar{x})^2\right] + \frac{\sigma^2}{n}.
\end{aligned}
$$

$(a+b)^2 = a^2 + 2ab + b^2$

Đến đây chúng ta chuyển vế và tính được như sau:

$$
\begin{aligned}
E\left[\frac{1}{n}\sum(x_i-\bar{x})^2\right] &= \sigma^2 - \frac{\sigma^2}{n} \\
&= \frac{n-1}{n}\sigma^2
\end{aligned}
$$

Công thức cuối cùng chính là điều mà tôi muốn nói. Khi chúng ta lấy vô số dữ liệu mẫu và tính toán giá trị kỳ vọng của độ phân tán dữ liệu mẫu $\frac{1}{n}\sum(x_i-\bar{x})^2$ thì nó nhỏ hơn độ phân tán dữ liệu cha σ^2 và có tỷ số là $\frac{n-1}{n}$.

Nói cách khác nếu lầy độ phân tán của dữ liệu mẫu $\frac{1}{n}\sum(x_i-\bar{x})^2$ nhân với $\frac{n}{n-1}$ ta sẽ thu đươc giá trị bằng với độ phân tán của dữ liệu cha σ^2.

Đến đây tôi nghĩ các bạn đã hiểu lý do mà độ phân tán bất thiên lại chia cho $n-1$.

Cách chứng minh này không khó, những người có hứng thú tìm hiểu hãy nhớ công thức này. Tuy nhiên cách chứng minh công thức này không quan trọng cho nên dù bạn không chứng minh được nó thì cũng không sao, các bạn vẫn có thể tiếp tục học khoa học dữ liệu.

7.4 Tổng kết

Bài học này tương đối dài nhưng có mấy điểm quan trọng dưới đây mà tôi muốn các bạn ghi nhớ.

- Giá trị ước lượng mà khi ta lấy giá trị khi lấy trung bình của nó ta thu được tham số của dữ liệu cha (là giá trị kỳ vọng), giá trị ước tính này được gọi là bất thiên và công cụ ước tính như vậy gọi là công cụ ước tính bất thiên.

- Độ phân tán bất thiên (Phương sai bất thiên) là một công cụ ước tính bất thiên của độ phân tán dữ liệu cha.

- Giá trị trung bình của dữ liệu mẫu là một công cụ ước tính bất thiên của giá trị trung bình của dữ liệu cha.
- Độ phân tán của giá trị trung bình của dữ liệu mẫu là $\frac{\sigma^2}{n}$.
- Đại lượng thống kê từ dữ liệu mẫu là một biến xác suất, giá trị cảu nó biến động có tính xác suất.

Mục nào ở trên cũng vô cùng quan trọng, đặc biệt là mục cuối cùng, giá trị ước tính thống kê từ dữ liệu mẫu là một điểm rất quan trọng khi chúng ta lý giải cũng như tìm hiểu về thống ke học. Tôi nghĩ nếu có điều kiện tôi sẽ bổ sung thêm những chú ý về nó.
Đến đây tôi đã trình bày về độ phân bố. Khái niệm phân tán bất thiên đã dẫn dắt chúng ta đi chệch khỏi câu chuyện về độ phân tán (phân bố) ban đầu.
Thực tế khi giải thihs về mức độ phân tán (phân bố), chúng ta thường sử dụng độ lệch chuẩn. Nhưng mà ngay cả khi bạn đươc cho biết rằng độ lệch chuẩn là từng này, tôi không nghĩ rằng bạn có thể biết các giá trị dữ liệu thay đổi bao nhiêu, như thế nào.
Trong bài tiếp theo, tôi ngĩ mình muốn nói về cách sử dụng độ phân tán (phân bố) này, nó được sử dụng như thế nào, đó sẽ là câu chuyện mà tôi muốn cùng các bạn thảo luận.

Bài 8

Làm thế nào để đọc độ phân tán từ độ lệch chuẩn?

Trong các bài học trước, chúng ta đã tìm hiểu về độ phân tán phân bố.
Bài học 4: Phạm vi, IQR(Phạm vi phần tư), QD(Độ lệch phần tư).
Bài học 5: Trung bình độ lệch, phân tán, độ lệch chuẩn.
Bài học 6, Bài học 7: Phân tán bất thiên.
Phạm vi là khoảng giá trị từ vị trí dữ liệu có giá trị nhỏ nhất tới vị trí dữ liệu có giá trị lớn nhất, do đó nó có nhược điểm là chứa cả những giá trị ngoại lệ, đó là những giá trị gây nhiễu không có ý nghĩa. Với IQR, QD có điểm mạnh là lược bớt những giá trị ngoại lệ nhưng vì nó không sử dụng toàn bộ dữ liệu trong tính toán do đó mà độ tin cậy thấp. Vì lý do đó có thể suy nghĩ tới trung bình độ lệch nhưng vì nó chứa dấu giá trị tuyệt đối nên rất khó để xử lý, cúng ta thực hiện bình phương chúng và từ đó có khái niệm về độ phân tán. Vì khi bình phương sẽ làm giá trị biến đổi do đó, do đó chúng ta lấy căn bậc hai của độ phân tán và lúc này chúng ta có giá trị có tên gọi là độ lệch chuẩn. Trường hợp chúng ta ước lượng độ phân tán của dữ liệu cha, chúng ta có phân tán bán bất thiên.

Xâu chuỗi các sự việc như trên chúng ta sẽ thấy kiến thức trở nên dễ nhớ hơn.
Chúng ta đã nói rất nhiều về độ phân bố thế nhưng quan trọng nhất đó là độ lệch chuẩn. Tôi nghĩ rằng trong hầu hết các lý luận thì độ lệch chuẩn đều được sử dụng để biểu thị về độ phân tán, phân bố của dữ liệu.

8.1 Có bao nhiêu dữ liệu nằm trong khoảng: Trung bình $\pm$ độ lệch chuẩn

Vì độ lệch chuẩn là $\cdots$ cho nên sẽ có chừng này dữ liệu nằm trong khoảng từ giá trị trung bình $\pm$ độ lệch chuẩn. Nếu như hiểu được điều này thì ta có thể hiểu được phân bố của dữ liệu, đúng không nào?
Ví dụ, điểm trung bình là 50 điểm, độ lệch chuẩn là 10 điểm, như vậy nói về độ phân bố của dữ liêu, giả sử trong khoảng từ 50 ± 10 điểm, thì có bao nhiêu dữ liệu nằm trong khoảng này? Nếu như hiểu được điều này thì bạn có thể nắm bắt được dữ liệu được phân

tán như thế nào rồi đúng không?

Trong thực tế, điều này phụ thuộc vào sự phân bố của dữ liệu, vì vậy không thể tính toán chính xác tỷ lệ dữ liệu có trong khoảng từ giá trị trung bình ± độ lệch chuẩn.

