

# Python Data Science for Developer



# All About Me Boyd



**Sorratat Sirirattanajakarin,  
Boyd**

Data Scientist @3dsinteractive

“Data is a new UX”

## Education

- Master degree, DPU, Big Data Engineering
- Bachelor of Science, B.Sc. (Hons),  
Agro-industry, Biotechnology, KU



## ประวัติการทำงานภาคเอกชน

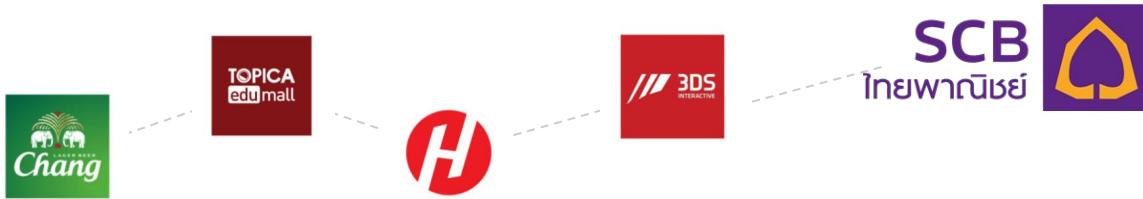
2018 - ปัจจุบันทำงานในตำแหน่ง Data Scientist ในเรื่องของ AI Marketing Automation ณ ที่ บริษัท 3dsinteractive

2017 - ทำงานอยู่ในบริษัท Startup ของต่างประเทศในเครือของ Capstream Ventures Inc ในตำแหน่งของ Digital marketing Executive and Data Scientist (นักการตลาดออนไลน์ นักวิทยาศาสตร์ และวิเคราะห์ด้านข้อมูล)

2016 - Vice Digital Marketing Manager บริษัท Startup จากต่างประเทศ ดูแลเกี่ยวกับการตลาดด้านการศึกษา ผ่าน Social media และ Online Channel

2013 - ประสบการณ์ทำงานในเครือ Thaibeve 3 ปี (หัวหน้าฝ่ายปฏิบัติการการผลิตเบียร์ และให้คำแนะนำฝ่ายการตลาด รวมถึงชุมชน)

2012 - ประสบการณ์ทำงาน Startup กับเครือต่างชาติและช่วยในการวิจัย และออกแบบผลิตภัณฑ์ใหม่ (ผู้ช่วยแผนกต้นแบบออกแบบผลิตภัณฑ์ และฝ่ายขาย)



## ประวัติการทำงานภาคสังคม

- อาจารย์ และวิทยากรด้านการตลาดออนไลน์ ที่บริษัท "siamhr" จัดสอนและอบรมให้กับทางระดับผู้จัดการ ผู้อำนวยการ และผู้ประกอบการ

- อาจารย์ และวิทยากรด้านการตลาดออนไลน์ ให้กับผู้ประกอบการที่ สำนักงานพาณิชย์ จังหวัดปทุมธานี และชุมชน BizClub จังหวัดปทุมธานี

- กรรมการด้าน สารสนเทศและเทคโนโลยี (การตลาดออนไลน์) และให้คำแนะนำเพื่อสร้างความแข็งแกร่งให้กับผู้ประกอบการไทย ที่ "สมาคมนักธุรกิจอาชีวิน"

- สมาชิกกลุ่ม BKK AI Volunteer มอบความรู้ด้าน AI ให้กับผู้ที่เริ่มต้นไปจนถึงขยายความรู้ ด้าน AI ให้กับวงข่าวยิ่งขึ้น

[Bangkok AI]



**FANPAGE - [www.facebook.com/bigdatarpg](https://www.facebook.com/bigdatarpg)**



# Afternoon

---

Day-01



Boyd BigData RPG - Sorratat Sirirattanajakarin

1. Read Data
2. Numpy
3. Pandas
4. Visualization
5. Linear Regression Modeling
6. Project First Day

# Afternoon

---

Day-01



Boyd BigData RPG - Sorratat Sirirattanajakarin

1. **Read Data**
2. Numpy
3. Pandas
4. Visualization
5. Linear Regression Modeling
6. Project First Day

# 2 Main Sources



**Local**



**Server**

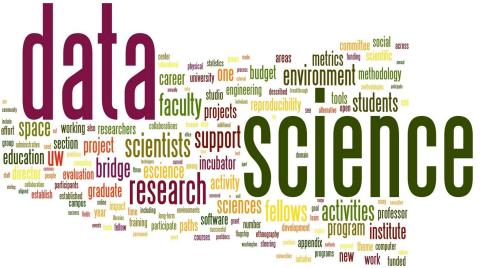


# Read Data From Local



ถึงครึ่งทางกันแล้ว บทนี้เราจะได้เริ่มนำข้อมูลจากภายนอกเข้ามาอ่านใน Python เพื่อทำการประมวลผลต่อ ซึ่งแน่นอน ข้อมูลจากแหล่งต่างๆ ก็จะมีมากมายเหลือเกิน

แต่หลักๆ เลย ก็จะเป็นไฟล์ text, excel, csv นั่นเองครับ



## Plain Text

| PowerBI_Test_Database - Excel |            |        |                |             |               |              |              |                |                             | Mark Klein   |      |
|-------------------------------|------------|--------|----------------|-------------|---------------|--------------|--------------|----------------|-----------------------------|--------------|------|
| File                          | Home       | Insert | Page Layout    | Formulas    | Data          | Review       | View         | ?              | Tell me what you want to do | Share        | Help |
| L14                           | A          | B      | C              | D           | E             | F            | G            | H              | I                           | Annual Yield |      |
| Symbol                        | Stock Name | Shares | Purchase Price | Cost Basis  | Current Price | Market Value | Gain/Loss    | Dividend/share |                             |              |      |
| Apple                         | AAPL       | 100    | \$90.00        | \$9,000.00  | \$144.13      | \$14,413.27  | \$4,469.24   | \$0.28         | 1.58%                       |              |      |
| Microsoft                     | MSFT       | 200    | \$32.00        | \$6,400.00  | \$65.57       | \$13,114.00  | \$6,714.00   | \$1.56         | 2.88%                       |              |      |
| Facebook                      | FB         | 100    | \$100.00       | \$10,000.00 | \$100.00      | \$10,000.00  | \$0.00       | \$0.00         | 0.00%                       |              |      |
| Oracle                        | ORCL       | 250    | \$50.00        | \$12,500.00 | \$44.56       | \$11,381.75  | \$1,392.25   | \$0.44         | 1.44%                       |              |      |
| Hewlett Packard Enterprise    | HPE        | 500    | \$180.00       | \$9,000.00  | \$17.69       | \$8,840.25   | \$824.25     | \$0.26         | 1.47%                       |              |      |
| Alphabet                      | GOOG       | 100    | \$225.00       | \$22,500.00 | \$83.36       | \$8,336.00   | \$20,562.00  | \$0.00         | 0.00%                       |              |      |
| Intel                         | INTC       | 200    | \$220.00       | \$44,000.00 | \$60.67       | \$12,130.00  | \$17,864.00  | \$0.00         | 1.76%                       |              |      |
| Cisco                         | CSCO       | 230    | \$180.00       | \$41,400.00 | \$133.34      | \$30,996.20  | \$10,596.20  | \$0.46         | 2.49%                       |              |      |
| Qualcomm                      | QCOM       | 185    | \$65.00        | \$12,025.00 | \$56.48       | \$10,478.00  | \$0.391,340  | \$0.12         | 3.75%                       |              |      |
| Amazon                        | AMZN       | 50     | \$800.00       | \$40,000.00 | \$897.64      | \$44,882.00  | \$4,882.00   | \$43,984.00    | \$0.00                      | 0.00%        |      |
| Redhat                        | RH         | 100    | \$95.00        | \$9,500.00  | \$86.26       | \$8,626.00   | \$837.94     | \$0.00         | 0.00%                       |              |      |
| Facebook                      | FB         | 1000   | \$17.00        | \$17,000.00 | \$141.94      | \$141,940.00 | \$124,940.00 | \$0.00         | 0.00%                       |              |      |
| Twitter                       | TWTR       | 500    | \$40.00        | \$20,000.00 | \$44.60       | \$22,300.00  | \$2,300.00   | \$0.278,344    | \$0.00                      | 0.00%        |      |

# Excel

## Comma-separated values



# Read Data From Text

สำหรับ path หรือที่อยู่ของ file ที่เราจะทำการเรียกอ่าน ให้ทำการวางแผนในภาพ



python101-01-afternoon



python101-01-01-ReadData.ipynb



input-data



music.txt

**'input-data/music.txt'**

การเขียน path



# Read Data From Text

```
file = open(filename, mode='r')
```

ใส่ path ทำการ open file  
แทนค่าว่า filename

ใส่ mode เพื่อทำการเปิด file  
โดยที่ r คือการอ่าน file

```
text = file.read()
```

ทำการใช้ method .read() เพื่อ  
ทำการอ่าน string ในไฟล์

```
print(text)
```



# Write Data From Text

```
file = open(filename, mode='w')
```

ใส่ path ที่ทำการ save file  
ที่ส่วนของ filename

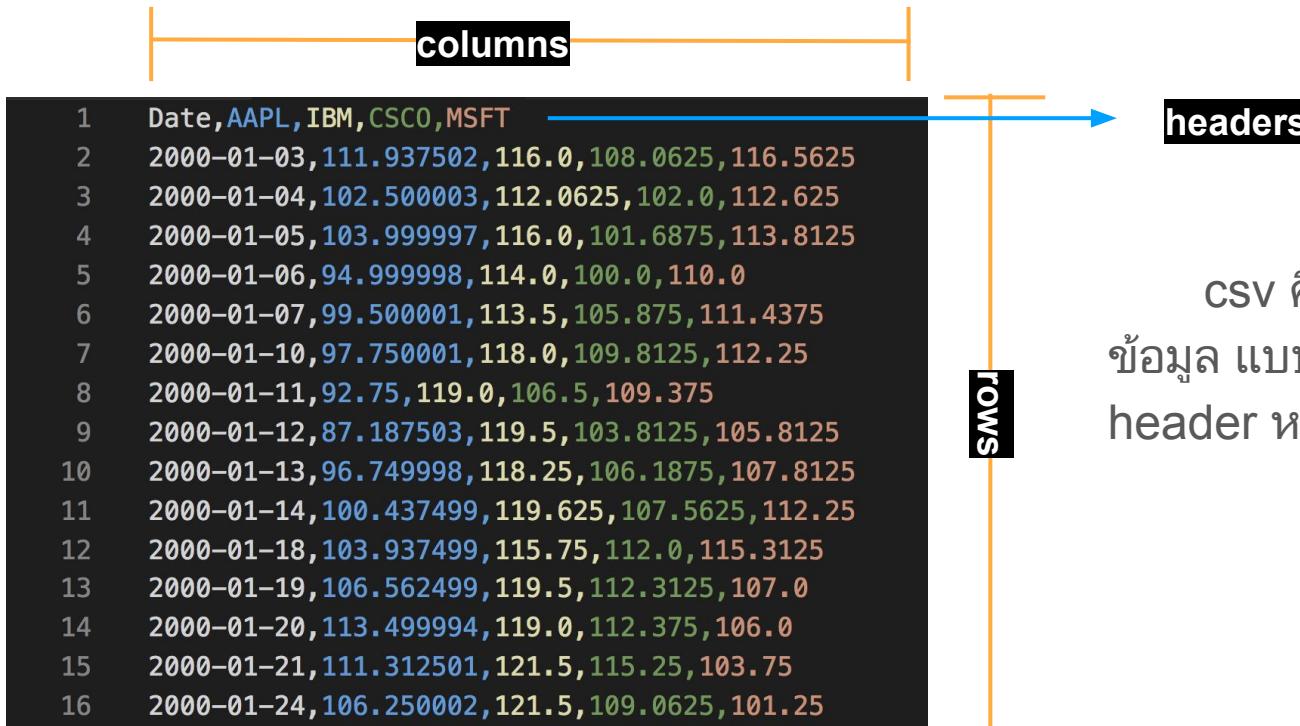
ใส่ mode ที่ทำการเรียกใช้งาน  
w คือการเขียนไฟล์

```
file.write(text)
```

text ทำการใส่ค่า string ที่เรา  
ต้องการทำ การ save



# Understand CSV or Flat File Data

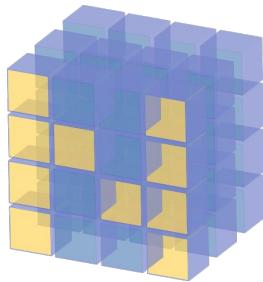


csv คือ flat file ที่มีการเก็บข้อมูลแบบ row, column อาจจะมี header หรือไม่มีก็ได้



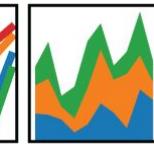
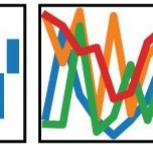
# Useful Library for Data Scientist

วิธีอ่านไฟล์พาก flat file ง่ายที่สุดคือใช้ lib ยอดนิยม 2 lib ได้แก่



NumPy    pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



**numpy**

**pandas**



# How to Install Lib

1

**conda install numpy**

2

**pip install numpy**

Proceed ([y]/n)?

