HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Nhập môn
Khoa học dữ liệu
(IT4142)

PGS.TS Thân Quang Khoát & PGS.TS Phạm Văn Hải
Team lecturers

ONE LOVE. ONE FUTURE.

2

Contents

Nội dung

1. Chọn loại biểu đồ phù hợp?

- Trực quan hoá dữ liệu là kỹ thuật cho phép truyền đạt những thông tin ẩn chứa trong dữ liệu thông qua biểu diễn trực quan
- Mục tiêu biến các tập dữ liệu lớn thành các biểu đồ trực quan cho phép hiểu rõ hơn về mối quan hệ phức tạp bên trong dữ liệu
- Việc lựa chọn loại biểu đồ nào để biểu diễn dữ liệu là rất quan trọng và nó phụ thuộc vào mục đích cũng như đặc tính của dữ liệu

What story do you want to tell?

- Hiểu tại sao chúng ta cần những loại biểu đồ khác nhau để trực quan dữ liệu rất quan trọng
  - Graphs
  - Plots
  - Maps
  - Diagrams
  - ...
- Các mục đích chính của trực quan hoá dữ liệu:
  - Mối quan hệ
  - Dữ liệu theo thời gian
  - Xếp hạng
  - Phân bố
  - So sánh

Mối quan hệ giữa các thuộc tính của tập dữ liệu

- Mỗi thuộc tính của dữ liệu = one variable
- Mục tiêu: hiển thị mối tương quan giữa các thuộc tính dữ liệu, sự phụ thuộc có thể có giữa chúng
- Khi đánh giá mối quan hệ giữa các tập dữ liệu chúng ta mong muốn hiểu được cách mà các tập dữ liệu này kết hợp hoặc tương tác với nhau
- Mối quan hệ đó có thể là positive hoặc negative:
  - Tuỳ thuộc vào các thuộc tính dữ liệu có thể hỗ trợ lẫn nhau hay hoạt động trái ngược nhau

Relationship

- 4 loại biểu đồ thường dùng:
  - Scatter plot
  - Histogram
  - Pair Plot
  - Heat map

Biểu diễn dữ liệu theo thời gian

- Mục tiêu: khám phá mối quan hệ giữa các thuộc tính dữ liệu để tìm ra xu hướng hoặc sự thay đổi theo thời gian
- Thời gian được biểu diễn như một loại thuộc tính liên kết giữa các biến nên về cơ bản mục đích vẫn là khám phá một loại quan hệ giữa các tập dữ liệu
- Các loại biểu đồ thường dùng:
  - Line chart
  - Area chart
  - Stack Area Chart
  - Area Chart Unstacked

Xếp hạng

- Goal: Biểu diễn mối quan hệ về mặt thứ tự giữa các thuộc tính dữ liệu
- Các loại biểu đồ thường dùng:
  - Vertical bar chart
  - Horizontal bar chart or Column Chart
  - Multi-set bar chart
  - Stack bar chart
  - Lollipop Chart

Phân phối

- Mục tiêu: cho phép quan sát các thuộc tính dữ liệu phân phối ra sao
- Các loại biểu đồ thường dùng:
  - Histogram
  - Density Curve with Histogram
  - Density plot
  - Box plot
  - Strip plot
  - Violin Plot
  - Population Pyramid

So sánh

- Mục tiêu: biểu diễn xu hướng giữa các biến trong tập dữ liệu hoặc các nhóm dữ liệu khác nhau của cùng một biến
- Các loại biểu đồ hay sử dụng:
  - Bubble chart
  - Bullet chart
  - Pie chart
  - Net pie chart
  - Donut chart
  - TreeMap
  - Diverging bar
  - Choropleth map
  - Bubble map

2. Bar/Column Chart

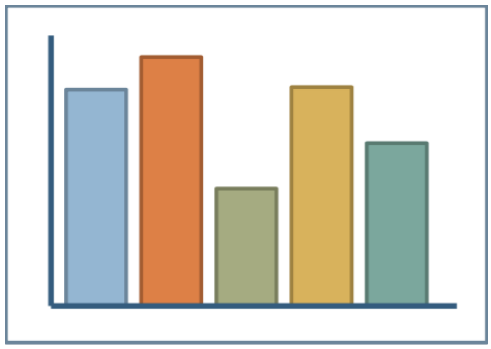
- 4 types of bar charts:
  - Horizontal bar chart
  - Vertical bar chart
  - Group bar chart
  - Stacked bar chart
- Loại biểu đồ này được sử dụng khi chúng ta muốn theo dõi sự phát triển của một hoặc hai biến theo thời gian
- Một trục hiển thị các danh mục cụ thể đang được so sánh (biến độc lập)
- Trục còn lại đại diện cho một giá trị đo được (biến phụ thuộc)

Vertical Bar Chart (Column Chart)

- Phân biệt với Histogram:
  - không hiển thị sự phát triển liên tục trong một khoảng thời gian
  - Dữ liệu rời rạc
  - dữ liệu được phân loại và được sử dụng để trả lời câu hỏi về kích thước trong mỗi danh mục
- Được sử dụng để so sánh một số mục trong một phạm vi giá trị cụ thể
- Lý tưởng để so sánh một danh mục dữ liệu giữa các mục con riêng lẻ

Vertical Bar Chart (Column Chart)

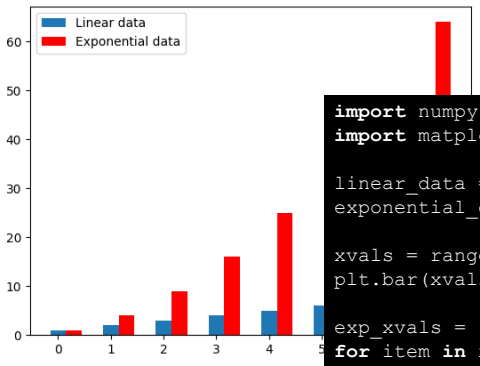
Quantitative
Dependent
variable



Benefits from both position (top of bar) and length (size of bar)