Tính toán chính xác có thể là không cần thiết, ở đây tôi đưa ra giá trị mang tính cảm quan, trong đó $\bar{x}$ là giá trị trung bình, s là độ lệch chuẩn, khi đó:

Nằm trong phạm vi $\bar{x} \pm s$ ước chừng có $\frac{2}{3}$ dữ liệu.

Nằm trong phạm vi $\bar{x} \pm 2s$ ước chừng có 95% dữ liệu.

Nằm trong phạm vi $\bar{x} \pm 3s$ ước chừng có $99\% \sim 100\%$ dữ liệu.

Như vậy có ba trường hợp mà tôi đã nêu ở trên, điều này có nghĩa là gì? Ví dụ trong một lớp học, kết quả bài kiểm tra của học sinh có điểm trung bình là 50 điểm và độ lêch chuẩn là 10 điểm. Khi đó ta dự đoán có khoảng 67% học sinh có số điểm nằm trong phạm vi 50 ± 10 điểm.

Tóm lại, trong trường hợp này (điểm trung bình 50 điểm, độ lệch chuẩn là 10 điểm), có lẽ không có ai có điểm 0 và cũng không có ai có số điểm là 100.

Nó khá tiện lợi phải không? Tuy nhiên điều này không phải lúc nào cũng chính xác. Hãy thử nghiệm với các số ngẫu nhiên với đoạn code dưới đây.

```
import numpy as np
#Lay ngau nhien cac so co gia tri trong khoang 0~1
randoms = np.random.rand(1000)
mean = np.mean(randoms)
std = np.std(randoms)
#Dem xem co bao nhieu gia tri nam trong khoang tu Gia Tri Trung Binh cong tru Do Lech
    Chuan
count = 0
coef = 1
thresh = coef * std
for num in randoms:
if num > mean - thresh and num < mean + thresh:
count += 1
print('{}% of the numbers are included within mean ± {} std'.format(int(count/len(
    randoms)*100), coef))
```

Kết quả:

```
57% of the numbers are included within mean ± 1 std
```

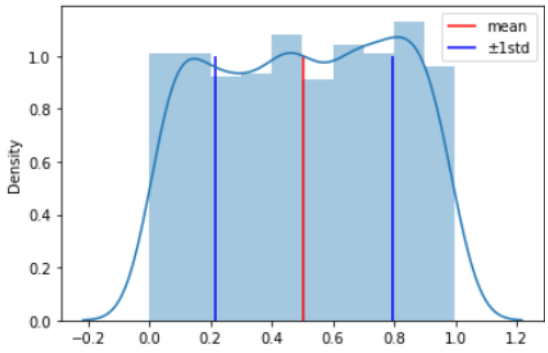
Chương trình trên không khó, các bạn hãy tự thử tự mình code xem sao nhé. `np.random.rand(1000)` sẽ tạo ra 1000 số có giá trị trong khoảng $0 \sim 1$, sau đó ta đếm xem trong phạm vi từ giá trị trung bình của các số ngẫu nhiên trên ± độ lệch chuẩn, có bao nhiêu số? Tôi nghĩ có thể viết code với hiệu suất cao hơn nhưng ở đây tôi viết code sao cho dễ hiểu nên tập trung vào việc sử dụng vòng lặp.

Dù chạy code nhiều lần thì có các kết quả khác nhau nhưng nhìn chung kết quả đều xoay quanh giá trị 57%.

Chúng ta thử vẽ đồ thị cho dễ hình dung, sau đây tôi sẽ sử dụng thư viện `seaborn` và `mathplotlib` và thực hiện code như sau:

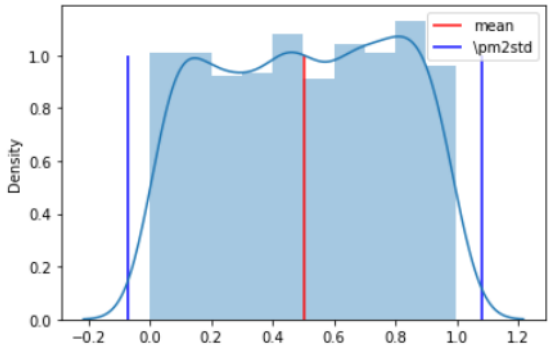
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

sns.distplot(randoms)
```

```
plt.vlines(mean, 0, 1, 'r', label='mean')
plt.vlines(mean+coef*std, 0, 1, 'b', label='±{}std'.format(coef))
plt.vlines(mean-coef*std, 0, 1, 'b')
plt.legend()
```



Nhìn vào biểu đồ trên ta thấy rằng dữ liệu nằm trong vùng hai đường kẻ màu xanh chiếm 57%. Biến số `coef` (hệ số của độ lệch chuẩn (coefficient) có tên viết tắt là coef) nếu thay đổi giá trị `coef=2` thì kết quả sẽ như thế nào, các bạn thử xem sao nhé. Chúng ta thấy rằng khi `coef=2` thì toàn bộ dữ liệu sẽ nằm trong hai đường kẻ màu xanh.

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
coef = 2
sns.distplot(randoms)
plt.vlines(mean, 0, 1, 'r', label='mean')
plt.vlines(mean+coef*std, 0, 1, 'b', label='\pm{}std'.format(coef))
plt.vlines(mean-coef*std, 0, 1, 'b')
plt.legend()
```



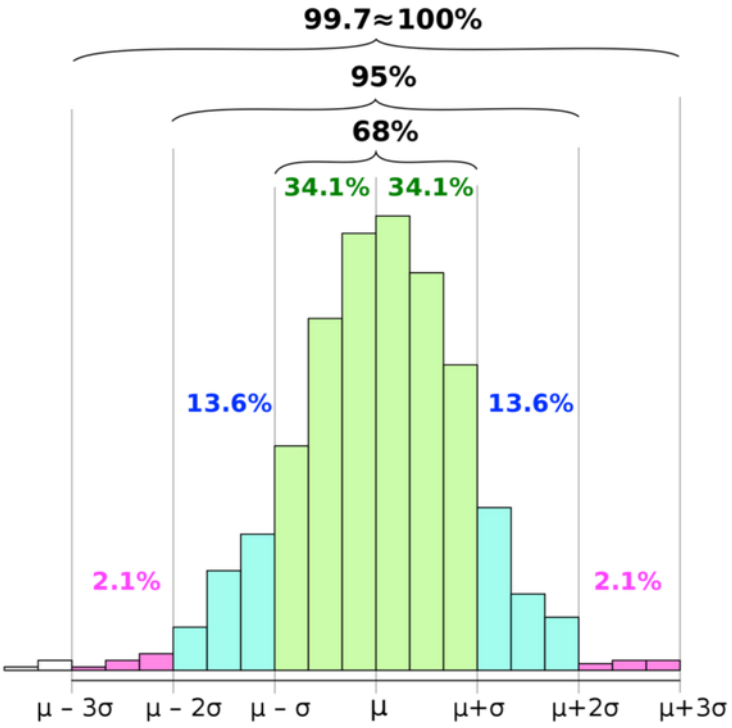
Như vậy giá trị cảm quan mà tôi nói ở trên trong trường hợp này không còn đúng nữa. Tôi đã nói rằng " Nằm trong phạm vi $\bar{x} \pm s$ ước chừng có $\frac{2}{3}$ dữ liệu". Và còn nói " Nằm trong phạm vi $\bar{x} \pm 2s$ ước chừng có 95% dữ liệu". Nhưng điều ấy đã không đúng trong ví dụ mà chúng ta vừa mới thực nghiệm.