พิมพ์ y และกด enter  
เพื่อทำการติดตั้ง

หากขั้นแรกไม่ได้ให้พิมพ์ขั้น 2  
แทนในการติดตั้ง

1

**conda install pandas**

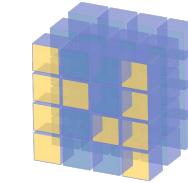
2

**pip install pandas**

Proceed ([y]/n)?

พิมพ์ y และกด enter  
เพื่อทำการติดตั้ง

หากขั้นแรกไม่ได้ให้พิมพ์ขั้น 2  
แทนในการติดตั้ง



**NumPy**

สำหรับการติดตั้ง  
package numpy



**pandas**  
 $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

สำหรับการติดตั้ง  
package pandas



# The Creator of Numpy

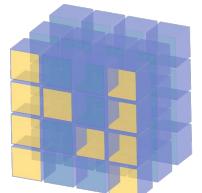
---



**Travis Oliphant**



**ANACONDA**  
Powered by Continuum Analytics®



**NumPy**



# Numpy Reading Flat File Data

เริ่มต้นทำการ import package ที่ต้องใช้ก่อน

```
import numpy as np
```

ทำการเรียกใช้ package  
numpy

ทำการย่อชื่อ package เหลือแค่  
np



# Numpy Reading Flat File Data

```
filename = 'input-data/mnist_kaggle_some_rows.csv'
```

เก็บไว้ในตัวแปรตัวหนึ่ง

ใส่ path ที่อยู่ที่ทำการเก็บไฟล์ไว้

```
data = np.loadtxt(filename, delimiter=',')
```

โดยข้อมูลที่เปิดไปเก็บ  
ในตัวแปรชื่อ data

numpy ในที่นี้ตั้งชื่อ  
เรียกง่ายๆว่า np

ใช้ method ชื่อ  
loadtxt

นำตัวแปรที่เราสร้างไว้  
มาใส่ที่ตรงจุดนี้

ทำการตั้งค่าการแยกค่า  
โดยใช้ , (comma)



# Numpy Skip Row

ทำการเลือก ROW ที่เราไม่ต้องการ ออกไป

```
data2 = np.loadtxt(filename, delimiter=',', skiprows=1)
```



ใช้ส่วนหัวข้าม row ที่  
เราไม่ต้องการนำเข้า  
มาใช้งาน



# Numpy Use specific Column

ทำการเลือก COLUMNS ที่เราต้องการนำมาใช้งาน

```
data3 = np.loadtxt(filename, delimiter=',', skiprows=1,  
                   usecols=[0,2])
```



ใช้สำหรับข้าม เลือก Column  
ที่จะนำมาใช้งาน



# Numpy Customize Data Type

ทำการเปลี่ยนชนิดของค่าที่เรา Import เข้ามาใช้

```
data4 = np.loadtxt(filename, delimiter=',', skiprows=1,  
                   usecols=[0,2], dtype=str)
```



ทำการเปลี่ยนค่าที่ทำการ import เข้า  
มาเป็นค่า string



# Save Flat File via Numpy

```
save_filename = 'output-data/flat-mnist.csv'
```

สร้างตัวแปรเก็บ path สำหรับทำการ save file

```
np.savetxt(save_filename, data3, delimiter=',')
```

method ในการ savefile  
สำหรับ numpy

ใส่ตัวแปรของ path ที่เราจะ  
ทำการ save

ใส่ตัวแปรที่เก็บค่าที่เรา  
ต้องการ save

ทำการใช้ comma ในการ  
แยกค่าตาม column



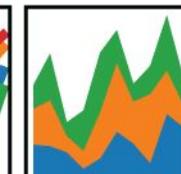
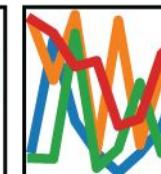
# The Creator of Pandas



**Wes McKinney**



pandas  
 $y_i t = \beta' x_{it} + \mu_i + \epsilon_{it}$



กลุ่มเพื่อน BDFL



# Pandas Reading Flat File Data

เริ่มต้นทำการ import module ที่ต้องใช้ก่อน

```
import pandas as pd
```

ทำการเรียกใช้ module numpy

ทำการย่อชื่อ module เหลือแค่ np



# Pandas Reading Flat File Data

```
filename = 'input-data/titanic_sub.csv'
```

เก็บไว้ในตัวแปรตัวหนึ่ง

ใส่ path ที่อยู่ที่ทำการเก็บไฟล์ไว้

```
df = pd.read_csv(filename)
```

โอนข้อมูลที่เปิดไปเก็บ  
ในตัวแปรชื่อ df หรือย่อ<sup>มาจากการ</sup> dataframe

pandas ในที่นี้ตั้งชื่อ<sup>เรียกง่ายๆว่า</sup> pd

ใช้ method ชื่อ  
read\_csv

นำตัวแปรที่เราสร้างไว้  
มาใส่ที่ตรงจุดนี้



# Pandas Reading Flat File Data

ผลลัพธ์ที่ได้จากการใช้ `read_csv` ค่าที่ได้คือ `data frame`

In [42]: `df.head()`

Out[42]:

|   | PassengerId | Survived | Pclass | Sex    | Age  | SibSp | Parch | Ticket           | Fare    | Cabin | Embarked |
|---|-------------|----------|--------|--------|------|-------|-------|------------------|---------|-------|----------|
| 0 | 1           | 0        | 3      | male   | 22.0 | 1     | 0     | A/5 21171        | 7.2500  | NaN   | S        |
| 1 | 2           | 1        | 1      | female | 38.0 | 1     | 0     | PC 17599         | 71.2833 | C85   | C        |
| 2 | 3           | 1        | 3      | female | 26.0 | 0     | 0     | STON/O2. 3101282 | 7.9250  | NaN   | S        |
| 3 | 4           | 1        | 1      | female | 35.0 | 1     | 0     | 113803           | 53.1000 | C123  | S        |
| 4 | 5           | 0        | 3      | male   | 35.0 | 0     | 0     | 373450           | 8.0500  | NaN   | S        |



# Pandas Dataframe

---

'A matrix has rows and columns.  
A data frame has observation and variables.'



**Hadley Wickham**



# Pandas Skip Row

ทำการเลือก ROW ที่เราไม่ต้องการ ออกไป

```
df_skip = pd.read_csv(filename, skiprows=range(1, 10))
```

ใช้ส่าหรับข้าม row ที่เราไม่ต้องการ  
นำเข้ามาใช้งาน ในที่นี่ทำการข้าม  
row ที่ 1 ถึง row ที่ 5 ออกไป

ใช้ส่าหรับข้าม row ที่เราไม่ต้องการ  
นำเข้ามาใช้งาน ในที่นี่ทำการข้าม  
row ที่ 1 ถึง row ที่ 10 ออกไป

```
df_skip2 = pd.read_csv(filename, skiprows=[1,2,3,4,5])
```



# Pandas Use specific Column

ทำการเลือก COLUMNS ที่เราต้องการนำมาใช้งาน

```
df_col = pd.read_csv(filename,  
usecols=["Survived", "Pclass", "Age", "Ticket"])
```



ใช้สำหรับข้าม เลือก Column  
ที่จะนำมาใช้งาน



# Pandas Customize Data Type

ทำการเปลี่ยนชนิดของค่าที่เรา Import เข้ามาใช้

```
df_col['อายุ'] = df_col['อายุ'].astype(object)
```



ทำการเปลี่ยนค่าที่ทำการ import เข้ามาเป็นค่า object  
ทั้งนี้เพื่อทำการลดเวลาในการประมวลผล และทำให้รูป<sup>แบบของขนาดไฟล์เล็กลงด้วย</sup>



# Save Flat File via Pandas

```
save_filepath = 'output-data/titanic_head.csv'
```

สร้างตัวแปรเก็บ path สำหรับทำการ save file

```
dfHead.to_csv(save_filepath)
```

ตัวแปรที่เก็บค่า  
รูปแบบของ data  
frame

เรียกใช้ method เพื่อทำ  
การ save file ในรูปแบบ  
csv

ใส่ตัวแปรที่เก็บค่า path ที่  
เราต้องการ save



# Understand Excel File

|                                    | 2002      | 2004 |
|------------------------------------|-----------|------|
| War, age-adjusted mortality due to | 2002      |      |
| Afghanistan                        | 36.08399  |      |
| Albania                            | 0.1289084 |      |
| Algeria                            | 18.31412  |      |
| Andorra                            | 0         |      |
| Angola                             | 18.96456  |      |
| Antigua and Barbuda                | 0         |      |
| Argentina                          | 0         |      |
| Armenia                            | 0.1702969 |      |
| Australia                          | 0         |      |
| Austria                            | 0         |      |
| Azerbaijan                         | 0.0819914 |      |
| Bahamas                            | 0         |      |
| Bahrain                            | 0         |      |
| Bangladesh                         | 0.180369  |      |
| Barbados                           | 0         |      |
| Belarus                            | 0         |      |
| Belgium                            | 0         |      |
| Belize                             | 0         |      |
| Benin                              | 0         |      |
| Bhutan                             | 0         |      |
| Bolivia                            | 0         |      |
| Bosnia and Herzegovina             | 1.095922  |      |
| Botswana                           | 0         |      |
| Brazil                             | 0         |      |

sheets

headers

Excel คือ flat file ประเภท  
หนึ่ง มีความคล้ายกับ csv file ที่มี  
การเก็บข้อมูล แบบ row, column  
อาจจะมี header หรือไม่มีก็ได้ แต่  
สำคัญเลย Excel File มีส่วนที่เพิ่ม  
เข้ามาคือ sheets



# Pandas Reading Excel Data

```
filepath_excel = 'input-data/battledeath.xlsx'
```

เก็บไว้ในตัวแปรตัวนึง

ใส่ path ที่อยู่ที่ทำการเก็บไฟล์ไว้

```
df_excel = pd.ExcelFile(filepath_excel)
```

โอนข้อมูลที่เปิดไปเก็บใน  
ตัวแปรชื่อ df\_excel หรือย่อ<sup>มา</sup>จาก dataframe

pandas ในที่นี้ตั้งชื่อ<sup>เรียกง่ายๆว่า</sup> pd

ใช้ method ชื่อ  
excelFile เพื่อทำการ  
ตรวจสอบ Sheets

นำตัวแปรที่เราสร้างไว้  
มาใส่ที่ตรงจุดนี้



# Pandas Reading Excel Data

**df\_excel.sheet\_names**

นำตัวแปรที่เก็บ sheets ของ excel file  
มาทำการเรียกดูว่ามี  
Sheet อะไรบ้าง

ทำการเรียกใช้ method  
sheet\_names เพื่อเรียกดูชื่อของ  
sheets ต่างๆที่มีอยู่ภายในไฟล์ excel



# Pandas Reading Excel Data

```
df_2002 = df_excel.parse('2002')
```

```
df_2004 = df_excel.parse('2004')
```

สร้างตัวแปรมารับค่า data frame ของแต่ละ sheet

ใช้ชื่อ sheet ที่ต้องการเข้าไปทำการวัด data ออกมา

ทำการเรียกใช้ method เพื่อนำค่าจาก sheet ที่เราต้องการโynไปเก็บยังตัวแปรที่เราสร้างขึ้น