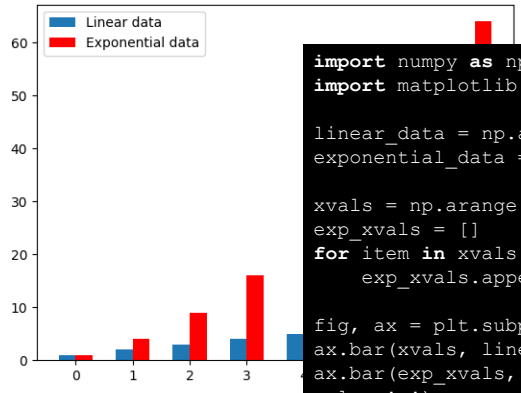
Discrete/Nominal
Independent variable

Vertical Bar Chart (Column Chart)




```
import numpy as np
import matplotlib.pyplot as plt

linear_data = np.array([1, 2, 3, 4, 5, 6, 7, 8])
exponential_data = linear_data ** 2

xvals = range(len(linear_data))
plt.bar(xvals, linear_data, width=0.3)

exp_xvals = []
for item in xvals:
    exp_xvals.append(item+0.3)
plt.bar(exp_xvals, exponential_data, width=0.3,
color='r')

plt.legend(['Linear data', 'Exponential data'])
plt.show()
```

Vertical Bar Chart (Column Chart)




```
import numpy as np
import matplotlib.pyplot as plt

linear_data = np.array([1, 2, 3, 4, 5, 6, 7, 8])
exponential_data = linear_data ** 2

xvals = np.arange(len(linear_data))
exp_xvals = []
for item in xvals:
    exp_xvals.append(item+0.3)

fig, ax = plt.subplots()
ax.bar(xvals, linear_data, width=0.3)
ax.bar(exp_xvals, exponential_data, width=0.3,
color='r')
ax.legend(['Linear data', 'Exponential data'])
ax.set_xticks(xvals + 0.3 / 2)
ax.set_xticklabels(xvals)
plt.show()
```

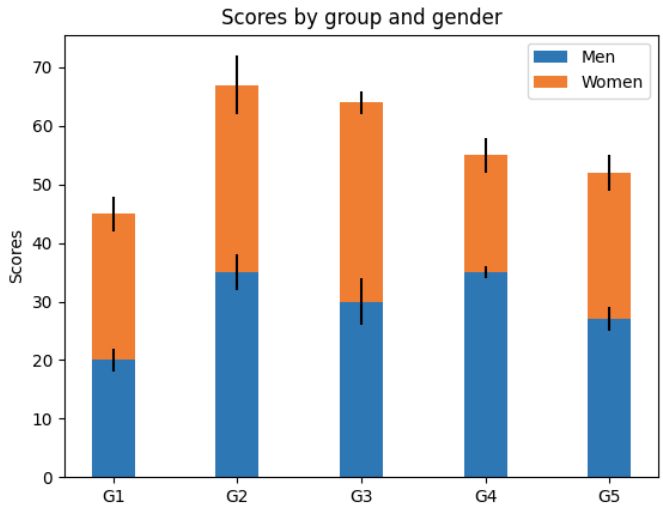

Horizontal Bar Chart

- Biểu diễn dữ liệu theo chiều ngang
- Các danh mục dữ liệu được hiển thị trên trục y
- Các giá trị dữ liệu được hiển thị trên trục x
- Độ dài của mỗi thanh bằng giá trị tương ứng với loại dữ liệu
- Tất cả các thanh đi ngang từ trái sang phải
- Sử dụng hàm barh()

Stacked Bar Chart

- Biểu đồ biểu diễn các thanh xếp chồng lên nhau ứng với các thuộc tính khác nhau
- Được sử dụng để hiển thị một danh mục rộng hơn được chia thành các danh mục nhỏ hơn
- Cho phép quan sát được mối quan hệ của từng phần trên tổng số
- Đặt từng giá trị cho phân đoạn sau giá trị trước đó Tổng giá trị của biểu đồ thanh là tất cả các giá trị phân đoạn được cộng lại với nhau
- Lý tưởng để so sánh tổng số trên mỗi nhóm / thanh được phân đoạn

Stacked Bar Chart



Stacked Bar Chart

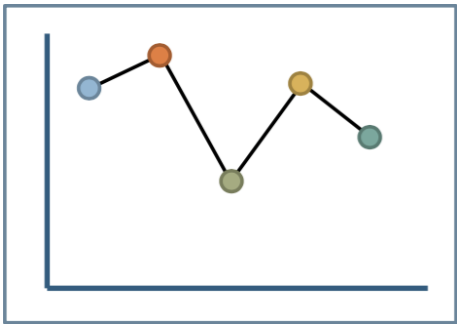
```
import matplotlib.pyplot as plt


labels = ['G1', 'G2', 'G3', 'G4', 'G5']
men_means = [20, 35, 30, 35, 27]
women_means = [25, 32, 34, 20, 25]
men_std = [2, 3, 4, 1, 2]
women_std = [3, 5, 2, 3, 3]
width = 0.35       # the width of the bars: can also be len(x) sequence

fig, ax = plt.subplots()

ax.bar(labels, men_means, width, yerr=men_std, label='Men')
ax.bar(labels, women_means, width, yerr=women_std, bottom=men_means,
       label='Women')

ax.set_ylabel('Scores')
ax.set_title('Scores by group and gender')
ax.legend()

plt.show()
```

3. Line Chart

- Biểu đồ đường được sử dụng để hiển thị các giá trị định lượng trong một khoảng thời gian hoặc khoảng thời gian liên tục
- Được vẽ bằng cách vẽ các điểm dữ liệu đầu tiên trên lưới tọa độ Cartesian và sau đó kết nối chúng
- Trục Y có giá trị định lượng
- Trục X là một thang thời gian hoặc một chuỗi các khoảng thời gian
- Sử dụng cho dữ liệu liên tục
- Được sử dụng thường xuyên nhất để hiển thị xu hướng và phân tích dữ liệu đã thay đổi như thế nào theo thời gian