Do đó hãy nhớ rằng giá trị cảm quan này không đúng hoàn toàn tùy thuộc vào sự phân bố của dữ liệu.

8.2 Nên nhớ về phân bố chuẩn

Rút cuộc sẽ chẳng có ý nghĩa gì nếu những điều chúng ta suy đoán thay đổi tùy thuộc vào phân bố của dữ liệu. Đây là điều mà có lẽ các bạn đang nghĩ tới. Nhưng không hẳn là không có ý nghĩa. Trước tiên cần biết rằng độ lệch chuẩn không phỉa là thứ hoàn mỹ để chúng ta có thể nắm rõ về dữ liệu. Tuy nhiên cái cảm giác rằng nằm trong phạm vi $\bar{x} \pm 3s$ ước chừng có $99\% \sim 100\%$ dữ liệu là rất tốt đấy chứ.
Điều thứ hai, đúng là nó không hoàn toàn tuân theo các giá trị cảm quan như đã thấy ở ví dụ thực nghiệm vừa qua, nhưng trong thống kê, việc giả định rằng dữ liệu phân bố như $\cdots$ vẫn thường xảy ra.Trong hầu hết các trường hợp chúng ta giả định một phân bố, phân bố giả định đó được gọi là phân bố chuẩn hay một số tài liệu khác gọi là phân phối chuẩn. Nếu dịch đúng nghĩa âm hán việt, thì người ta gọi phân bố chuẩn là phân bố chính quy.
Tôi sẽ viết một bài bình luận về phân bố chuẩn một lần nữa, nhưng nói tóm lại, đó là phân phối rất cơ bản, là một phân phối tiêu biểu trong thống kê học, nó rất quan trọng, bởi vì nhiều sự kiện tồn tại trong tự nhiên tuân theo phân bố chuẩn này.

Một phân bố được gọi là phân bố chuẩn nếu như có khoảng 68% dữ liệu nằm trong phạm vi giá trị trung bình ± độ lệch chuẩn. Và ước chừng có 95% dữ liệu nằm trong phạm vi giá trị trung bình ± 2 * độ lệch chuẩn. Và 99.7% dữ liệu sẽ nằm trong phạm vi giá trị trung bình ± 3 * độ lệch chuẩn.
Luật [68-95-99.7] đươc giới thiệu trong bài học số 9 của khóa học Python cho Data Science do diễn đàn tuhocvba.net tổ chức.



Quả nhiên để nhớ chi tiết những con số này thật là khó, nhưng cũng không cần thiết phải nhớ chi tiết như vậy. Bạn chỉ cần nhớ một cách đại khái $\left[68\%, 95\%\text{khoảng toàn bộ dữ liệu}\right]$

là được rồi.

8.3 Với phân bố chuẩn, 95% dữ liệu nằm trong phạm vi Giá Trị Trung Bình ±1.96 *Độ Lệch Chuẩn

Tôi muốn các bạn hãy nhớ điều này, đó là với phân bố chuẩn, 95% dữ liệu nằm trong phạm vi Giá Trị Trung Bình ±1.96 *Độ Lệch Chuẩn.
Có bạn sẽ thắc mắc, vậy không phải là 95% dữ liệu sẽ nằm trong phạm vi Giá Trị Trung Bình± Độ Lệch Chuẩn hay sao?
Không phải 95% dữ liệu sẽ nằm vừa vặn trong phạm vi Giá Trị Trung Bình±Độ Lệch Chuẩn, theo wikipedia thì **ước chừng** có 95%, chính xác hơn phải nói là 95.4%. Nếu bạn có độ lệch chuẩn trong tay và tự hỏi rằng dữ liệu được phân bố như thế nào, khi đó chỉ cần nghĩ tới quy tắc $[68\%, 95\%, 99.7\%]$ thì cũng không sao, không phải lúc nào ta cũng có hứng thú quan tâm tới phân bố của dữ liệu, trong nhiều trường hợp chúng ta quan tâm đến đường ranh giới nào của phân phối chuẩn đạt 95%.
Vì vậy các bạn hãy nhớ lấy số 1.96 vì nó sẽ được sử dụng thường xuyên.

Bổ sung

Chúng ta có những đường ranh giới sử dụng hệ số khác ±1.96 nhưng bạn chỉ cần nhớ một hệ số, đối với các hệ số khác bạn có thể tra cứu. Tôi không nghĩ các bạn cần phải ghi nhớ. Trong thời đại này, nếu có thể tìm kiếm ra là được rồi. Tuy nhiên chúng ta không thể tìm hiểu những thứ mà chúng ta hoàn toàn không có khái niệm về nó. Do đó chỉ cần nhớ ±1.96 và 95% vì nó thường được sử dụng nhất.

8.4 Tổng kết

Lần này ta đã sử dụng độ lệch chuẩn và đã hiểu dữ liệu được phân bố như thế nào. Từ bây giờ khi có các thông tin này, chúng ta có thể hình dung được lượng dữ liệu bao nhiêu được phân bố như thế nào. Nó được kiểm tra bằng cách tính toán từ giá trị trung bình tới bao nhiêu lần độ lệch chuẩn, tương ứng với khoảng cách đó sẽ có bao nhiêu dữ liệu được phân bố. Điều này thật là tiện lợi phải không nào.

- Thông thường trong phạm vi $\bar{x} \pm s$ ước chừng có $\frac{2}{3}$ dữ liệu, trong phạm vi $\bar{x} \pm 2s$ ước chừng có 95% dữ liệu, trong phạm vi $\bar{x} \pm 3s$ ước chừng có $99\% \sim 100\%$ dữ liệu được phân bố.

- Điều này không phải lúc nào cũng đúng, nhưng có thể xem xét nó ở một mức độ nào đó.

- Trong phân bố chuẩn có luật $[68 - 95 - 99.7]$. Bạn không cần phải nhớ chi tiết các con số này.

- Đường ranh giới mà ở đó có 95% dữ liệu được phân bố là Giá Trị Trung Bình ±1.96*Độ lệch chuẩn. Hệ số 1.96 được sử dụng nhiều trong thống kê học, hãy ghi nhớ lấy nó.