# Save Excel via Pandas

```
writer = pd.ExcelWriter('output-data/battledeath-output.xlsx')
```

สร้างตัวแปรเก็บ path สำหรับทำการ save file ไว้ที่ตัวแปร writer

```
df_2002.to_excel(writer,'2002')
```

```
df_2004.to_excel(writer,'2004')
```

↓  
ชื่อตัวแปร data frame

ใช้ method เพื่อทำการ save ใน  
รูปแบบไฟล์ excel อยู่ใน  
ไฟล์เดียวกับตัวแปร writer

ชื่อ sheets ที่จะไปอยู่  
ในไฟล์excel ที่ทำการ save

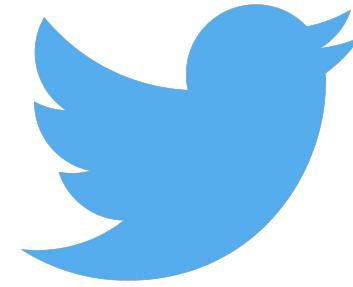
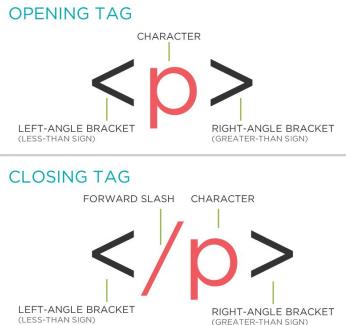
```
writer.save()
```

สร้างตัวแปรเก็บ path สำหรับทำการ save file ไว้ที่ตัวแปร writer



# Read Data From Server

ข้อมูลอีกส่วนที่เรารสามารถเข้าถึงได้โดยใช้ภาษา Python ก็คือจากแหล่ง Server เช่น เข้าไปภาัดจากหน้าเวปไซต์ จากแหล่ง API หรือจาก Streaming แบบ Realtime ซึ่งหาก เรารู้วิธีได้ข้อมูลเหล่านี้มาย่อมเป็นประโยชน์มากในการวิเคราะห์ และทำ Machine Learning



Tag

JSON

Streaming



# Automate file download in Python

ในมุมของนักวิเคราะห์อย่างเรา หากข้อมูล haya นัก ก็โหลดมาเองซะเลย ชึ่งวิธีนี้  
สามารถทำเป็นอัตโนมัติได้อย่างง่ายๆ ด้วยวิธีการที่เรียกว่า Scrapping Website



<https://archive.ics.uci.edu/ml/datasets.html>





# Get Data From URL

```
from urllib.request import urlretrieve
```



Package ที่เราเรียกใช้



Function ที่เราเรียกใช้

urllib คือ package  
request คือ sub-package



# Get Data From URL

---

เข้ามาเวปแล้วก็เลือก ลิงค์ Data Set ที่เราต้องการจะโหลดมาใช้ได้เลย โดยเราทำการ copy link มาใส่ในตัวแปรที่เข้าใจง่ายๆ เช่น url

url =

```
'http://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv'
```



# Get Data From URL

หลังจากนั้นเรียกใช้ฟังก์ชัน urlretrieve โดยจะรับ ส่วน agreement ดังนี้

```
urlretrieve(url, 'path')
```

Link ที่มี dataset



Path ที่เราต้องการ  
save data  
ภายหลังการโหลด





# Result Get Data From URL



Files    Running    Clusters

Select items to perform actions on them.

0    / YoungDSWithPython / python101-01-afternoon / download-data

..

winequality-white.csv



# Get Text From URL

---

**import requests**



**function ที่เราเรียกใช้**

**requests เป็นฟังก์ชัน ที่อยู่ใน package urllib**



# Get Text From URL

---

สร้างตัวแปรชื่อ r เพื่อรับค่าจากฟังก์ชัน  
โดยครั้งต่อไปเรารอเรียกใช้เพียงแค่พิมพ์ว่า r

url ใส่ link ที่เราต้องการดูดข้อมูลมา  
นั่นเอง

**requests.get(url)**



r

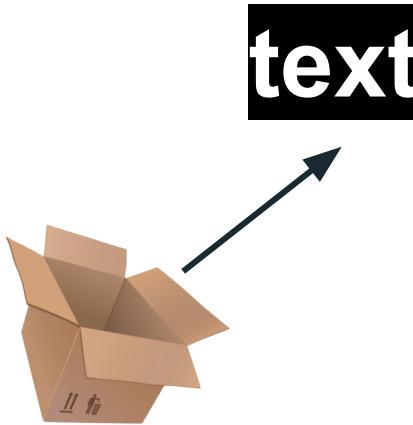


# Get Text From URL

---

ทำการเรียกใช้ method .text เพื่อทำการ get ค่าที่มีแต่ text ภายในกล่อง

เพราะ url ที่เราทำการเก็บค่าในกล่องนั้น มีหลายองค์ประกอบนั่นเอง แต่ในที่นี่เราต้องการส่วนที่เป็นแค่ text



r.text



# Result from Get Text From URL



r.text

```
<!DOCTYPE html><html class="client-nojs" lang="en" dir="ltr"><head><meta charset="UTF-8"/><title>BNK48 – Wikipedia</title><script>document.documentElement.className = document.documentElement.className.replace( /(^|\s)client-nojs(\s|$)/, "$1client-js$2" );</script><script>(window.RLQ>window.RLQ||[]).push(function(){mw.config.set({"wgCanonicalNamespace": "", "wgCanonicalSpecialPageName": false, "wgNamespaceNumber": 0, "wgPageName": "BNK48", "wgTitle": "BNK48", "wgCurRevisionId": 857799706, "wgRevisionId": 857799706, "wgArticleId": 53223027, "wgIsArticle": true, "wgIsRedirect": false, "wgAction": "view", "wgUserName": null, "wgUserGroups": ["*"], "wgCategories": ["CS1 Thai-language sources (th)", "CS1 Japanese-language sources (ja)", "Use dmy dates from February 2017", "Official website different in Wikidata and Wikipedia", "Articles with hCards", "Articles containing potentially dated statements from August 2018", "All articles containing potentially dated statements", "Articles containing Thai-language text", "BNK48", "AKB48 Group", "Japan-Thailand relations", "Musical groups established in 2017", "Musical groups from Bangkok", "Thai girl groups", "Thai pop music groups", "2017 establishments in Thailand"], "wgBreakFrames": false, "wgPageContentLanguage": "en", "wgPageContentModel": "wikitext", "wgSeparatorTransformTable": [ "", "" ], "wgDigitTransformTable": [ "", "" ], "wgDefaultDateFormat": "dmy", "wgMonthNames": [ "", "January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December" ], "wgMonthNamesShort": [ "", "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec" ], "wgRelevantPageName": "BNK48", "wgRelevantArticleId": 53223027, "wgRequestId": "W4yi4gpAADwAAJQW0SkAAABM", "wgCSPNonce": false, "wgIsProbablyEditable": true, "wgRelevantPageIsProbablyEditable": true, "wgRestrictionEdit": [ ], "wgRestrictionMove": [ ], "wgFlaggedRevsParams": { "tags": {} }, "wgStableRevisionId": null, "wgCategoryTreePageCategoryOptions": ":{\\\"mode\\\":0,\\\"hideprefix\\\":20,\\\"showcount\\\":true,\\\"namespaces\\\":false}", "wgWikiEditorEnabledModules": [ ], "wgBetaFeaturesFeatures": [ ], "wgMediaViewerOnClick": true, "wgMediaViewerEnabledByDefault": true, "wgPopupsShouldSendModuleToUser": true, "wgPopupsConflictsWithNavPopupGadget": false, "wgVisualEditor": { "pageLanguageCode": "en", "pageVariantFallbacks": "en", "usePageImages": true, "usePageDescriptions": true, "wgTrue": true, "wgMFEnableFontChanger": true, "wgMFDisplayWikibaseDescriptions": { "search": true, "tagline": false, "wgRelatedArticles": null, "wgRelatedArticlesUseCirrusOnlyUseCirrusSearch": false, "wgULSCurrentAutonym": "English", "wgNoticeProjectOkiesToDelete": [], "wgCentralNoticeCategoriesUsingLegacy": [ "Fundraising", "fundraising", "Q23592958" ], "wgScoreNoteLanguages": { "arabic": "\u0627\u062f\u0628\u062a\u0628\u0627\u062f", "catalan": "catal\u00e0", "english": "english", "espanol": "espa\u00f1ol", "italiano": "italiano", "nederlands": "Nederlands", "n\u00f3gu\u00e9s": "n\u00f3gu\u00e9s", "suomi": "suomi", "svenska": "svenska", "vlaams": "West-Vlaams" }, "wgScoreDelete": true, "wgCentralAuthMobileDomain": false, "wgCodeMirrorEnabled": true, "wgVisualEditorUnsupportedEditParams": [ "undo", "undoafter", "veswitched" ], "wgEditSubmit.loader.state": { "ext.gadget.charinsert-styles": "ready", "ext.globalCssJs.user.styles": "ready", "site.styles": "ready", "noscript": "ready", "user.styles": "ready" } } }
```



# Make text more cleaner !!



Mock Turtle singing BEAUTIFUL SOUP as illustrated by John Tenniel (1866)

เป็นชื่อบทเพลงในเรื่อง Alice In Wonderland บทที่ 5 โดยทำการดัดแปลงเนื้อจากเพลง Star of The Evening ของ James M. Sayle