Line charts

Quantitative continuous dependent variable



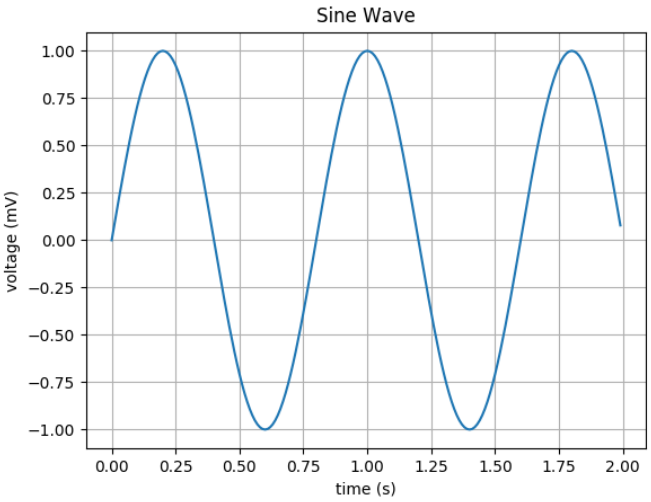
Benefits from position but not length

Quantitative continuous independent variable

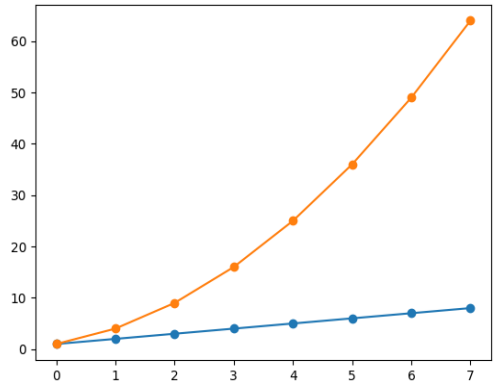
Line chart (pylab vs pyplot)

```
from pylab import *
t = arange(0.0, 2.0, 0.01)
s = sin(2.5*pi*t)
plot(t,s)

xlabel('time (s)')
ylabel('voltage (mV)')
title('Sine Wave')
grid(True)
show()
```

```
import numpy as np
import matplotlib.pyplot as plt
t = np.arange(0.0, 2.0, 0.01)
s = np.sin(2.5*np.pi*t)
plt.plot(t,s)

plt.xlabel('time (s)')
plt.ylabel('voltage (mV)')
plt.title('Sine Wave')
plt.grid(True)
plt.show()
```

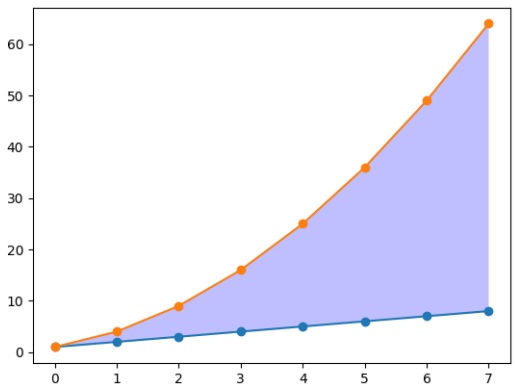


Line chart (cont.)




```
import numpy as np
import matplotlib.pyplot as plt
linear_data =
np.array([1,2,3,4,5,6,7,8])
exponential_data =
linear_data**2
plt.plot(linear_data, '-o',
exponential_data, '-o')
plt.show()
```

Line chart (cont.)

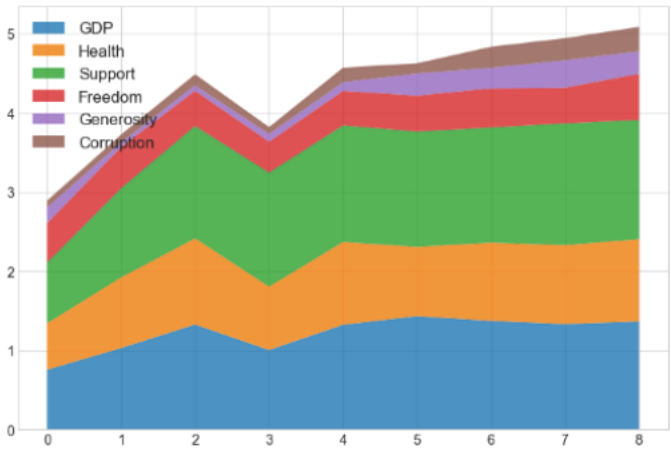



```
import numpy as np
import matplotlib.pyplot as plt
linear_data =
np.array([1,2,3,4,5,6,7,8])
exponential_data = linear_data**2
plt.plot(linear_data, '-o',
exponential_data, '-o')
plt.gca().fill_between(range(len(linear_
data)),
                       linear_data,
exponential_data,
                       facecolor='blue',
                       alpha=0.25)
plt.show()
```

Area Chart

- Được xây dựng dựa trên biểu đồ đường
- Khu vực giữa trục x và đường thẳng được tô màu hoặc tô bóng
- Lý tưởng để minh họa rõ ràng mức độ thay đổi giữa hai hoặc nhiều điểm dữ liệu
- Sử dụng hàm stackplot ()
- Hoặc chỉ cần tô màu vào khu vực giữa hai dòng

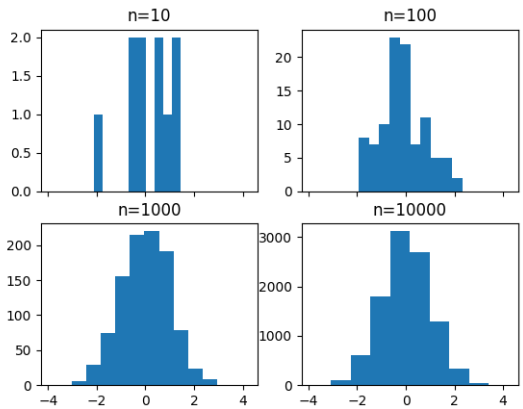
Area Chart



4. Histogram

- Biểu đồ là một đại diện chính xác của sự phân phối dữ liệu số
- Ước tính phân phối xác suất của một biến liên tục
- Để tạo biểu đồ, hãy làm theo các bước sau
  - Xác định gía trị của bins: toàn bộ dải giá trị của dữ liệu được nhóm vào từng khoảng, mỗi khoảng là 1 bin
  - Đếm xem có bao nhiêu gía trị rơi vào 1 khoảng

Histogram example



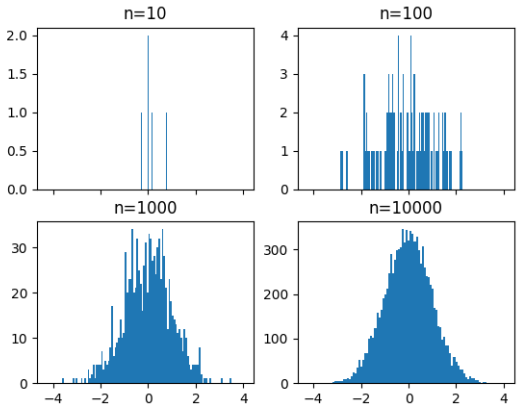
Histogram example

```
import numpy as np
import matplotlib.pyplot as plt

fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2,2,
        sharex=True)
axs = [ax1, ax2, ax3, ax4]

for n in range(0, len(axs)):
    sample_size = 10**(n+1)
    sample = np.random.normal(loc=0.0, scale=1.0,
                size=sample_size)
    axs[n].hist(sample)
    axs[n].set_title('n={}'.format(sample_size))

plt.show()
```