Trong thống kê, không có nhiều con số để nhớ, nhưng 1.96 là một trong số ít những con số đáng nhớ, vì vậy hãy ghi nhớ nó.
Các bạn có thể vận dụng lý thuyết này để kiểm chứng rất nhiều vấn đề trong thực tế. Chẳng hạn trong một trường học có bài kiểm tra tiếng anh, và một trường học khác cũng có bài kiểm tra tiếng anh. Nếu chúng ta chỉ nhìn vào điểm số thì không thể so sánh thành tích của trường nào tốt hơn. Tuy nhiên nếu sử dụng độ lệch chuẩn, và ở mỗi trường chúng ta tính toán các vị trí, từ đó có thể so sánh tương đối thành tích học sinh của mỗi trường với nhau. Tất nhiên mỗi trường thì trình độ của các học sinh không giống nhau hoàn toàn, do đó cũng không thể nói đây là phép so sánh hoàn mỹ nhưng vẫn có thể mang một ý nghĩa nào đó.
Những người có thể ứng dụng điều này đương nhiên đã biết về độ lệch (Thiên Sai Trị). Số liệu thống kê được sử dụng rất nhiều trong cuộc sống xung quanh ta.

Bài 9

Rất quan trọng! Chuẩn hóa và trị số lệch là gì? Tính điểm z và tính điểm T

Ở bài học trước ta đã sử dụng độ lệch chuẩn khi nói về mức độ phân bố, đây là một khái niệm vô cùng quan trọng vì nó giúp chúng ta phán đoán được ở trong phạm vi nào đó sẽ có bao nhiêu dữ liệu được phân bố. Chúng ta có thể so sánh các nhóm dữ liệu khác nhau khi sử dụng trung bình cộng và độ lệch chuẩn. Ví dụ chúng ta có hai ngôi trường cùng cho học sinh làm bài kiểm tra. Nếu chỉ nhìn vào điểm số, chúng ta không thể so sánh kết quả trường nào thì tốt hơn. Khi sử dụng trung bình cộng và độ lệch chuẩn, chúng ta có thể tính toán và so sánh thành tích điểm số ở một vị trí nào đó trong bảng phân bố dữ liệu.
Trong bài học này, chúng ta sẽ có những khái niệm rất quan trọng và tôi muốn các bạn sẽ ghi nhớ.

9.1 So sánh giữa các nhóm có dữ liệu khác nhau bằng cách tính điểm z.

Tính điểm z là gì? Hẳn là điều các bạn đang thắc mắc.
Trong bài học trước, ta có thể xác định vị trí của dữ liệu nằm trong phạm vi nào, từ độ trung bình tới bao nhiêu lần độ lệch chuẩn, từ đó có thể biết được phạm vi đó có mức độ phân bố dữ liệu là bao nhiêu phần trăm.
Và như tôi đã nói, lý thuyết này rất quan trọng, nó giúp chúng ta có thể so sánh hai dữ liệu khác nhau. Ví dụ, bạn Bình ở trường A có điểm tiếng anh là 40, bạn An ở trường B có điểm tiếng anh là 60. Nếu chỉ nhìn vào điểm kiểm tra như vậy thì không thể so sánh được mức độ thành thạo tiếng anh. Chẳng hạn trường A ra bài kiểm tra quá khó, mặc dù Bình được 40 điểm nhưng có thể đó là học sinh tốp đầu của trường A.
Vậy thì ta phải tính toán với mỗi bạn Bình và bạn An thì các bạn ấy phải tính toán điểm của mình nằm trong phạm vi nào từ giá trị trung bình tới bao nhiêu lần độ lệch chuẩn thì các bạn ấy mới biết trình độ của mình ở mức nào trong trường mình.

Từ vấn đề đặt ra đó, chúng ta có thể đề xuất công thức sau đây, ta gọi nó là công thức tính điểm z.

$$z = \frac{x - \bar{x}}{s}$$

trong đó $\bar{x}$ là giá trị trung bình, s là độ lệch chuẩn.
Ví dụ, chúng ta có hai lớp học và có điểm môn thi Tiếng Anh và môn Toán như sau:

Lớp 1	A	B	C	D	E	Trung Bình	Độ lệch chuẩn
Tiếng Anh	40	30	80	70	69	56	18.5472
Lớp 2	F	G	H	I	J	-	-
Toán	30	50	40	30	20	34	10.198

Vậy câu hỏi đặt ra, ai là người giỏi nhất? Nếu nhìn vào điểm số có lẽ các bạn sẽ khẳng định C là người giỏi nhất, thế nhưng điều đó có đúng hay không?
Dữ liệu khác nhau mà so sánh thông thường sẽ không có ý nghĩa gì. Vậy thì muốn so sánh ta phải chuẩn hóa.
Chuẩn hóa ở đây có nghĩa là ta phải thực hiện biến đổi để dữ liệu có giá trị trung bình là 0 và độ lệch chuẩn là 1.
Công thức ở trên có nghĩa là: Điểm mới = Điểm thực tế−Điểm trung bình÷Độ lệch chuẩn
Như vậy áp dụng công thức trên ta sẽ có dữ liệu mới sau khi chuẩn hóa như sau:

Lớp 1	A	B	C	D	E
Tiếng Anh	−0.8626	−1.4018	1.2939	0.7548	0.2156
Lớp 2	F	G	H	I	J
Toán	−0.39223	1.5689	0.5883	−0.3922	−1.3728

Các bạn cũng có thể tính toán bằng code Python như sau:

```
import numpy as np
data_tienganh = [40,30,80,70,60]
mean_tienganh = np.mean(data_tienganh)
std = np.std(data_tienganh)
#Chuan hoa
z = (data_tienganh - mean_tienganh) / std
print('standardized data tieng anh(z): {}'.format(z))
print('mean mean_tienganh: {:.2f}'.format(np.mean(z)))
print('std mean_tienganh: {}'.format(np.std(z)))
```

Kết quả cũng tương tự như trên:

```
standardized data tieng anh(z): [-0.86266219 -1.40182605  1.29399328  0.75482941
    0.21566555]
mean mean_tienganh: -0.00
std mean_tienganh: 1.0
```

Như vậy ta thấy $C = -1.29$ trong khi đó $G = 1.56$, cho nên so sánh một cách tương đối ta thấy rằng G là người đạt điểm cao nhất.

Chuẩn hóa (standardize) như thế nào?

Mục tiêu chuẩn hóa là cố gắng làm sao để dữ liệu có giá trị trung bình là 0.
Và một mục tiêu nữa đó là làm sao để độ phân tán (đồng thời cả độ lệch chuẩn) có giá

trị là 1.

▌Tính chính xác ra sao?

Điều lo ngại là mức độ tin cậy của dữ liệu sau khi đã được chuẩn hóa. Thật sự thì cách so sánh nói trên có ổn không?

Hãy cẩn thận trong những trường hợp sau đây.