## BeautifulSoup





# Start Open BeautifulSoup

---

## From bs4 Import BeautifulSoup



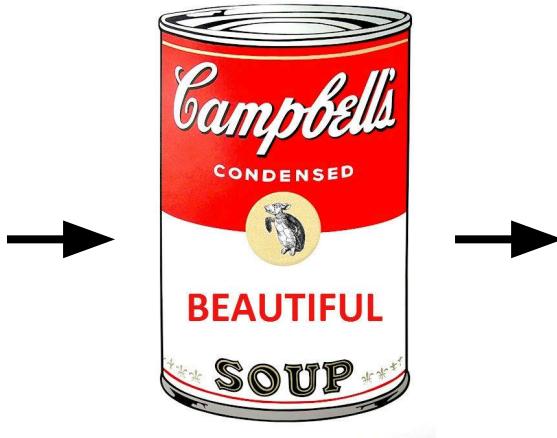
function ที่เราเรียกใช้

BeautifulSoup เป็นพังก์ชัน ที่อยู่ใน package bs4

# Start Open BeautifulSoup



ทำการโyn Text ที่ปุ่งเหยิงเข้าไปในฟังกชัน BeautifulSoup ลด



# Text form

# BeautifulSoup

# HTML form

Boyd BigData RPG - Sorratat Sirirattanajakarin



# Is it Enough Scrapping !??

```
<!DOCTYPE html>
<html class="client-nojs" dir="ltr" lang="en">
<head>
<meta charset="utf-8"/>
<title>BNK48 - Wikipedia</title>
<script>document.documentElement.className = document.documentElement.className.replace( /(^\s|$)/, "$1client-js$2" );</script>
<script>(window.RLQ=window.RLQ||[]).push(function(){mw.config.set({"wgCanonicalNamespace":"",
"wgCanonicalSpecialPageName":false,"wgNamespaceNumber":0,"wgPageName":"BNK48","wgTitle":"BNK48","wgCurRevisionId":857799706,"wgRevisionId":857799706,"wgArticleId":53223027,"wgIsArticle":true,"wgIsRedirect":false,"wgAction":"view","wgUserName":null,"wgUserGroups":["*"],"wgCategories":["CS1 Thai-language sources (th)","CS1 Japanese-language sources (ja)","Use dmy dates from February 2017","Official website different in Wikidata and Wikipedia","Articles with hCards","Articles containing potentially dated statements from August 2018","All articles containing potentially dated statements","Articles containing Thai-language text","BNK48","AKB48 Group","Japan-Thailand relations","Musical groups established in 2017","Musical groups from Bangkok","Thai girl groups","Thai pop music groups","2017 establishments in Thailand"],"wgBreakFrames":false,"wgPageContentLanguage":"en",
"wgPageContentModel":"wikitext","wgSeparatorTransformTable":[["",""], "wgDigitTransformTable":[["",""]], "wgDefaultDateFormat":"dmy","wgMonthNames":["","","January","February","March","April","May","June","July","August","September","October","November","December"], "wgMonthNamesShort":["","Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"], "wgRelevantPageName":"BNK48","wgRelevantArticleId":53223027,"wgRequestId":"W4yi4gpAAJQW0SkAAABM","wgCSPNonce":false,"wgIsProbablyEditable":true,"wgRelevantPageIsProbablyEditable":true,"wgRestrictionEdit":[], "wgRestrictionMove":[]}, {"tags":{}}, "wgStableRevisionId":null, "wgCategoryTreePageCategoryOptions":{},"wgHidePrefix":20, "wgShowCount":true, "wgNamespaces":false}, "wgWikiEditorEnabledModules":[], "wgFeaturesFeatures":[], "wgMediaViewerOnClick":true, "wgMediaViewerEnabledByDefault":true, "wgShouldSendModuleToUser":true, "wgPopupsConflictsWithNavPopupGadget":false, "wgVisualEditor":{},"wgLanguageCode": "en", "wgPageLanguageDir": "ltr", "wgPageVariantFallbacks": "en", "usePageImages": true, "wgDescriptions": true}, "wgMFExpandAllSectionsUserOption": true, "wgMFEnableFontChanger": true, "wgDisplayWikibaseDescriptions": {"search": true, "nearby": true, "watchlist": true, "tagline": false}, "wgRelatedArticles": null, "wgRelatedArticlesUseCirrusSearch": true, "wgRelatedArticlesOnlyUseCirrusSearch": true, "wgULSCurrentAutonym": "English", "wgNoticeProject": "wikipedia", "wgCentralNoticeCookiesToDelete": "all", "wgCentralNoticeCategoriesUsingLegacy": ["Fundraising", "fundraising"], "wgWikibaseItemId": "Q23592958", "wgCoreNoteLanguages": {"arabic": "العربية", "catalan": "català", "deutsch": "Deutsch", "english": "English", "espanol": "español", "italiano": "italiano", "nederlands": "Nederlands", "norsk": "norsk", "portuguese": "guês", "suomi": "suomi", "svenska": "svenska", "vlaams": "West-Vlaams"}, "wgScoreDefaultNoteLanguage": "English", "wgCentralAuthMobileDomain": false, "wgCodeMirrorEnabled": true}
```

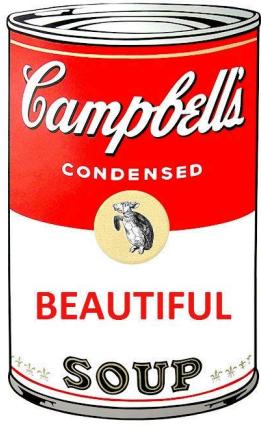


# Make script more prettier !!



ทำง่ายๆ เพียง เรียกใช้ method ของทาง BeautifulSoup

A black arrow pointing to the right, indicating the direction of the next section.



→

# Text form

# prettify()

# HTML form

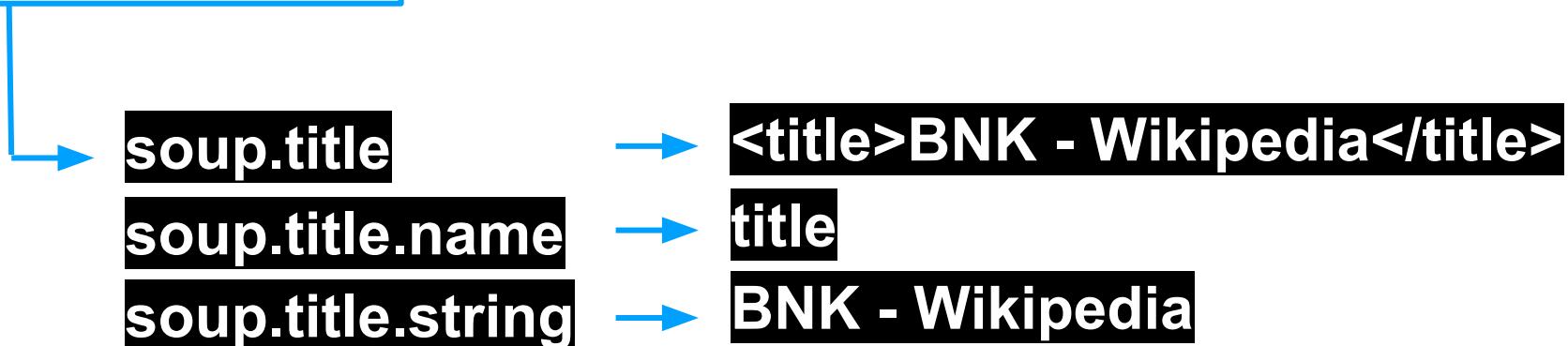
# At least it Improve !!





# Now, Let's get the Data You Want !!

```
<!DOCTYPE html>
<html class="client-nojs" dir="ltr" lang="en">
<head>
<meta charset="utf-8"/>
<title>
  BNK48 – Wikipedia
</title>
```





# Now, Let's get the Data You Want !!

```
<a class="mw-jump-link" href="#mw-head">  
Jump to navigation  
</a>  
<a class="mw-jump-link" href="#p-search">  
Jump to search  
</a>
```

กลุ่ม tag <a></a>

ด้านในมี href อยู่ใช่ไหม  
เอี่ย !?? เอาล่ะมาลอง get ค่า  
เหล่านี้กันดีกว่า !!

```
for link in soup.find_all('a'):  
    print(link.get('href'))
```

#mw-head  
#p-search

Code ที่เขียน

ค่าที่ได้



# Get JSON From API



JSON

```
{ Object Starts
  "Title": "The Cuckoo's Calling",
  "Author": "Robert Galbraith",
  "Genre": "classic crime novel",
  "Detail": { Object Starts
    "Publisher": "Little Brown" Value string
    "Publication_Year": 2013, Value number
    "ISBN-13": 9781408704004,
    "Language": "English",
    "Pages": 494
  } Object ends
  "Price": [ Array starts
    {
      Object Starts
      "type": "Hardcover",
      "price": 16.65,
    } Object ends
    {
      Object Starts
      "type": "Kindle Edition",
      "price": 7.03,
    } Object ends
  ] Array ends
} Object ends
```

JSON



# Get JSON From API

## OMDb API

The Open Movie Database

The OMDb API is a RESTful web service to obtain movie information, all content and images on the site are contributed and maintained by our users.

If you find this service useful, please consider making a [one-time donation](#) or [become a patron](#).



Poster API

The Poster API is only available to patrons.

Currently over 280,000 posters, updated daily with resolutions up to 2000x3000.

<http://www.omdbapi.com/>



# Method to Get JSON From API

1 import requests

2 url = 'http://www.omdbapi.com/?apikey=ff21610b&t=social+network'

3 r = requests.get(url)

4 json\_data = r.json()

5 for k in json\_data.keys():  
 print(k + ': ', json\_data[k])

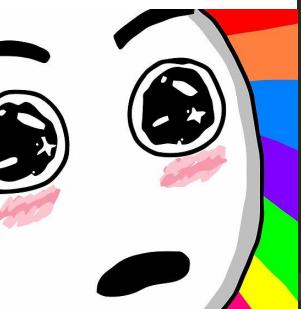
API ที่ใช้เข้าถึงข้อมูล  
ได้จากการสมัคร

ชื่อหนังที่เรา  
ต้องการข้อมูล



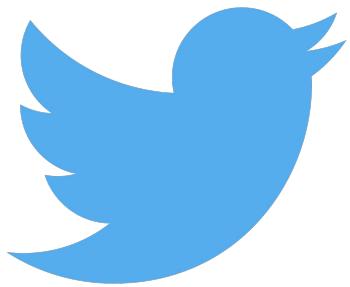
# Result After Using Method

Language: English, French  
Awards: Won 3 Oscars. Another 165 wins & 168 nominations.  
Genre: Biography, Drama  
Director: David Fincher  
imdbRating: 7.7  
Response: True  
Rated: PG-13  
Released: 01 Oct 2010  
imdbVotes: 546,709  
imdbID: tt1285016  
Title: The Social Network  
Country: USA  
Plot: Harvard student Mark Zuckerberg creates the social networking site that would become known as Facebook, but is later sued by two brothers who claimed he stole their idea, and the co-founder who was later squeezed out of the business.  
DVD: 11 Jan 2011  
Ratings: [{"Value": "7.7/10", "Source": "Internet Movie Database"}, {"Value": "95%", "Source": "Rotten Tomatoes"}, {"Value": "95/100", "Source": "Metacritic"}]  
Writer: Aaron Sorkin (screenplay), Ben Mezrich (book)  
Production: Columbia Pictures  
BoxOffice: £96,400,000  
Type: movie  
Runtime: 120 min  
Metascore: 95  
Poster: [https://m.media-amazon.com/images/M/MV5BMTM2ODk0NDAwMF5BMl5BanBnXkFtZTcwNTM1MDc2Mw@@.\\_V1\\_SX300.jpg](https://m.media-amazon.com/images/M/MV5BMTM2ODk0NDAwMF5BMl5BanBnXkFtZTcwNTM1MDc2Mw@@._V1_SX300.jpg)  
Website: <http://www.thesocialnetwork-movie.com/>  
Actors: Jesse Eisenberg, Rooney Mara, Bryan Barter, Dustin Fitzsimons  
Year: 2010





# Get Data From Twitter



ไปสมัครกันก่อนได้เลย

เวปสมัคร สมาชิก

**<https://twitter.com>**

เวปลงทะเบียนเพื่อรับ Access Token

**Twitter**

**<https://apps.twitter.com>**



# Install Package

---

**conda install tweepy**

1

**pip install tweepy**

2



หากขั้นแรกไม่ได้ให้พิมพ์ขั้น 2  
แทนในการติดตั้ง



# How to Access Data From Twitter

Application Management

redthegxtest

Details    Settings    Keys and Access Tokens    Permissions

## Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key) [REDACTED]

Consumer Secret (API Secret) [REDACTED]

Access Level    Read and write ([modify app permissions](#))

Owner [REDACTED]

Owner ID [REDACTED]

สิ่งที่ใช้มี 4 ตัวด้วยกันในการ  
เข้าถึงข้อมูล twitter

1. access\_token
2. access\_token\_secret
3. consumer\_key
4. consumer\_secret



# Method to Get Tweet Streaming

1

```
import tweepy
```

2

```
access_token =  
"808869773249384448-eq2ffryjCiSvWb9EG63eFvwlf9cPFnz"  
  
access_token_secret =  
"5sELYS8FSRDxgNxPR9PJk2DykQ48pU4oyolzTcdvXkmJL"  
  
consumer_key = "I6dlHIFHK3un2jG9KcRGINQSq"  
  
consumer_secret =  
"0KYTsuOwAFoh1Hil6J0EtpfdcSAkf1KVrZQzToYqMZgnsrnQNP"
```



# Method to Get Tweet Streaming

- 3      `auth = tweepy.OAuthHandler(consumer_key, consumer_secret)`
- 4      `class MyStreamListener(tweepy.StreamListener):`  
          `def on_status(self, status):`  
          `print(status.text)`
- 5      `l = MyStreamListener()`
- 6      `stream = tweepy.Stream(auth, l)`
- 7      `tweets = stream.filter(track=['got7'], async=True)`
- 8      `print(tweets)`



# Result Access Data From Twitter

```
# Filter Twitter Streams to capture data by the keywords:  
tweets = stream.filter(track=['got7'], async=True)
```

```
: print(tweets)
```

```
#GOT7  
#갓세븐  
#PresentYOU  
#Lullaby...  
RT @GOT7Official: GOT7 3rd Album <Present : YOU>;  
2018.09.17 MON 6PM
```

<https://t.co/LzT9t6jUcj>

RT @annngyPimberry: ไถทวิตเก็บยังมือกช.ตปท. จับผิดภาพที่ปล่อยออกมา บอกจินยองใส่เสื้อสีค่อนข้างเข้มไม่พาสเทลเหมือนเพื่อนๆ บางลั่ง相手を想うがゆえのすれ違い...

でも「あの時と、今と、その間」に  
二人の気持ちが変わらずに、  
新しい恋としてまた始まって良かった  
(\*^-^\*)..:.\*♡

チャラマクさんシリーズ  
切なくも楽しかったです。  
ありがとうございました... <https://t.co/LodStULfAh>  
RT @GOT7VotingTeam: 🏆 IDOL CHAMP 🏆

EMERGENCY 🔥 🔥 🔥 🔥

น้ำตาจะไหล  
รีชื่อเราด้วย !!