Histogram example

```
import numpy as np
import matplotlib.pyplot as plt

fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2,2,
        sharex=True)
axs = [ax1, ax2, ax3, ax4]

for n in range(0, len(axs)):
    sample_size = 10**(n+1)
    sample = np.random.normal(loc=0.0, scale=1.0,
                size=sample_size)
    axs[n].hist(sample, bins=100)
    axs[n].set_title('n={}'.format(sample_size))

plt.show()
```

Histogram example



5. Scatter plot

- Một loại biểu đồ thường được sử dụng trong thống kê và khoa học dữ liệu.
  - Nó bao gồm nhiều điểm dữ liệu được vẽ trên hai trục
- Mỗi biến được mô tả trong một biểu đồ phân tán sẽ có các quan sát khác nhau
- Được sử dụng để xác định mối quan hệ của dữ liệu với từng biến (nghĩa là tương quan, các mẫu xu hướng)
- Trong học máy, biểu đồ Scatter Plot thường được sử dụng trong hồi quy, trong đó x và y là các biến liên tục Cũng được sử dụng trong phân tán phân cụm hoặc phát hiện ngoại lệ

Practice with Pandas and Seaborn to manipulating data

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

iris =
pd.read_csv("../input/Iris.csv")

iris.head()
```

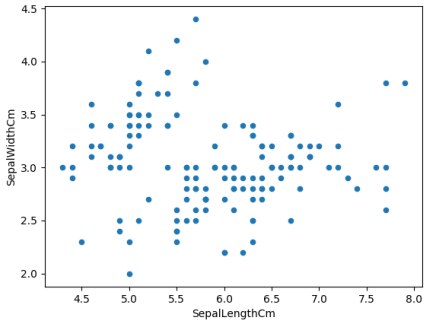
Import the dataset Iris

Practice with Pandas and Seaborn to manipulating data

Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
5	5.0	3.6	1.4	0.2	Iris-setosa
6	5.4	3.9	1.7	0.4	Iris-setosa
7	4.6	3.4	1.4	0.3	Iris-setosa
8	5.0	3.4	1.5	0.2	Iris-setosa
9	4.4	2.9	1.4	0.2	Iris-setosa
10	4.9	3.1	1.5	0.1	Iris-setosa
11	5.4	3.7	1.5	0.2	Iris-setosa
12	4.8	3.4	1.6	0.2	Iris-setosa
13	4.8	3.0	1.4	0.1	Iris-setosa
14	4.3	3.0	1.1	0.1	Iris-setosa
15	5.8	4.0	1.2	0.2	Iris-setosa

Use scatter plot for Iris data

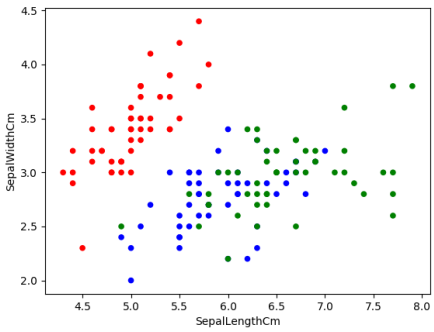
- Plot two variables: SepalLengthCm and SepalWidthCm




```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

iris =
pd.read_csv("../input/Iris.csv")
iris.head()

iris["Species"].value_counts()
iris.plot(kind="scatter",
x="SepalLengthCm",
        y="SepalWidthCm")

plt.show()
```

Use scatter plot for Iris data

- Display color for each kind of Iris

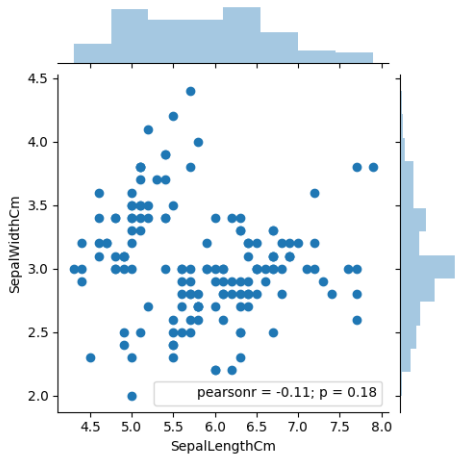



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

iris =
pd.read_csv("../input/Iris.csv")
iris.head()

iris["Species"].value_counts()
col = iris['Species'].map({"Iris-
setosa":'r',     "Iris-
virginica":'g', "Iris-
versicolor":'b'})
iris.plot(kind="scatter",
x="SepalLengthCm",
y="SepalWidthCm", c=col)

plt.show()
```

Marginal Histogram

- Biểu đồ được thêm vào lề của mỗi trục của một biểu đồ phân tán để phân tích sự phân bố của từng số đo
- Đánh giá mối quan hệ giữa hai biến và kiểm tra sự phân bố của chúng

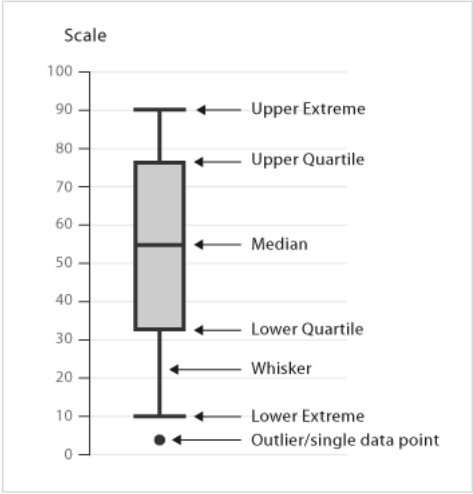
Marginal Histogram




```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

iris =
pd.read_csv("../input/Iris.csv")
iris.head()

iris["Species"].value_counts()
sns.jointplot(x="SepalLengthCm",
y="SepalWidthCm",       data=iris,
size=5)

plt.show()
```

6. Other kinds of chart Box Plot

- Box and Whisker Plot (or Box Plot) là một loại biểu đồ cho phép hiển thị trực quan sự phân bố dữ liệu thông qua các phần tư của chúng