▌Quy mô dữ liệu thống kê quá nhỏ

Đương nhiên nhưng vẫn cần nói, nếu mẫu số quá nhỏ thì sẽ không còn tính chính xác.

Chẳng hạn ta có ba người A có số điểm 20, B có điểm 50, C có điểm 80.

Giá trị trung bình là 50.

Trong trường hợp này thì điểm số chuẩn hóa của C trở nên ít đi đáng kể. Giả sử bài kiểm tra này là rất khó, thậm chí B và C là những chuyên gia trong lĩnh vực này.

Khi đó nếu so sánh với kết quả của những người làm bài kiểm tra thông thường, thậm chí C còn bị đánh giá là kém.

Khi quy mô dữ liệu thống kê quá ít thì các giá trị cực đoan sẽ xuất hiện.

▌Không phải là phân bố chuẩn

Giả sử ta có 3 người đạt số điểm là 0, và có 3 người đạt số điểm là 100. Trong trường hợp này điểm trung bình cũng là 50. Với thí nghiệm như thế này ta cũng thấy rằng việc chuẩn hóa cũng không có ý nghĩa gì. Các phân bố mà dữ liệu tập trung vào hai đầu mút, hoặc quá lớn hay quá bé, thì độ chính xác cũng không có.

Chuẩn hóa được sử dụng rất nhiều trong Machine Learning. Có nhiều trường hợp không thể xử lý được nếu dữ liệu để nguyên như ban đầu mà không thực hiện chuẩn hóa. Khi đào tạo một mô hình bằng Machine Learning, dữ liệu thường được chuẩn hóa trước đó, ta gọi là tiền xử lý (preprocessing).

```
from sklearn.preprocessing import StandardScaler
data = np.array([30,50,40,30,20])
print('data shape: {}'.format(data.shape))
data = np.expand_dims(data, axis=-1)
print('reshaped data shape: {}'.format(data.shape))
# Instance creation
scaler = StandardScaler()
# Doi so cho fit_transform phai la mot mang hai chieu
scaled = scaler.fit_transform(data)
print(scaled)
```

Kết quả:

```
data shape: (5,)
reshaped data shape: (5, 1)
[[-0.39223227]
[ 1.56892908]
[ 0.58834841]
[-0.39223227]
[-1.37281295]]
```

Cách tính ở đây cũng sẽ đưa ra kết quả tính toán giống với code đã giới thiệu ở trên.

Lần này tôi đã sử dụng `sklearn.preprocessing.StandardScaler` , nó được gọi là một lớp (Class) và được viết theo cách hướng đối tượng.

`scaler = StandardScaler()` khi được thực thi, nó sẽ tạo ra các mô hình (instance) dựa

trên Class `StandardScaler` .
Có lẽ các bạn chưa quen với hướng đối tượng, nói một cách đơn giản, từ cái sơ đồ thiết kế đó sẽ tạo ra một thứ (instance) được gọi là `scaler`. Dương nhiên cái `scaler` này có chức năng chuẩn hóa bằng cách gọi hàm `.fit_transform()`. Tôi nghĩ code không khó, các bạn hãy làm quen với code như vậy.
`.fit_transform()` sẽ chuẩn hóa đối số đầu vào mà mảng `ndarray`. Và vì nó chỉ tiếp nhận mảng hai chiều cho nên trước đó chúng ta thông qua `np.expand_dims()` để chuyển thành mảng hai chiều. Về điều này đã được nêu chi tiết trong bài học số 9 của khóa học Data Science. Bạn có thể vừa xem tài liệu tham khảo vừa tiếp tục theo dõi bài học này. Bạn cũng không cần phải nhớ, vì khi thực thi sẽ xuất hiện lỗi do đầu vào không phải là mảng hai chiều, và khi đó bạn sẽ phải xoay sở để chuyển mảng một chiều thành mảng hai chiều, bằng cách này hay cách khác bạn cũng sẽ khắc phục được vì google sẽ giúp bạn tìm ra giải pháp. `sklearn.preprocessing.StandardScaler` được sử dụng rất nhiều, từ giờ các bạn hãy nhớ nó nhé.

9.2 Trị số lệch là gì?

Kết quả kỳ thi hôm trước thế nào?

So với lần trước, điểm tiếng anh cao hơn 10 điểm.

Chúc mừng nhé. Trị số lệch thì thế nào?

À, so với kỳ thi trước thì giảm một chút.

Tôi thì điểm toán giảm nhưng trị số lệch thì tăng. Vậy thì sẽ thế nào nhỉ?

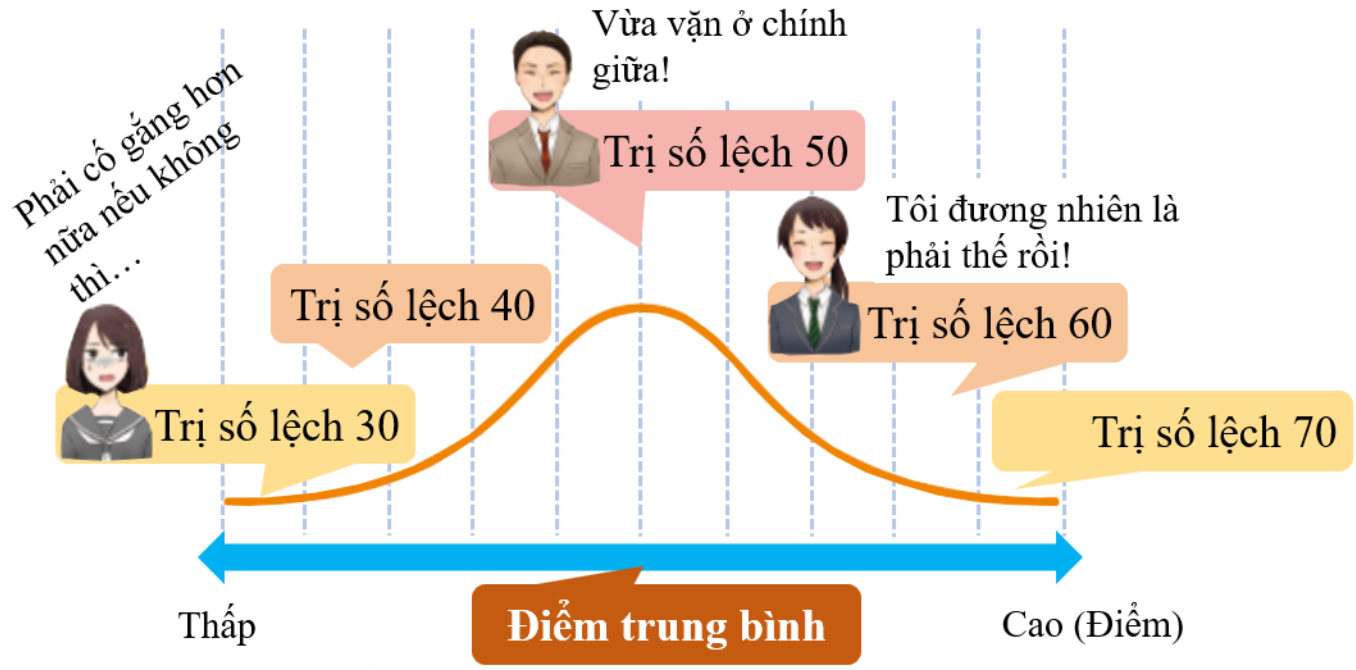
Trị số lệch (giá trị độ lệch) có liên quan rất lớn đến điểm trung bình của bài kiểm tra và sự phân bố điểm của tất cả các thí sinh tham dự kỳ thi. Vì điểm trung bình được biểu thị bằng 50 nên nó phụ thuộc vào phân bố điểm của toàn bộ nhóm.