# Twitter Streaming

TIP

สำหรับการดึงข้อมูล Twitter นั้นเรา เราใส่ `async` เพื่อให้ยังทำงานต่อห้ายอย่างพร้อมๆกันได้ ถึงแม้จะไม่มี `filter` เรื่องที่เราต้องการก็ตามที และไม่สอดคล้องในการอ่านผ่าน `thread` ของเครื่อง

```
tweets = stream.filter(track=['got7'], async=True)
```

หลังจากรันเสร็จแล้วทุกครั้งหากเลิกใช้งาน หรือทำอย่างอื่นต่อ ควรหยุด `stream` ด้วยการเรียกใช้ method `disconnect()`

```
stream.disconnect()
```



# Homework Session 4

1. เราร่วมกับการ Input ข้อมูลจากแหล่งใดได้บ้าง ยกตัวอย่างมา 4 แหล่ง
2. หากเราต้องการข้อมูลที่ import เข้ามาในรูปแบบ csv โดยไฟล์ต้นฉบับมีจำนวน columns ถึง 100 columns แต่เราต้องการเรียกมาใช้งานแค่ 5 columns แรก ต้องเขียน code อย่างไร
3. ลักษณะรูปแบบไฟล์ที่เป็น flat file เป็นอย่างไร
4. รูปแบบข้อมูลแบบ JSON เป็นอย่างไร
5. หากเราอยากรู้ข้อมูลหนังเรื่อง The lord of the ring ต้องเขียน code อย่างไรบ้าง
6. ลองทำการ streaming ข้อมูลจาก twitter โดยใช้ filter ที่ตัวเองสนใจมา 1 เรื่อง



# Afternoon

---

Day-01



Boyd BigData RPG - Sorratat Sirirattanajakarin

1. Input Data
2. Numpy
3. Pandas
4. Visualization
5. Linear Regression Modeling
6. Project First Day



# The Creator of Numpy

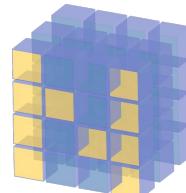
---



**Travis Oliphant**



**ANACONDA**  
Powered by Continuum Analytics®



**NumPy**



# Create Array

---

## Import numpy as np

ทุกครั้งเราจะเริ่มต้นด้วยสิ่งนี้เสมอ  
ดังนั้นครั้งต่อไปเราเรียกใช้ Package numpy เพียงแค่

np



# Create Array

Array ใน python จะมีรูปคล้าย List แต่เป็น type array([])  
โดยที่เราสามารถสร้าง array ในรูปแบบของ numpy ได้จาก list อย่างง่ายๆดังนี้

ทำการสร้าง list — **data1 = [1, 2, 3, 4, 5]**

ครอบ array จะ — **array1 = np.array(data1)**

ผลลัพธ์ — **array([1, 2, 3, 4, 5])**



# Create Array

การเรียกใช้งานยังคงเหมือนกับ list แต่เราสามารถใช้ method ทางคณิตศาสตร์ได้อิ่มมากจากการใช้ numpy package

**array1** = ( [ “Prayad” “Prayard” “Prayaad” “Prayadd” ] , , , )

“Prayad”

“Prayard”

“Prayaad”

“Prayadd”

**array1[0]**

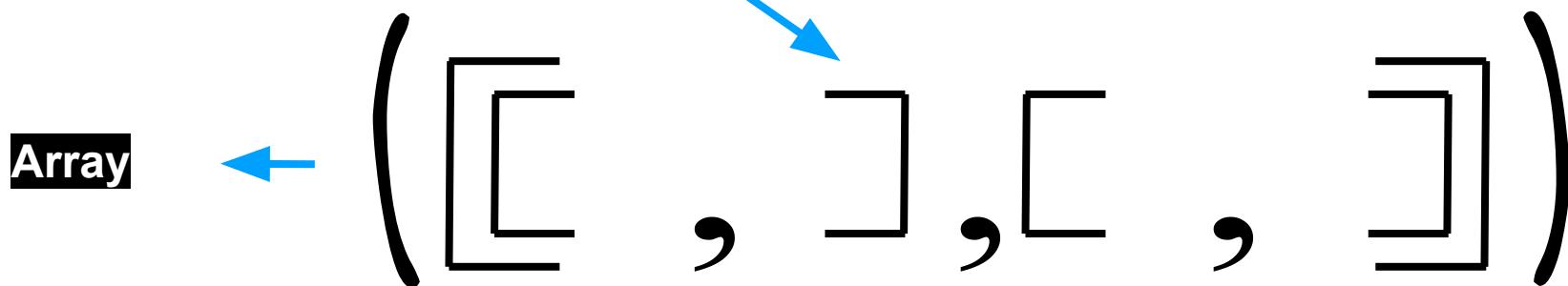
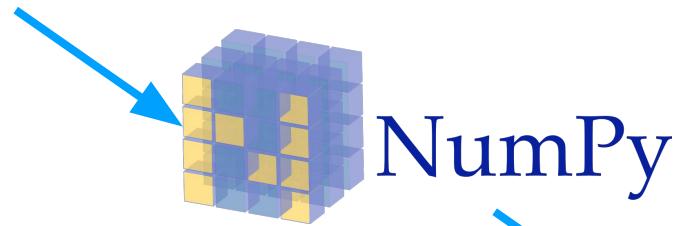
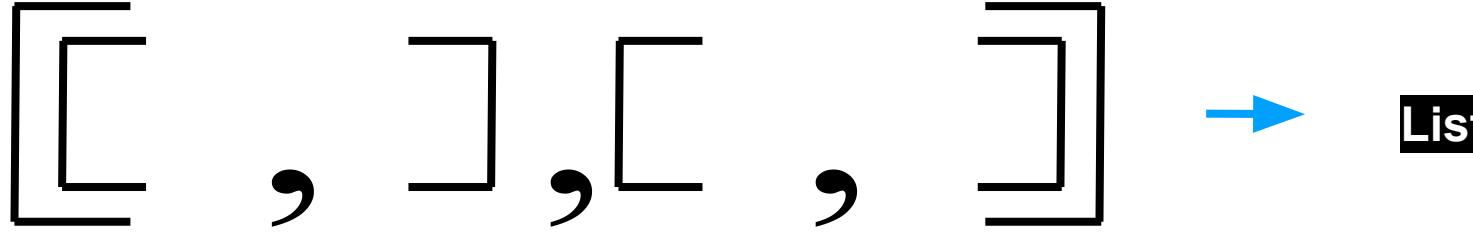
**array1[1]**

**array1[2]**

**array1[3]**



# Numpy Array





# Create Special Array

Numpy มี Method มากมายในการ generate ข้อมูลให้เราเลือกใช้ ยกตัวอย่างเช่น

| Method             | Examples  |
|--------------------|---|
| np.zeros(10)       | array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])   |
| np.zeros(3,6)      | array([[0., 0., 0., 0., 0., 0.],<br>[0., 0., 0., 0., 0., 0.],<br>[0., 0., 0., 0., 0., 0.]]) |
| np.ones(10)        | array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1.])   |
| np.linspace(0,1,5) | array([0. , 0.25, 0.5 , 0.75, 1. ])   |
| np.arange(5)       | array([0, 1, 2, 3, 4])  |



# Create Special Array

Numpy ยังสามารถ Generate ข้อมูล แบบ Random และ Set Seed ในการ Random ค่าได้อีกด้วย

| Method                                 | Examples   |
|--|--|
| <code>np.random.seed(42)</code>        | Set random seed on number 42                                   |
| <code>np.random.rand(3,10)</code>      | Random 3 array 10 elements on each array                       |
| <code>np.random.randint(0,10,5)</code> | Random int from [0,10] 5 number                                |
| <code>np.random.randn(3,10)</code>     | Random 3 array 10 elements on each array (Normal Distribution) |



# Examining Array

บางทีข้อมูลที่เราทำการเปิดมาหรือเก็บไปยัง variable มีจำนวนใหญ่มากๆ หากทำการ print คงไม่ใช่เรื่องดีแน่ ดังนั้นการที่เรารู้ก่อนว่า ข้อมูลภายในมีลักษณะอย่างไรย่อมดีกว่า แน่นอน !?

| Method     | Examples                        |
|------------|---------------------------------|
| .dtype     | Check type ของ element ใน array |
| .ndim      | Check dimension ของ array       |
| .shape     | Check รูปร่าง (row,column)      |
| len(array) | Check จำนวน element ชั้นนอกสุด  |
| .size      | Check total number of elements  |



# Understand Shape

สำคัญเลยคือการเข้าใจรูป่างของ Array ก่อนการใช้ method แปลงรูป่าง

```
array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
```



Shape: (10, )

```
array([[0., 1., 2., 3., 4., 5., 6., 7., 8., 9.],  
[0., 1., 2., 3., 4., 5., 6., 7., 8., 9.],  
[0., 1., 2., 3., 4., 5., 6., 7., 8., 9.]])
```



Shape: (3, 10)

```
array([[0.],  
[0.],  
[0.]])
```



Shape: (3, 1)

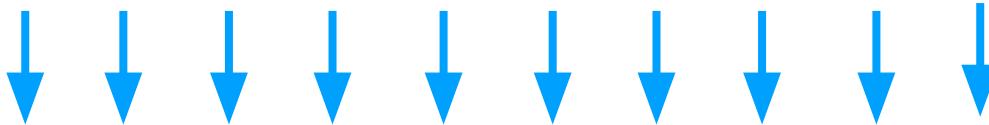
# สงสัยในม เกี่ยวกับ shape ระหว่าง !??

(10,) กับ (10,1)



# อันนี้ shape (10,) นะ !!

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```



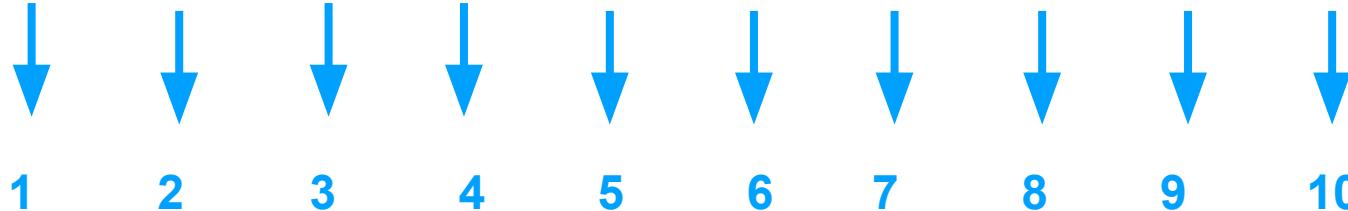
Elements: 1 2 3 4 5 6 7 8 9 10

สังเกตุใหม่ว่า axis 1 มีจำนวน column=0  
ดูได้จาก shape และ [] ที่มีแค่ชั้นเดียว นั่นหมาย<sup>ถึงว่า array นือยู่ลอยๆ แบบไม่ได้จัดเป็น</sup>  
column นั้นเอง แต่ก็มี ออย 10 elements นะ



# ส่วน...อันนี้ shape (10,1) นั่นเอง !!

```
array([[0],[1],[2],[3],[4],[5],[6],[7],[8],[9]])
```



สังเกตุใหม่ว่า axis 1 มีจำนวน column=10  
ดูได้จาก shape และ [] มีจำนวน 2 ชั้น ซึ่ง มี  
จำนวน [] ที่อยู่ด้านใน [] ทั้งหมด 10 elements  
เจ้าพกนี่ที่อยู่ด้านในเรารอเรียก columns นั่นเอง





# Reshaping

Reshaping คือการแปลงรูปทรงของ Array ให้อยู่ใน รูปทรงที่เราต้องการ

**np.arange(10, dtype=float)** → array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])



.reshape((2, 5)) → array([[0., 1., 2., 3., 4.],  
[5., 6., 7., 8., 9.]])



.shape → (2,5)



.reshape((5, 2)) → array([[0. 1.], [2. 3.], [6. 7.], [8. 9.]])