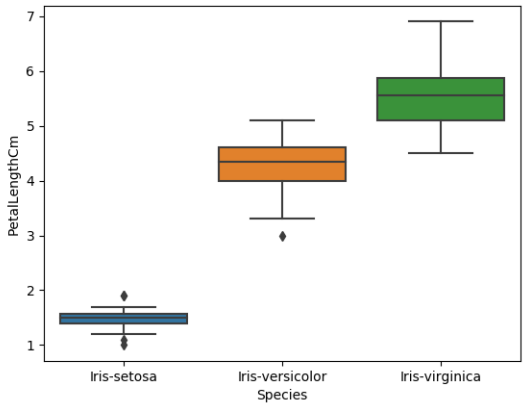
41

Box Plot

- Some observations from viewing Box Plot
- Một số giá trị quan sát được khi xem xét biểu đồ Box Plot của một tập dữ liệu
  - Các giá trị thống kê chính, chẳng hạn như: giá trị trung bình, median, phân vị thứ 25, v.v.
  - Nếu có bất kỳ ngoại lệ nào và giá trị của chúng là gì
  - Dữ liệu có đối xứng không
  - Dữ liệu được nhóm chặt chẽ như thế nào
  - Nếu dữ liệu bị lệch và nếu có thì theo hướng nào

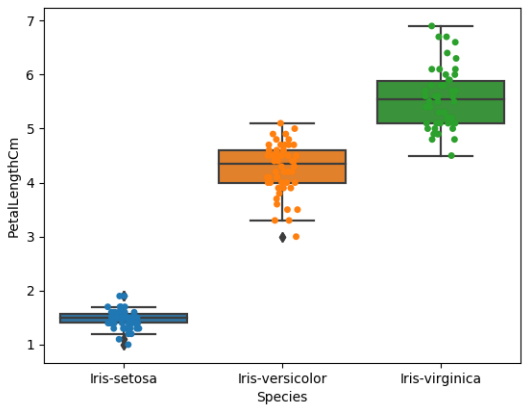
42

Box Plot




```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot
as plt

iris =
pd.read_csv("../input/Ir
is.csv")
iris.head()

sns.boxplot(x="Species",
        y="PetalLengthCm
", data=iris)

plt.show()
```

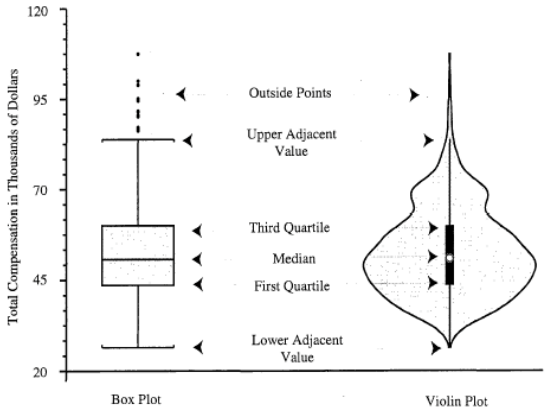

Box Plot




```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as
plt

iris =
pd.read_csv("../input/Iris.
csv")
iris.head()

ax =
sns.boxplot(x="Species",
        y="PetalLengthCm",
        data=iris)
ax =
sns.stripplot(x="Species",
        y="PetalLengthCm",
data=iris,
        jitter=True,
        edgecolor="gray")

plt.show()
```

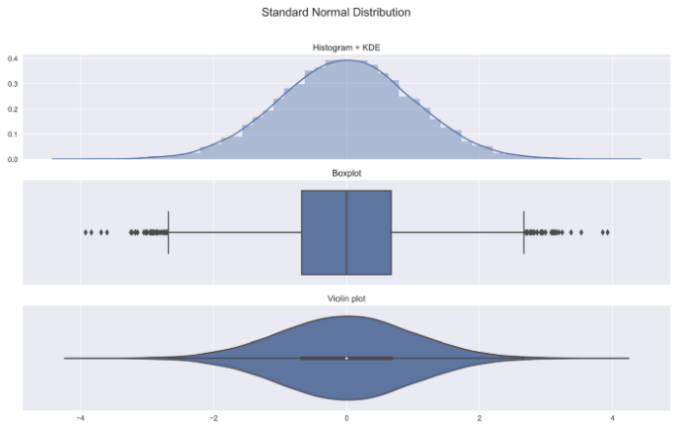
Violin Plot

- Sự kết hợp của box plot và kernel density plot
- Hiển thị thông tin giống hệt như box plot



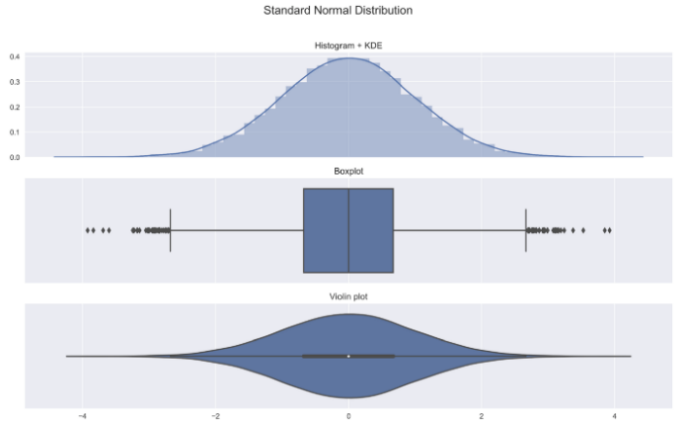
Violin Plot

- Tuy nhiên cho phép thể hiện dạng phân phối của dữ liệu



Violin Plot

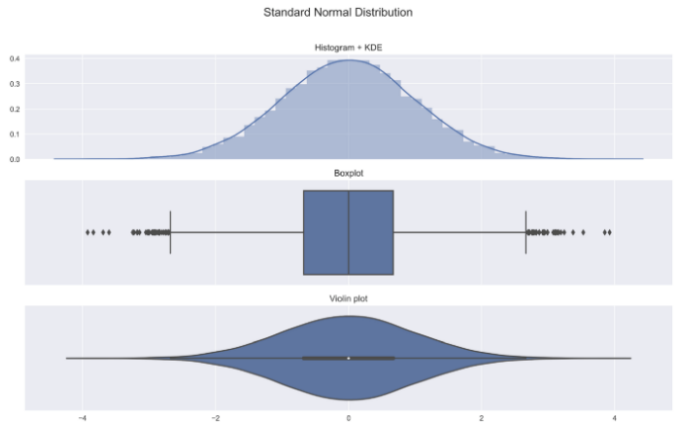
- Histogram cho phép hiển thị sự đối xứng của phân bố