Phức tạp quá nhỉ!

Hãy cùng suy nghĩ về ý nghĩa của trị số lệch và cách sử dụng nó.

Ví dụ, có hai kỳ thi diễn ra và cả hai kỳ thi tôi đều có 80 điểm. Tuy nhiên mức độ khó của mỗi kỳ thi là khác nhau, do đó điểm số trung bình của mỗi kỳ thi cũng khác nhau. Vậy thì kết quả của kỳ thi lần một và của lần hai, kết quả nào mới là tốt khi mà tôi có cùng số điểm là 80 ? Rõ ràng nếu chỉ dựa vào điểm số thì chúng ta không thể đưa ra phán đoán chính xác. Khi đó, chúng ta cần tới một khái niệm mới, đó là **trị số lệch**. Bằng cách biến đổi dữ liệu để giá trị trung bình có giá trị là 50 và có độ lệch chuẩn là

10. Điểm số mới sau khi đi qua bộ biến đổi như vậy gọi là **trị số lệch**. Trị số lệch càng cao có nghĩa rằng điểm số của tôi có thứ hạng cao. Và ngược lại, trị số lệch thấp có ngĩa rằng thứ hạng của tôi trong số những người tham dự kỳ thi là thứ hạng thấp.



Trị số lệch được tính như thế nào?

Bước 1: Chuyển điểm số về cách tính điểm z. Cách tính này sẽ tạo ra điểm số mới có giá trị trung bình là 0 và có độ lệch chuẩn là 1.

Bước 2: Trị số lệch = Điểm z $*$ 10 + 50 Như vậy điểm số mới chính là trị số lệch, chúng sẽ có giá trị trung bình là 50 và có độ lệch chuẩn là 10.

```
from sklearn.preprocessing import StandardScaler
data = np.array([30,50,40,30,20])
print('data shape: {}'.format(data.shape))
data = np.expand_dims(data, axis=-1)
print('reshaped data shape: {}'.format(data.shape))
# Instance creation
scaler = StandardScaler()
# Doi so cho fit_transform phai la mot mang hai chieu
scaled = scaler.fit_transform(data)
# Tinh diem T
scaled = scaled*10+50
print(f"mean of scaled {np.mean(scaled)}")
print(f"std of scaled {np.std(scaled)}")
print(scaled)
```

Kết quả:

```
data shape: (5,)
reshaped data shape: (5, 1)
mean of scaled 49.99999999999999
std of scaled 9.999999999999998
[[46.0776773 ]
[65.68929081]
[55.88348405]
[46.0776773 ]
[36.27187054]]
```

Giá trị trung bình khi tính sang trị số lệch sẽ là bao nhiêu? Trị số lệch có giá trị cao nhất là bao nhiêu?

Do việc chuẩn hóa để tính điểm T là làm sao để giá trị trung bình là 50 do đó nếu điểm của bạn ngoài thực tế có giá trị bằng số điểm trung bình thì khi chuyển sang cách tính điểm T, trị số lệch của bạn khi đó nhất định sẽ là 50.

Ngoài ra trị số lệch ở một kỳ thi thông thường sẽ nằm trong phạm vi từ $25 \sim 75$ nhưng khi tính toán sang điểm T, vẫn có trường hợp trị số lệch âm hoặc trị số lệch lớn hơn 100. Tôi lấy ví dụ trong một kỳ thi có 100 người dự thi trong đó 99 người có số điểm là 0, và chỉ có một người có số điểm là 100, khi đó nếu tính điểm T thì những người điểm 0 sẽ có trị số lệch là -49 và người có điểm 100 có trị số lệch là 149.5.

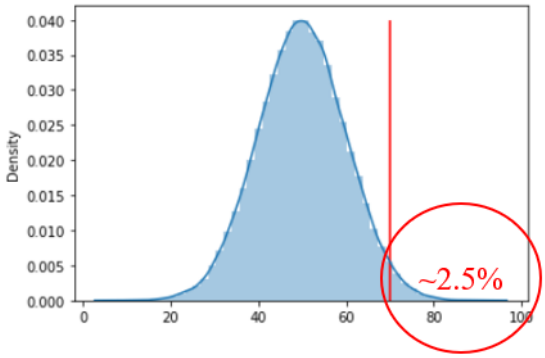
Những trường hợp cực đoan, khi mà dữ liệu tập trung phân bố vào hai đầu mút, hoặc nhỏ nhất, hoặc lớn nhất, trường hợp như thế thực sự là không thể suy nghĩ được, và kết quả khi tính trị số lệch cũng không có ý nghĩa gì.

Cách xem trị số lệch và tỷ lệ của nó

Bảng dưới đây mô tả mối quan hệ về thứ hạng với trị số lệch. Nếu trị số lệch là 75 thì nó ở vị trí 0.62% tính từ trên xuống. Điều đó tức là nếu có 1000 người dự thi thì nó sẽ có thức hạng trước hoặc sau thứ hạng thứ 6. Nếu trị số lệch là 35 thì thứ hạng tính từ trên xuống là 93.32% (tính từ dưới lên sẽ là 6.68%), tức là thứ hạng tính từ cao xuống thấp, nó sẽ ở vị trí trước hoặc sau thứ hạng thứ 933. Lợi điểm của trị số lệch là bạn có thể biết được vị trí năng lực học tập của mình trong nhóm bất kể điểm số hay thứ hạng.