# New Axis

**a = np.array([0, 1])**

→ array([0, 1])

shape (2,0)

**b = a[:, np.newaxis]**

→ array([[0],  
[1]])

เพิ่ม [] มาอีกชั้น

**b.T**

→ array([[0, 1]])

(2,1) => (1,2)

**a.T**

→ array([0, 1])

ไม่เปลี่ยนแปลง  
shape



# Stacking Array

เหมือนการนำ array มาต่อกันนั่นเอง !!

```
a = np.array([0, 1])
```

```
b = np.array([2, 3])
```



```
ab = np.stack((a, b))
```



```
array([[0, 1], [2, 3]])
```

# Selection-01

Numpy





# Selection

---

คือการเลือก element ต่างๆ ใน array

**EX**

```
arr = array([[0,1],["a","b"],[2,3]])
```

ถ้าอยากได้ [0,1] ต้องทำอย่างไร ??

# Array มาจาก List ดังนั้นมันก็เรียกเหมือนกัน !!

```
arr = array([[0,1],["a","b"],[2,3]])
```

INDEX:

0

1

2

ทำการเรียก Index ที่ 0 จาก  
arr[0] หรือ arr[0, :] ได้ทั้งคู่



# Selection-02

Numpy





# Selection cont.

**EX**

```
arr = array([[0,1],["a","b"],[2,3]])
```

แล้วถ้าอยากได้ [1] ล่ะต้องทำอย่างไร ??

# ເວົາລະຄ່ອຍໆແກະໄປທີລະຫັ້ນ !!

```
arr = array([[0,1],["a","b"],[2,3]])
```

INDEX:

↓

0

↓

1

↓

2

ทำการเรียก Index ที่ 0 จาก  
arr[0][1] หรือ arr[0, 1] ได้ทั้งคู่



# Selection-03

Numpy





# Selection cont.

**EX**

```
arr = array([[ 0, 33,  2,  3,  4],  
            [ 5,  6,  7,  8,  9]])
```

แล้วถ้าอยากได้ [33, 2, 3] ล่ะ  
ต้องทำอย่างไร ??

# เราก็ทำได้ด้วย Slice

```
arr = array([[ 0, 33, 2, 3,  
[ 5, 6, 7, 8, 9]])
```

ทำการเรียก Index ที่ 0 element ที่ 1-3

```
arr[0,1:3]
```



# Selection-04

Numpy





# Selection cont.

**EX**

```
arr = array([[ 0, 33,  2,  3,  4],  
            [ 5,  6,  7,  8,  9]])
```

แล้วถ้าอยากได้ **[2, 3, 4],[7, 8, 9]** ล่ะ  
ต้องทำอย่างไร ??

# เราก็ทำได้ด้วย Slice

```
arr = array([[ 0, 33, 2, 3, 4],
```

```
[ 5, 6, 7, 8, 9]])
```

ทำการเรียก Index ที่ 0 และ 1  
element ที่ 2-4

```
arr[:,2:]
```



# Selection-05

Numpy





# Selection cont.

**EX**

```
arr = array([[ 0, 33,  2,  3,  4],  
            [ 5,  6,  7,  8,  9]])
```

แล้วถ้าอยากได้  $[0, 2, 4], [5, 7, 9]$  ล่ะ  
ต้องทำอย่างไร ??

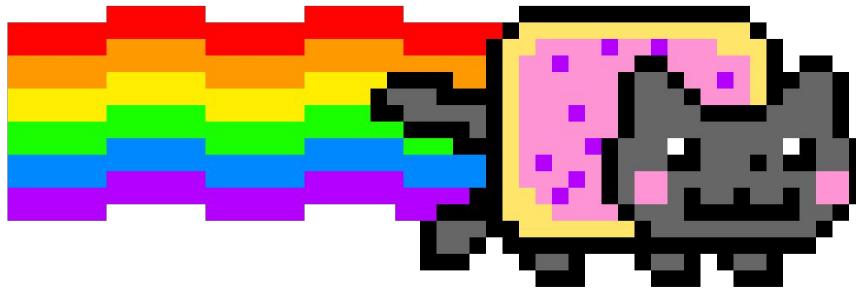
# เราก็ทำได้ด้วย Slice

```
arr = array([[ 0, 33,  2,  3,  4],  
            [ 5,  6,  7,  8,  9]])
```

ทำการเรียก Index ที่ 0 และ 1  
element ที่ 0,2,3

```
arr[:,[0,2,4]]
```





# Selection-06

Fancy Numpy





# Selection cont.

**EX**

```
arr = array([[ 0, 33,  2,  3,  4],  
            [ 5,  6,  7,  8,  9]])
```

แล้วถ้าอยากรอได้ค่า index ที่ 0  
แบบตัวเลขกลับหันหลังล่ะ ??

# เราก็ทำได้ด้วย Slice

```
arr = array([[ 0, 33,  2,  3,  4],
```

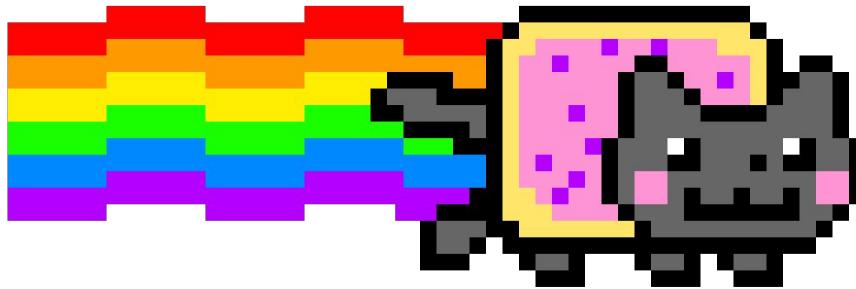
```
[ 5,  6,  7,  8,  9]])
```

ทำการเรียก Index ที่ 0

ทุก elements

```
arr[0,:,:-1]
```





# Selection-07

Fancy Numpy





# Selection cont.

**EX**

```
arr = array([[ 0, 33,  2,  3,  4],  
            [ 5,  6,  7,  8,  9]])
```

แล้วถ้าอยากรู้ค่าที่มากกว่า 5 ล่ะ  
ต้องทำอย่างไร ??

# เราก็ทำได้ด้วย Slice

```
arr = array([[ 0, 33, 2, 3, 4],  
            [ 5, 6, 7, 8, 9]])
```

ทำการเรียก Filter ดังนี้

```
arr[arr > 5]
```





# Logical Array

การเช็คค่าต่างๆใน array รวมถึงการแปลงค่าต่างๆเป็น boolean ก็ทำได้ด้วย operator ต่างๆ เช่น ==, != เป็นต้น

**EX**

```
names = array(['Bob', 'Joe', 'Will', 'Bob'])
```

Boolean

Logical

Assign

Unique

EX

```
names = array(['Bob', 'Joe', 'Will', 'Bob'])
```

Script

```
names == 'Bob'
```

Result

```
array([True, False, False, True])
```



# Logical

EX

```
names = array(['Bob', 'Joe', 'Will', 'Bob'])
```

Script

```
names[names != 'Bob']
```

Result

```
array(['Joe', 'Will'])
```



# Assign

## EX

```
names = array(['Bob', 'Joe', 'Will', 'Bob'])
```

## Script

```
names[names != 'Bob'] = 'Joe'
```

## Result

```
array(['Bob', 'Joe', 'Joe', 'Bob'])
```



# Unique

## EX

```
names = array(['Bob', 'Joe', 'Will', 'Bob'])
```

## Script

```
np.unique(names)
```

## Result

```
array(['Bob', 'Joe', 'Will'])
```

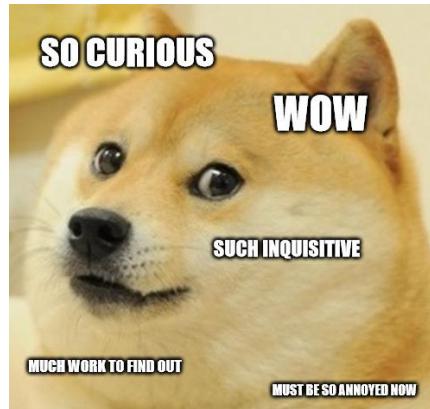




# Vectorization

เป็นการคำนวณ Vector ใน Array ซึ่งเป็นสามารถคำนวณได้ทั้ง คณิตศาสตร์ สถิติ และ การแก้ปัญหา Algorithms ต่างๆ

ก่อนอื่นมาเข้าใจกันก่อนว่า อะไรคือ Vector อะไรคือ Array  
เพื่อทบทวนก่อนลุยกันต่อ !!



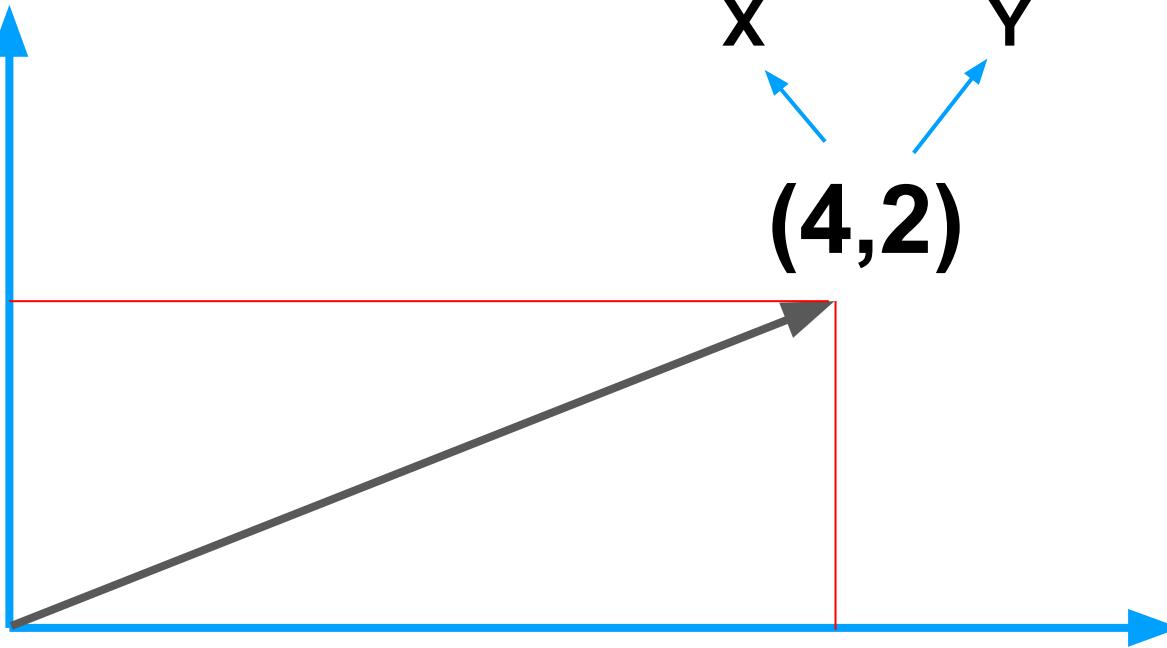
# Vector and Graph

Y-axis

2

X  
Y

(4,2)