47

Violin Plot

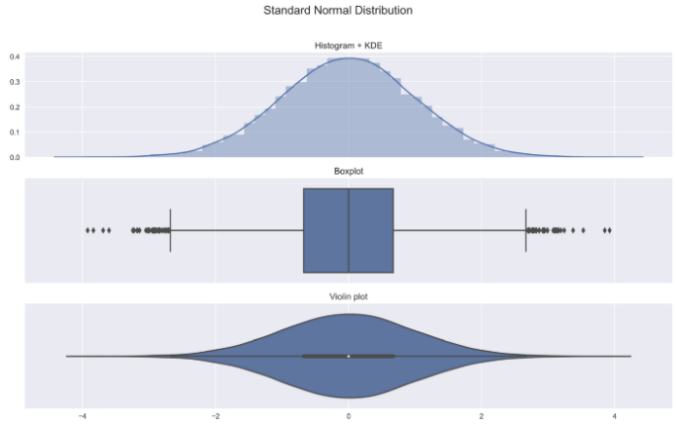
- The kernel density plot used for creating the violin plot is the same as the one added on top of the histogram



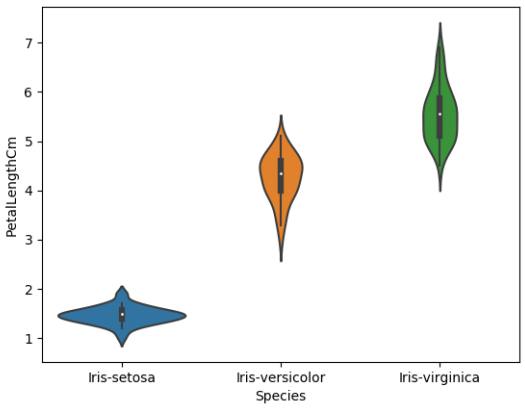
48

Violin Plot

- Các phần rộng hơn của âm mưu vĩ cầm thể hiện xác suất quan sát nhận được một giá trị nhất định cao hơn
- Các phần mỏng hơn tương ứng với một xác suất thấp hơn.



Violin Plot of Iris data




```
import pandas as pd
import seaborn as sns
import
matplotlib.pyplot as
plt

iris =
pd.read_csv("../input/
Iris.csv")
iris.head()

sns.violinplot(x="Spec
ies",
y="PetalLengthCm",
data=iris, size=6)

plt.show()
```

Regression Plot

- Create a regression line between 2 parameters and helps to visualize their linear relationships
- Tạo 1 đường hồi quy giữa hai tham số và giúp biểu diễn mối quan hệ tuyến tính giữa chúng
- Example: Tập dữ liệu về tiền tips chứa các thông tin sau
  - Số lượng người ăn uống tại các nhà hàng và xu hướng trả thêm tiền tips cho người phục vụ
  - Các thuộc tính bao gồm giới tính, độ tuổi, hút thuốc hay không...
- Sử dụng hàm lmplot() của seaborn để tạo regression plot

Regression Plot example

```
# import the library
import seaborn as sns

# load the dataset
dataset = sns.load_dataset('tips')

# the first five entries of the dataset
dataset.head()
```

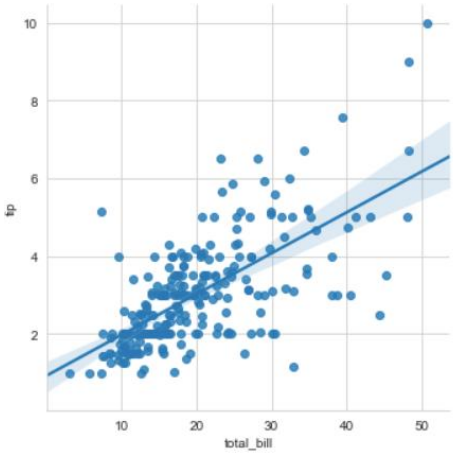
Output

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

Regression Plot Example

```
sns.set_style('whitegrid')
sns.lmplot(x ='total_bill', y ='tip', data = dataset)
```

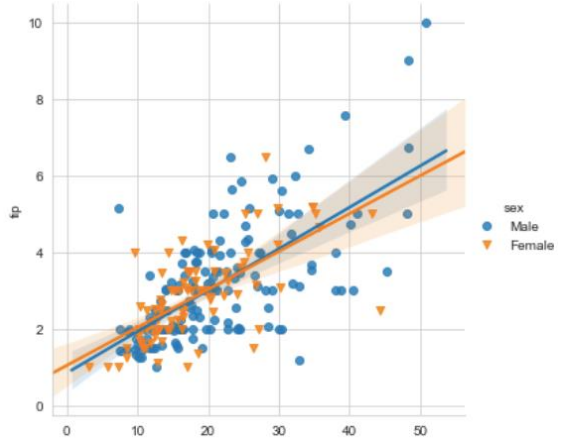
Show the linear relationship betweet the total bill of customers and the tips they gave



Regression Plot Example

```
sns.set_style('whitegrid')
sns.lmplot(x ='total_bill', y ='tip', data = dataset,
           hue ='sex', markers =['o', 'v'])
```

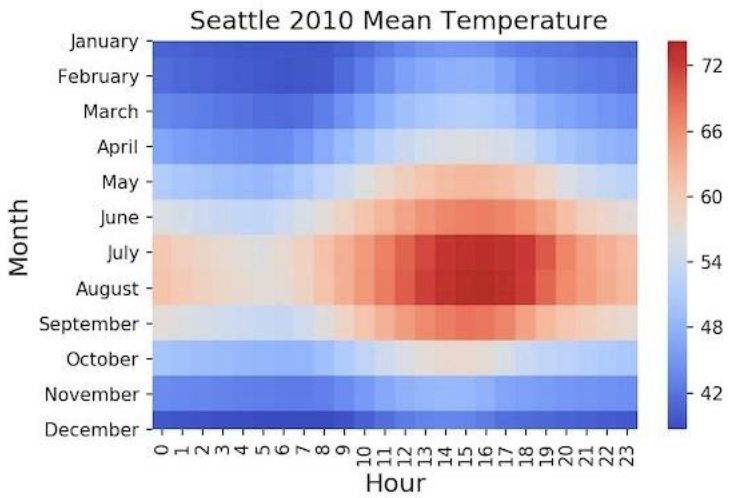
Distinguish two categories by sex



Heatmaps

- Ý tưởng cơ bản: thay thế các con số bằng màu sắc
- Mục tiêu của bản đồ nhiệt là cung cấp một bản tóm tắt thông tin trực quan có màu
- Bản đồ nhiệt rất hữu ích để kiểm tra chéo dữ liệu đa biến, thông qua việc đặt các biến trong các hàng và cột và tô màu các ô trong bảng
- Tất cả các hàng là một danh mục (nhãn hiển thị ở phía bên trái)
- Tất cả các cột là một danh mục khác (nhãn hiển thị ở dưới cùng)
- Dữ liệu trong một ô thể hiện mối quan hệ giữa hai biến trong hàng và cột kết nối