Trị số lệch	Tỷ lệ từ trên xuống	Thứ hạng trong 1000 người
80	0.13%	Vị trí 1.3
75	0.62%	Vị trí 6.2
70	2.28%	Vị trí 22.8
65	6.68%	Vị trí 66.8
60	15.87%	Vị trí 158.7
55	30.85%	Vị trí 308.5
50	50.00%	Vị trí 500.0
45	69.15%	Vị trí 691.5
40	84.13%	Vị trí 841.3
35	93.32%	Vị trí 933.2
30	97.72%	Vị trí 977.2

Trong bài học trước tôi cũng đã nói rằng trong phạm vi Giá trị trung bình$\pm 2*$Độ lệch chuẩn sẽ có 95% dữ liệu, như vậy có thể tính toán được rằng trị số lệch 70 (khi tính sang điểm T thì độ lệch chuẩn là 10 do đó phạm vi dữ liệu lúc này nằm từ $30 \sim 70$) sẽ có khoảng 2.5% dữ liệu, bởi vì $100\% - 95\% = 5\%$, và vì bảng phân bố chuẩn đối xứng trái phải nên ta có con số 2.5%. Tức là người có trị số lệch 70 sẽ nằm ở tốp 25 người cao nhất nếu có 1000 người dự thi. Điều này cũng phù hợp với bảng trên.



Hình vẽ trên các bạn có thể vẽ bằng code như sau:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