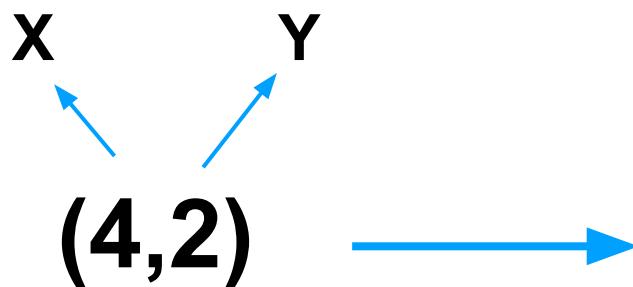


4

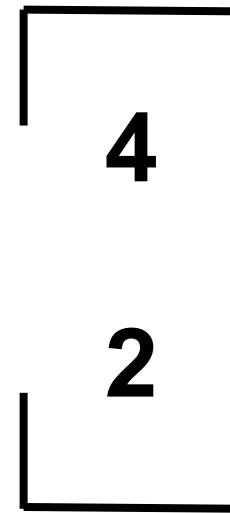
X-axis



# Vector and Graph



จุดบนเส้นจำนวน

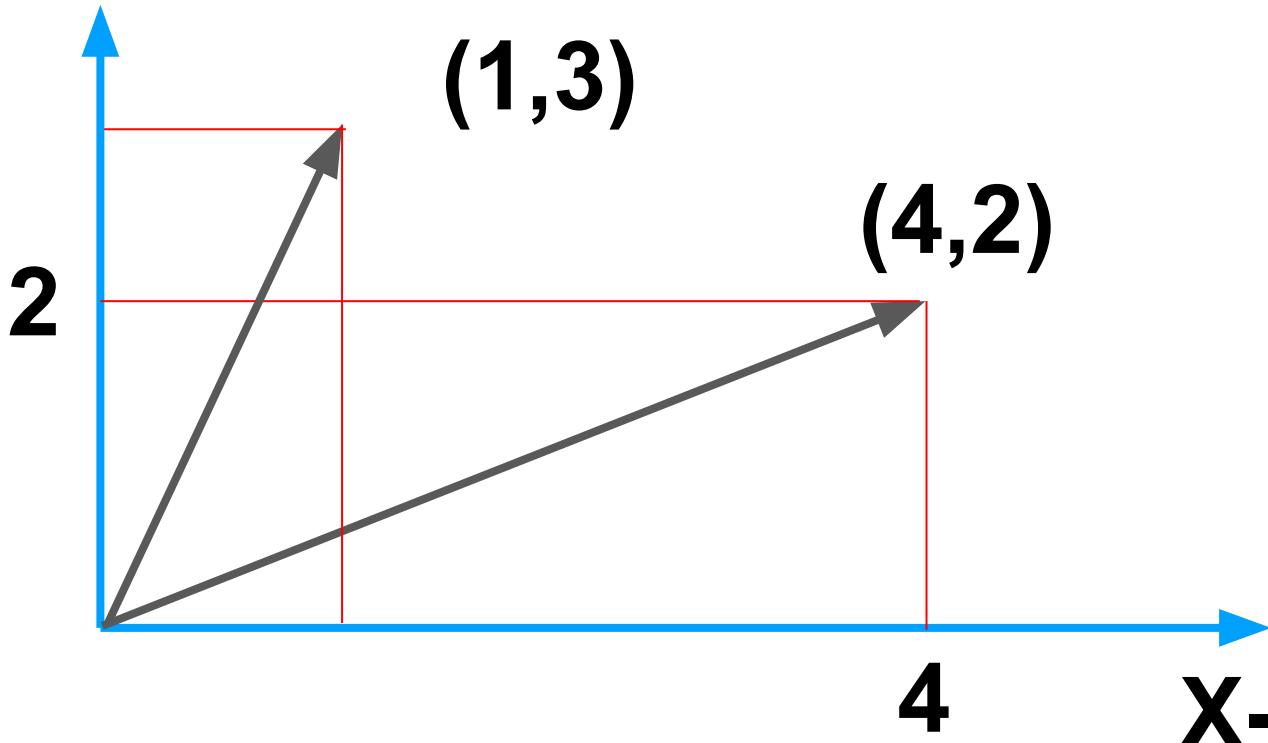


เขียนในรูป  
Matrix



# Vector and Graph

Y-axis



X-axis



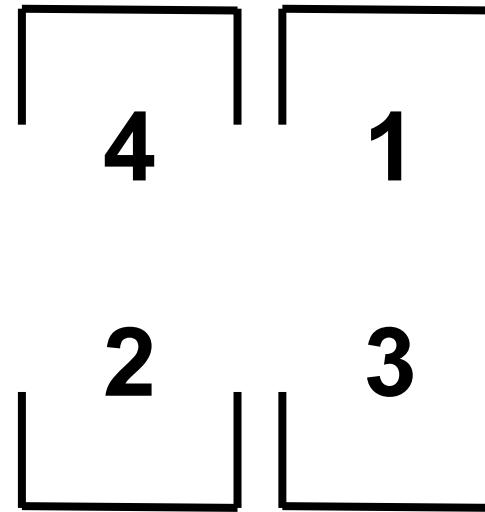
# Vector and Graph

(4,2)

Vector 1

(1,3)

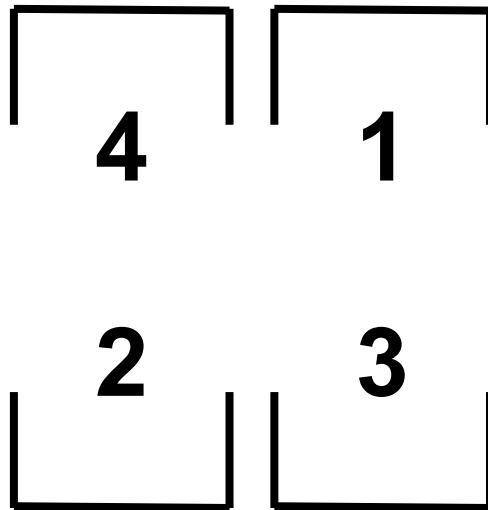
Vector 2



เขียนในรูป  
Matrix



# Vector and Graph



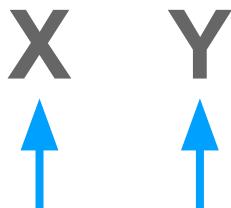
([ [4, 2],  
[1, 3] ])

เขียนในรูป  
Matrix

เขียนในรูป  
Array



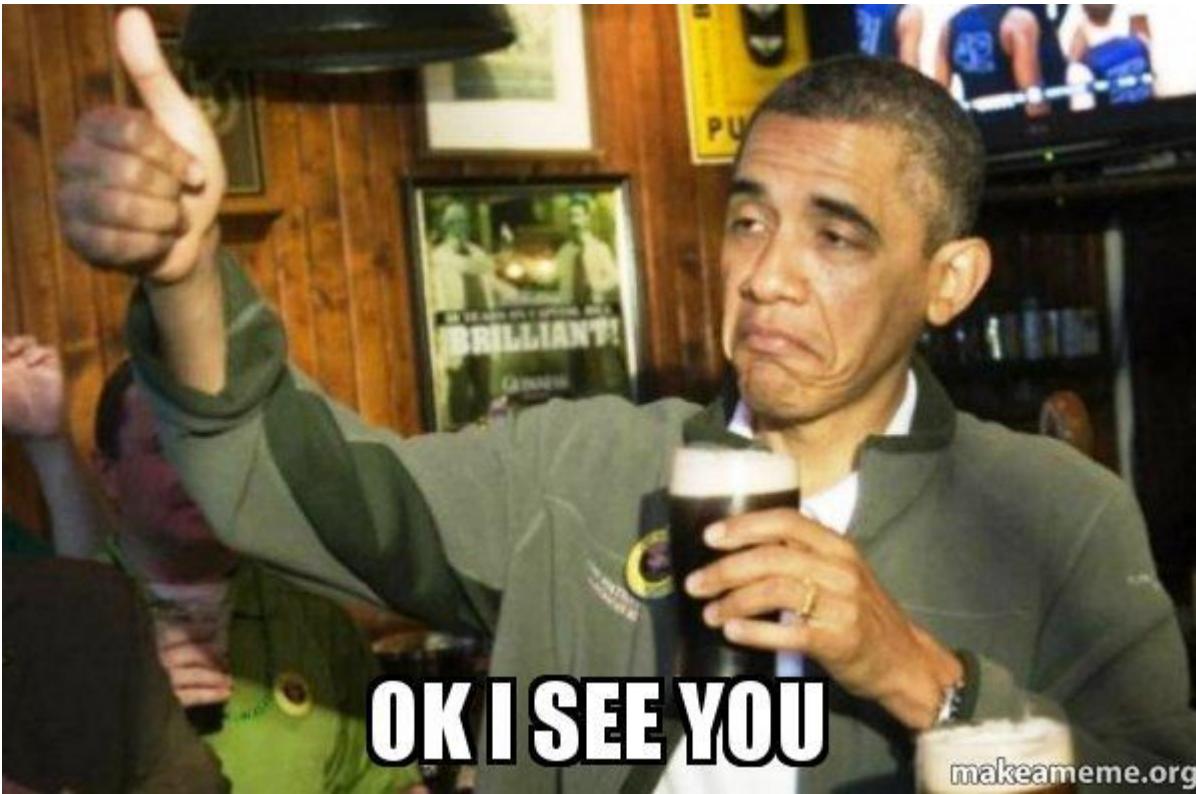
# Vector and Graph



([ [4, 2], → Vector 1

[1, 3] ]) → Vector 2





makeameme.org

Boyd BigData RPG - Sorratat Sirirattanajakarin



# Columns

**np.mean(axis=0)** → 2.5, 2.5

([ [4, 2],  
[1, 3] ])

# Means

# Rows

**np.mean(axis=1)**



3.0, 2.0



# Columns

**np.std(axis=0)**



# Standard Deviation

1.5, 0.5

([ [4, 2],  
[1, 3] ])

# Rows

**np.std(axis=1)**



1.0, 1.0



# Argmin

## Columns

`np.argmin(axis=0)` → 1, 0

([ [4, 2],  
[1, 3] ])

## Rows

`np.argmin(axis=1)`



1, 0



# Columns

**np.argmax(axis=0) → 0, 1**

([ [4, 2],  
[1, 3] ])

# Argmax

# Rows

**np.argmax(axis=1)**



0, 1



**Sum**

## Columns

**np.sum(axis=0)** → 5, 5

([ [4, 2],  
[1, 3] ])

## Rows

**np.sum(axis=1)**

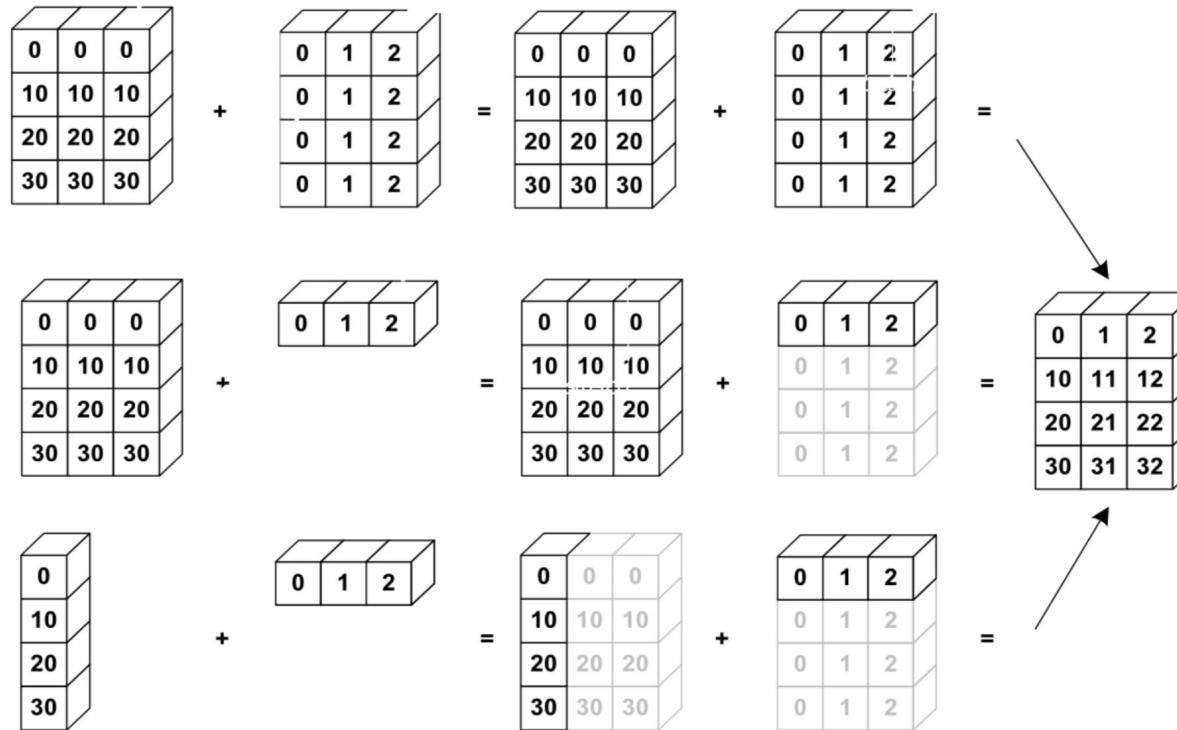


6, 4





# Broadcasting



# รูปแบบที่ 1

|    |    |    |
|----|----|----|
| 0  | 0  | 0  |
| 10 | 10 | 10 |
| 20 | 20 | 20 |
| 30 | 30 | 30 |