Heatmap Example



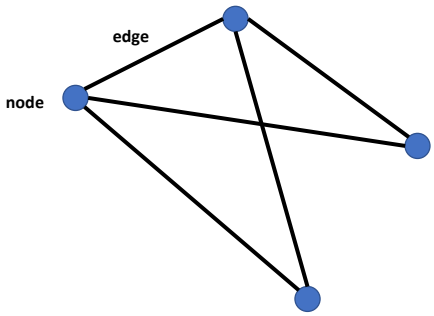
Heatmap with seaborn

```
>>> flights = sns.load_dataset("flights")
>>> flights = flights.pivot("month", "year", "passengers")
>>> ax = sns.heatmap(flights)
```

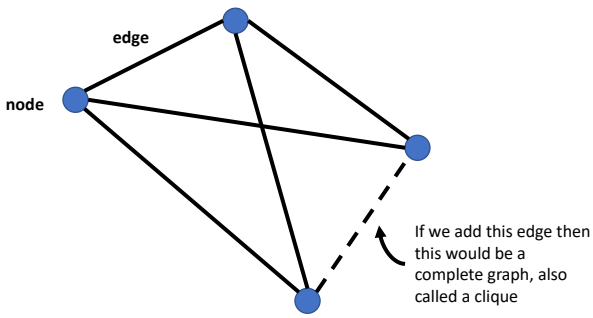
Heatmap with seaborn

```
>>> ax = sns.heatmap(flights, annot=True, fmt="d")
```

Graphs

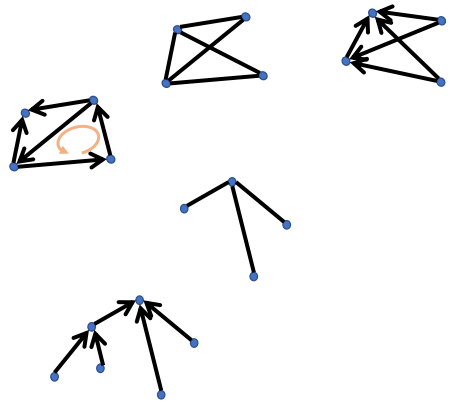


Graphs



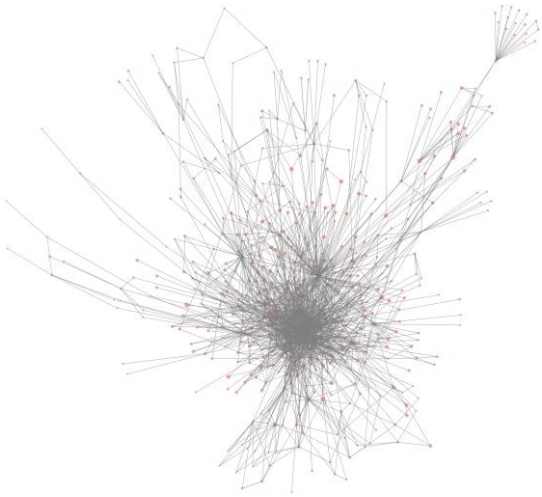
Directed Graphs and Hierarchies

- Có hướng vs Không hướng
- Tuần hoàn hoặc không
- Tree
  - Minimally connected
  - N nodes, n-1 edges
  - Single parent node can have multiple child nodes
- Hierarchy
  - Acyclic directed graph
  - Having a root node



Node Degree

- Degree of a node = number of edges
- Directed graph nodes have an in-degree and an out-degree
- Social Networks
  - Many low degree nodes and fewer high degree nodes
  - Also called power-law or scale-free graphs



Graph Visualization

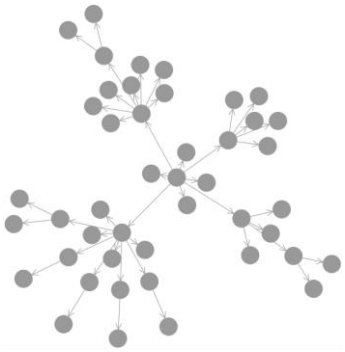
- For visualizing more abstract and non-quantitative data
- For example:
  - The relationship/contacts of individuals in a population (also called network of contacts)
  - The hierarchical structure of classes in a module
- Matplotlib does not support this kind of visualization

Roassal: an agile visualization tool

- Roassal is a DSL, written in Smalltalk and integrated in Pharo/Moose – an open source platform for software and data analysis
- Installing from: http://www.moosetechnology.org

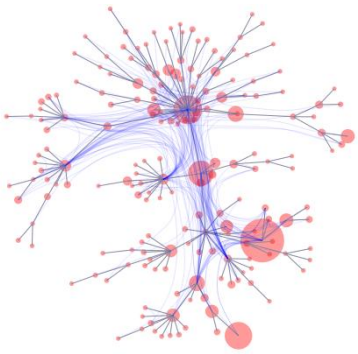
Hierarchy

```
| b |
b := RTMondrian new.
b shape circle size: 30.
b nodes: RTShape withAllSubclasses.
b shape arrowedLine
          withShorterDistanceAttac
hPoint.
b edgesFrom: #superclass.
b layout forceWithCharge: -500.
b build.
^ b view
```



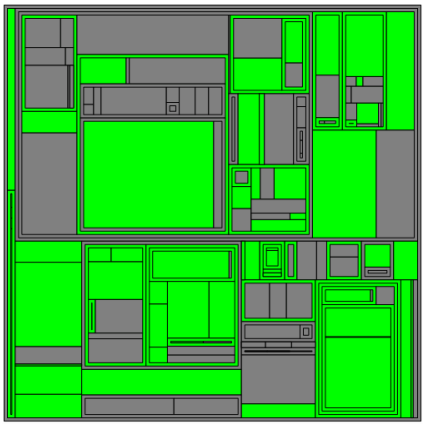
Network structure

```
| b lb |
b := RTMondrian new.
b shape circle color: (Color red alpha: 0.4).
b nodes: Collection withAllSubclasses.
b edges connectFrom: #superclass.
b shape
          bezierLineFollowing: #superclass;
          color: (Color blue alpha: 0.1).
b edges
          notUseInLayout;
          connectToAll: #dependentClasses.
b normalizer normalizeSize: #numberOfMethods min: 5
max: 50.
b layout force.
b build.
lb := RTLegendBuilder new.
lb view: b view.
lb addText: 'Circle = classes, size = number of
methods; gray links = inheritance;'.
lb addText: 'blue links = dependencies; layout =
force based layout on the inheritance links'.
lb build.
^ b view @ RTZoomableView
```