samples = np.random.randn(100000)
tscore = samples*10 + 50
sns.distplot(tscore)
plt.vlines(70, 0,0.04, 'r')
```

Lợi điểm sử dụng trị số lệch

Thầy ơi, kết quả kỳ thi lần này, môn Toán em được 70 điểm và môn Tiếng Anh em được 90 điểm. Môn Toán khó nên điểm không tốt, tuy nhiên môn Tiếng Anh được điểm cao nên em rất vui.

Chúc mừng em, vậy trị số lệch thì thế nào?

Dù kỳ thi có điểm trung bình khác nhau, tuy nhiên sau khi chuẩn hóa và tính trị số lệch, chúng ta có thể so sánh điểm số của hai kỳ thi khác nhau. Đây là một lợi điểm của trị số lệch. Giống như bạn học sinh ở trên, tuy điểm Tiếng Anh và điểm Toán khác nhau, nhưng khi tính toán trị số lệch thì đều có giá trị là 62.2.

Cả hai môn đều có trị số lệch là 62.2 ạ.

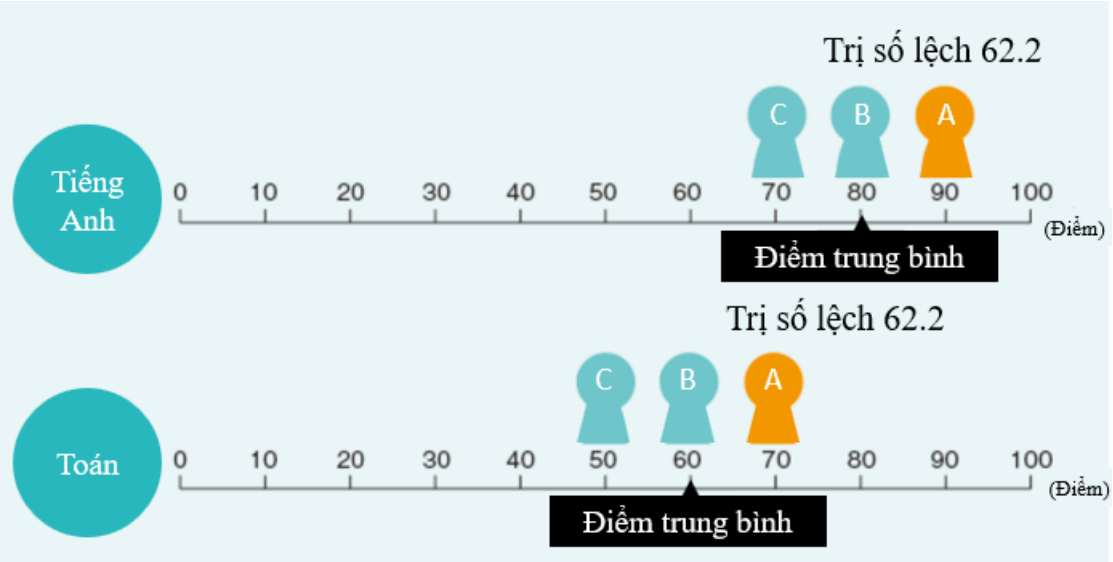
Môn Toán quả nhiên là khó nên điểm thấp hơn nhỉ.

Để dễ hình dung ta hãy xem bạn học sinh A:

Người dự thi	Tiếng Anh	Toán
A	90	70
B	80	60
C	70	50
Điểm trung bình	80	Vị trí 60
Trị số lệch của A	62.2	62.2

Điểm trung bình của môn Tiếng Anh là 80 và điểm trung bình của môn Toán là 60, có thể thấy vì môn Toán khó hơn môn Tiếng Anh, nên trị số lệch của cả hai môn thi đều có giá trị như nhau mặc dù điểm thi của bạn A ở môn Tiếng Anh là cao hơn môn Toán. Xét

về mặt thứ hạng thì không thay đổi. Như vậy có thể thấy, chúng ta không chỉ so sánh được điểm số hay thứ hạng trong một kỳ thi, mà còn có thể so sánh điểm của hai kỳ thi khác nhau.



Chú ý đến sự thay đổi trong điểm số

Để đánh giá kết quả bài thi một cách công bằng hơn, cần xem xét không chỉ sự chênh lệch so với điểm trung bình, mà cần quan tâm cả sự phân bố của điểm số. Ví dụ dưới đây là tóm tắt kết quả của năm người đã làm bài kiểm tra Tiếng Anh và Toán.

Người dự thi	Tiếng Anh	Toán
D	70	70
E	36	16
F	50	38
G	64	82
H	80	94
Điểm trung bình	60	60
Độ lệch chuẩn	15.4	28.8
Trị số lệch của D	56.5	53.5

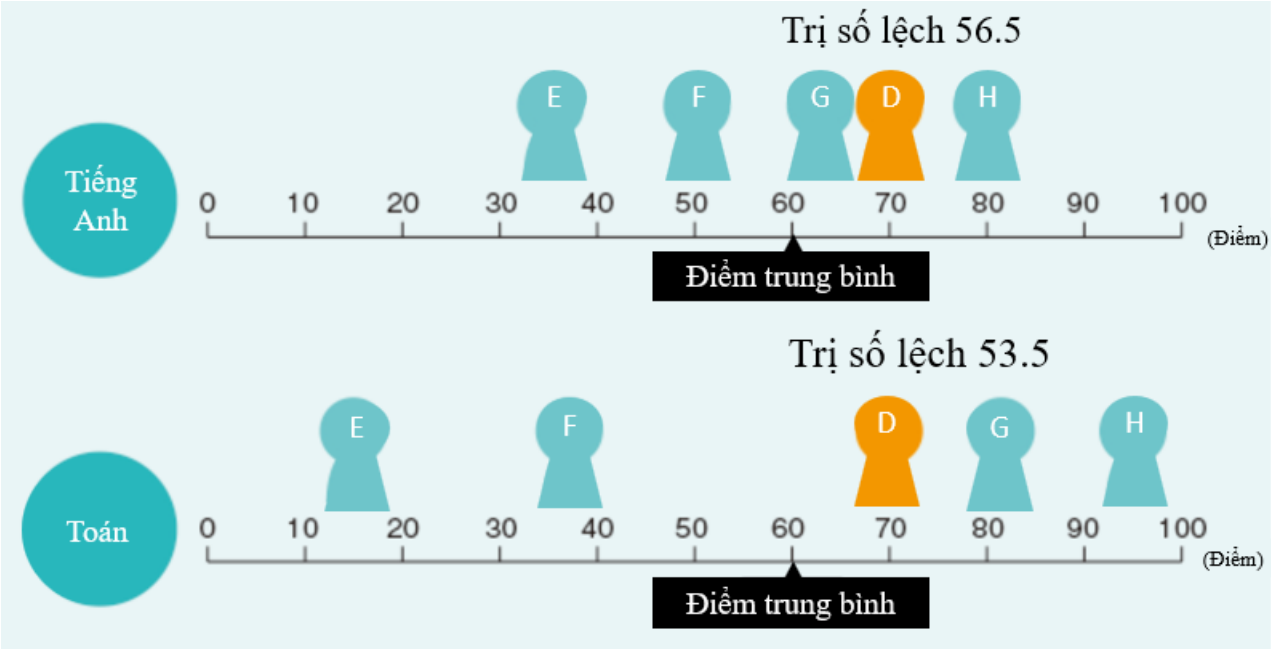
Điểm của bạn D ở cả môn Toán và Tiếng Anh đều là 70 điểm, và điểm trung bình của cả Toán và Tiếng Anh đều là 60 điểm. Tuy nhiên trong môn Tiếng Anh, 5 người có điểm số rất gần điểm số trung bình. Trong khi đó với môn Toán thì điểm của 5 người khác nhau rất nhiều.

Trong một kỳ thi nếu như phần lớn mọi người đều có số điểm tập trung quanh điểm số trung bình với mật độ dày đặc thì điều đó chứng tỏ việc có được điểm số vượt qua con số trung bình này là một việc khó khăn. Do đó nếu ai đó có điểm cao hơn mức trung bình thì trị số lệch sẽ trở nên rất lớn.

Ngược lại trong một kỳ thi mà điểm thi của nhiều người cách xa mốc điểm trung bình, khi đó dù điểm số cao hơn mức điểm trung bình thì trị số lệch hầu như không gia tăng.

Do đó để đánh giá công bằng thành tích trong một kỳ thi, cần xem xét cả hai yếu tố, độ lệch so với điểm số trung bình, và phân bố điểm.

Do đó, D có điểm số giống nhau trong cả hai kỳ thi mà các kỳ thi này đều có điểm trung bình giống nhau, tuy nhiên trị số lệch lại có sự khác nhau.



9.3 Tổng kết

Lần này tôi đã giải thích cho các bạn về cách tính điểm z và trị số lệch.
Nhờ tính toán từ điểm dữ liệu tới giá trị trung bình, khoảng cách đó bằng bao nhiêu lần độ lệch chuẩn, chúng ta có thể biết được phân bố của toàn bộ dữ liệu và vận dụng nó để tính được vị trí thứ hạng của điểm dữ liệu trong toàn bộ tổng thể dữ liệu, thông qua đó ta có thể so sánh được hai dữ liệu khác nhau cho dù chúng có độ trung bình cũng như độ lệch chuẩn khác nhau. Để có thể thực hiện được việc so sánh như thế chúng ta cần một bước biến đổi dữ liệu, công việc đó gọi là **tiêu chuẩn hóa** hoặc **chuẩn hóa**.

- Chuẩn hóa là biến đổi dữ liệu để giá trị trung bình là 0 và độ lệch chuẩn là 1, ta gọi đây là cách tính điểm z.
- Công thức chuẩn hóa là $z = \dfrac{x - \bar{x}}{s}$.
- Trị số lệch là giá trị mới của dữ liệu sau khi biến đổi dữ liệu ban đầu sao cho giá trị trung bình là 50 và có độ lệch chuẩn là 10.

Trị số lệch trong bài này chỉ mang tính giới thiệu, sau này trong thống kê học hay trong máy học thì nó không thực sự quan trọng.
Về cơ bản các bạn hãy nắm rõ cách tính điểm z. Từ công thức này ta có thể biến đổi dữ liệu sao cho nó có giá trị trung bình cũng như độ lệch chuẩn tùy ý mà ta mong muốn.

Phần I

Thuật Ngữ

STT	Tiếng Nhật	Cách đọc	Tiếng Anh	Tiếng Việt
1	分散	ブンサン	Variance	Phân tán
2	分布	ブンプ	Distribution	Phân bố
3	範囲	ハンイ	Range	Phạm vi
4	四分位数	シブンイスウ	P	Phần tư
5	四分位偏差	シブンイヘンサ	quartile deviation	Độ lệch phần tư
6	四分位範囲	シブンイハンイ	interquartile range: IQR	Phạm vi phần tư
7	四分位偏差	シブンイヘンサ	quartile deviation: QD	Độ lệch phần tư
8	平均偏差	ヘイキンヘンサ	mean deviation	Độ lệch trung bình
9	平均絶対偏差	ヘイキンゼッタイヘンサ	mean absolute deviation	Độ lệch trung bình tuyệt đối
10	標準偏差	ヒョウジュンヘンサ	standard deviation	Độ lệch chuẩn
11	不偏分散	フヘンブンサン	unbiased variance	Phân tán bất thiên
12	不偏	フヘン	unbiased	Bất thiên
13	不偏推定量	フヘンスイテイリョウ	unbiased estimator	Ước lượng bất thiên
14	期待値	キタイチ	expected value	Giá trị kỳ vọng
15	統計的記述(記述統計)	トウケイテキキジュツトウケイ	descriptive statistics	Thống kê mô tả
16	統計的推論(推測統計)	トウケイテキスイロンスイソクトウケイ	inferential statistics	Thống kê suy luận
17	母集団	ボシュウダン	population	Dữ liệu cha
18	標本	ヒョウホン	sample	Dữ liệu tiêu bản, dữ liệu mẫu
19	算術平均	サンジュツヘイキン	arithmetic mean	Trung bình cộng

STT	Tiếng Nhật	Cách đọc	Tiếng Anh	Tiếng Việt
20	幾何平均	G	geometric mean	Trung bình hình học
21	調和平均	G	harmonic mean	Trung bình điều hòa
22	偏差	G	deviation	Thiên sai, độ lệch
23	中央値	G	median	Trung vị
24	外れ値	G	outlier	Giá trị ngoại lệ
25	最頻値	G	mode	Tối tần trị
26	平方根	ヘイホウコン	-	Căn bậc hai
27	感覚値	カンカクチ	-	Giá trị cảm quan
28	境目	サカイメ	-	Đường ranh giới
29	偏差値	ヘンサチ	-	Trị số lệch