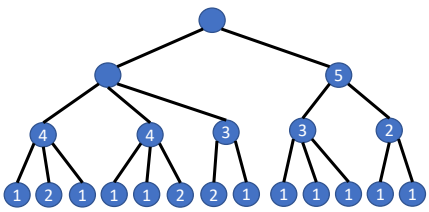


Tree Map

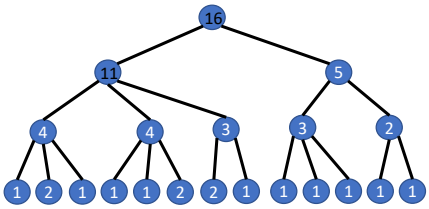
- Maps quantities to area
- Color used to differentiate areas
- Shading delineates hierarchical regions
- When to use?
  - Limited space but large amount of hierarchical data
  - Values can be aggregated in the tree structure
- Advantages
  - Saving space, display a large number of item simultaneously
  - Using color and size of areas to detect special sample data



Tree map layout

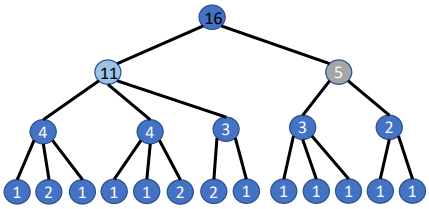


Tree map layout

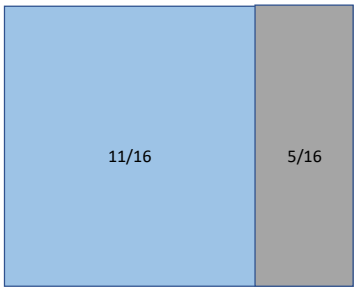


- Set parents node values to sum of child node values from bottom up

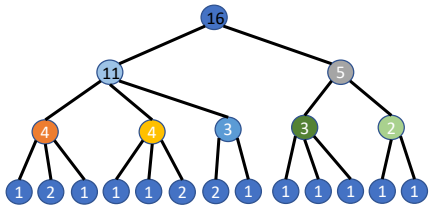
Tree map layout



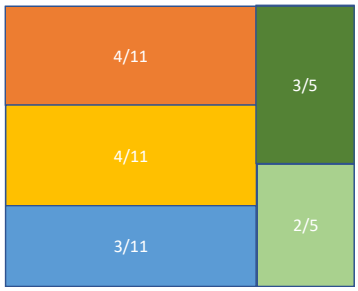
- Set parents node values to sum of child node values from bottom up
- Partition based on current node's value as a portion of parent node's value from top down



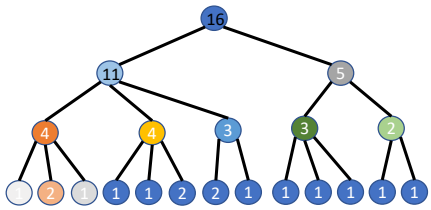
Tree map layout



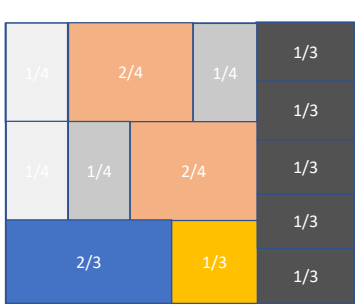
- Set parents node values to sum of child node values from bottom up
- Partition based on current node's value as a portion of parent node's value from top down



Tree map layout

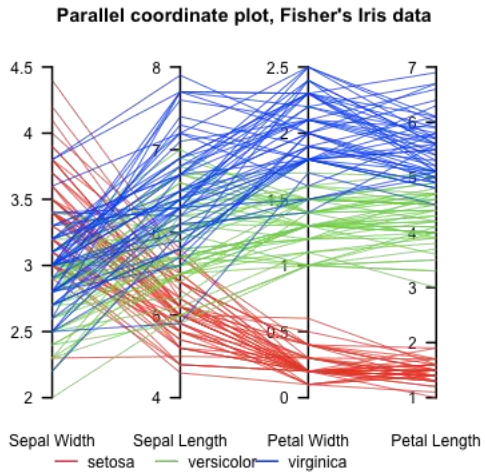


- Set parents node values to sum of child node values from bottom up
- Partition based on current node's value as a portion of parent node's value from top down



7. Trực quan hoá đa chiều

- Đối với dữ liệu nhiều chiều (*n>3* variables): thường dùng **parallel coordinates**
  - Mỗi trục thẳng đứng biểu diễn một biến
  - Mỗi điểm dữ liệu là 1 biểu diễn trong một không gian n chiều
  - **polyline** với **các đỉnh vertices nằm trên các trục song song với nhau**
    - Vị trí của điểm này trên trục i tương ứng với giá trị của thuộc tính i trong tập dữ liệu



Parallel Plot

- Đồ thị Tọa độ Song song cho phép so sánh đặc điểm của một số quan sát riêng lẻ trên một tập hợp các biến số
- Mỗi thanh dọc đại diện cho một biến và thường có thang đo riêng
- Giá trị được vẽ dưới dạng chuỗi các đường nối qua mỗi trục
- Màu có thể được sử dụng để đại diện cho các nhóm cá nhân khác nhau hoặc làm nổi bật một nhóm cụ thể
- Cho phép so sánh các biến thể của trục liền kề
- Thay đổi thứ tự có thể dẫn đến việc phát hiện ra các mẫu mới trong tập dữ liệu

Parallel plot with pandans for Iris data

- Samples are grouped in 3 species
- Setosa seems have smaller petals but its sepal tends to be wider

```
# libraries
import pandas
import matplotlib.pyplot as plt
from pandas.tools.plotting import parallel_coordinates

# Take the iris dataset
import seaborn as sns
data = sns.load_dataset('iris')

# Make the plot
parallel_coordinates(data, 'species', colormap=plt.get_cmap("Set2"))
plt.show()
```

THANK YOU !