+

|   |   |   |
|---|---|---|
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |

=

|    |    |    |
|----|----|----|
| 0  | 0  | 0  |
| 10 | 10 | 10 |
| 20 | 20 | 20 |
| 30 | 30 | 30 |

+

|   |   |   |
|---|---|---|
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |

|    |    |    |
|----|----|----|
| 0  | 1  | 2  |
| 10 | 11 | 12 |
| 20 | 21 | 22 |
| 30 | 31 | 32 |



## รูปแบบที่ 2

|    |    |    |
|----|----|----|
| 0  | 0  | 0  |
| 10 | 10 | 10 |
| 20 | 20 | 20 |
| 30 | 30 | 30 |

+

|   |   |   |
|---|---|---|
| 0 | 1 | 2 |
|---|---|---|

=

|    |    |    |
|----|----|----|
| 0  | 0  | 0  |
| 10 | 10 | 10 |
| 20 | 20 | 20 |
| 30 | 30 | 30 |

+

|   |   |   |
|---|---|---|
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |

|    |    |    |
|----|----|----|
| 0  | 1  | 2  |
| 10 | 11 | 12 |
| 20 | 21 | 22 |
| 30 | 31 | 32 |



# รูปแบบที่ 3

|    |
|----|
| 0  |
| 10 |
| 20 |
| 30 |

|   |   |   |
|---|---|---|
| 0 | 1 | 2 |
|---|---|---|

+

|    |    |    |
|----|----|----|
| 0  | 0  | 0  |
| 10 | 10 | 10 |
| 20 | 20 | 20 |
| 30 | 30 | 30 |

=

|   |   |   |
|---|---|---|
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |
| 0 | 1 | 2 |

+

|    |    |    |
|----|----|----|
| 0  | 1  | 2  |
| 10 | 11 | 12 |
| 20 | 21 | 22 |
| 30 | 31 | 32 |



# Homework Session 5

1. จากโค้ดดังกล่าว จงหาค่าที่น้อยที่สุดของแต่ละ column ว่าอยู่ที่ตำแหน่งใด (Argmin)
  - a.  $X = np.random.randn(10, 4)$
2. จากโค้ดดังกล่าว จงหาค่าที่มากที่สุดของแต่ละ row ว่าอยู่ที่ตำแหน่งใด (Argmax)
  - a.  $Y = np.random.randn(5, 10)$
3. จากข้อ 1. จงหาค่า mean และ std ของแต่ละ column
4. จากข้อ 2. จงหาค่า mean และ std ของแต่ละ column



# Afternoon

---

Day-01



Boyd BigData RPG - Sorratat Sirirattanajakarin

1. Input Data
2. Numpy
3. Pandas
4. Visualization
5. Linear Regression Modeling
6. Project First Day



# The Creator of Pandas



**Wes McKinney**



กลุ่มเพื่อน BDFL





# Data Structures

Pandas ยังช่วยในการจัดการข้อมูล และคำนวณได้ง่ายกว่า Numpy เพราะเจ้า Pandas มีลักษณะเหมากับการดูข้อมูลหลายๆ Dimension (Multi-dimensional Array)

โดยรูปแบบของชนิดข้อมูลมี 2 ชนิดใหญ่ๆ ดังนี้

|   | age  | gender | job       | name  | height |
|---|------|--------|-----------|-------|--------|
| 0 | NaN  | F      | student   | alice | 165.0  |
| 1 | 26.0 | None   | student   | john  | 180.0  |
| 2 | NaN  | M      | student   | eric  | 175.0  |
| 3 | 58.0 | None   | manager   | paul  | NaN    |
| 4 | 33.0 | M      | engineer  | peter | NaN    |
| 5 | 44.0 | F      | scientist | julie | 171.0  |

0            19  
0            22  
0            33

Name: age, dtype: int64

Data Frame

Series



# Data Structures

รูปแบบของชนิดของข้อมูลในแต่ละ Dimension เราสามารถแปลงไปมาให้อยู่ในรูปแบบของ

**1,2,3,4**

**'Prayad'**

**3.41432**

**'PrayadZa'**

**Integer**

**String**

**Float**

**Object**



# Create Data Frame

---

มี 2 แบบด้วยกันคือ สร้างจาก List และสร้าง จาก Dict

[.....]

From List

{.....}

From Dict

# From List

1. สร้าง list ของ columns ก่อนว่าอย่างไรก็ได้กี่ Dimension

```
columns = ['name', 'age', 'gender', 'job']
```

2. สร้าง List ของแต่ละ Row ID ว่ามีข้อมูลอะไรบ้าง ในแต่ละ Dimension

```
pd.DataFrame([['alice', 19, "F", "student"],  
['john', 26, "M", "student"]],columns=columns)
```



# DataFrame From List

|   | name  | age | gender | job     |
|---|-------|-----|--------|---------|
| 0 | alice | 19  | F      | student |
| 1 | john  | 26  | M      | student |



# From Dict

1. สร้าง Dict โดย Key เปรียบเสมือน ชื่อ Column หรือ Dimension นั้นๆ และ Values คือ List ของค่าต่างๆ เเรียงกันไปตาม RowID

```
pd.DataFrame(dict(
```

```
name=['prawitt', 'prayad'],  
age=[70, 65],  
gender=['M', 'F'],  
job=['chef', 'PM'] ))
```



# DataFrame From Dict

|   | name    | age | gender | job  |
|---|---------|-----|--------|------|
| 0 | prawitt | 70  | M      | chef |
| 1 | prayad  | 65  | F      | PM   |





# Combine Data Frame

การนำ Data Frame มาต่อกัน

|   | name  | age | gender | job     |   | name    | age | gender | job  |
|---|-------|-----|--------|---------|---|---------|-----|--------|------|
| 0 | alice | 19  | F      | student | 0 | prawitt | 70  | M      | chef |
| 1 | john  | 26  | M      | student | 1 | prayad  | 65  | F      | PM   |

df1

df2

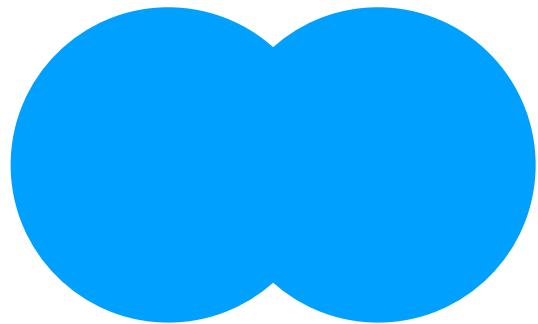
**df1.append(df2)**

# Combine DataFrame Result

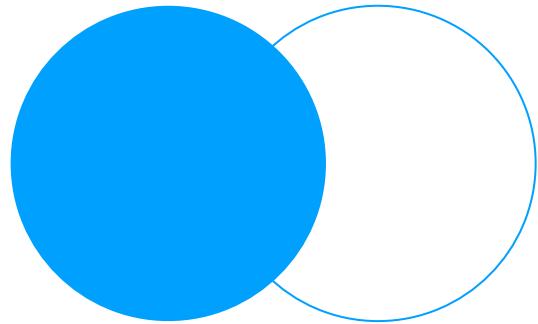
|   | name    | age | gender | job     |
|---|---------|-----|--------|---------|
| 0 | alice   | 19  | F      | student |
| 1 | john    | 26  | M      | student |
| 0 | prawitt | 70  | M      | chef    |
| 1 | prayad  | 65  | F      | PM      |



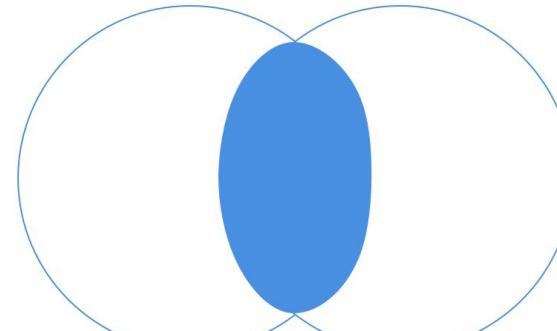
# Join DataFrame



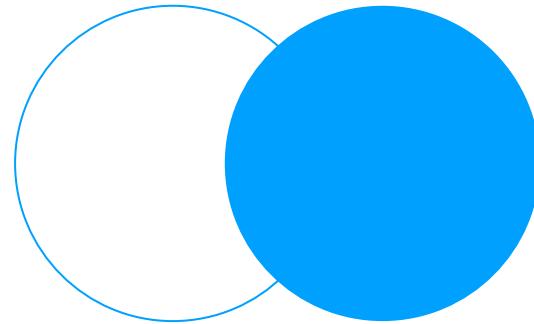
**Outer Join**



**Left Join**



**Inner Join**



**Right Join**



# Outer join

|   | name    | age |
|---|---------|-----|
| 0 | alice   | 19  |
| 1 | john    | 26  |
| 2 | prawitt | 70  |
| 3 | prayad  | 65  |

df1

|   | name    | job  |
|---|---------|------|
| 0 | prawitt | chef |
| 1 | prayad  | PM   |
| 2 | manee   | VP   |
| 3 | mana    | CEO  |

df2





# Outer join

```
pd.merge(df1, df2, on="name", how='outer')
```

CODE

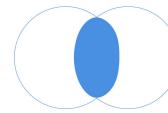




# Outer join

|   | name    | age | job  |
|---|---------|-----|------|
| 0 | alice   | 19  | NaN  |
| 1 | john    | 26  | NaN  |
| 2 | prawitt | 70  | chef |
| 3 | prayad  | 65  | PM   |
| 4 | manee   | NaN | VP   |
| 5 | mana    | NaN | CEO  |





# Inner join

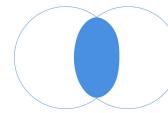
|   | name    | age |
|---|---------|-----|
| 0 | alice   | 19  |
| 1 | john    | 26  |
| 2 | prawitt | 70  |
| 3 | prayad  | 65  |

df1

|   | name    | job  |
|---|---------|------|
| 0 | prawitt | chef |
| 1 | prayad  | PM   |
| 2 | manee   | VP   |
| 3 | mana    | CEO  |

df2



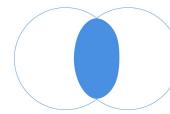


Inner join

```
pd.merge(df1, df2, on="name")
```

CODE

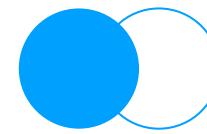




# Inner join

|   | name    | age | job  |
|---|---------|-----|------|
| 0 | prawitt | 70  | chef |
| 1 | prayad  | 65  | PM   |





# Left join

|   | name    | age |
|---|---------|-----|
| 0 | alice   | 19  |
| 1 | john    | 26  |
| 2 | prawitt | 70  |
| 3 | prayad  | 65  |

df1

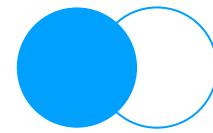
Left

|   | name    | job  |
|---|---------|------|
| 0 | prawitt | chef |
| 1 | prayad  | PM   |
| 2 | manee   | VP   |
| 3 | mana    | CEO  |

df2

Right



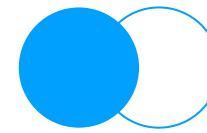


# Left join

```
pd.merge(df1, df2, on="name", how='left')
```

CODE

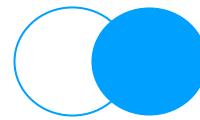




# Left join

|   | name    | age | job  |
|---|---------|-----|------|
| 0 | alice   | 19  | NaN  |
| 1 | john    | 26  | NaN  |
| 2 | prawitt | 70  | chef |
| 3 | prayad  | 65  | PM   |





# Right join

name age

|   |         |    |
|---|---------|----|
| 0 | alice   | 19 |
| 1 | john    | 26 |
| 2 | prawitt | 70 |
| 3 | prayad  | 65 |

df1

Left

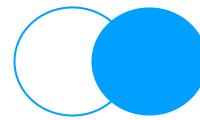
name job

|   |         |      |
|---|---------|------|
| 0 | prawitt | chef |
| 1 | prayad  | PM   |
| 2 | manee   | VP   |
| 3 | mana    | CEO  |

df2

Right



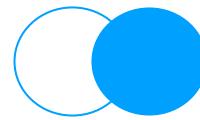


Right join

```
pd.merge(df1, df2, on="name", how='right')
```

CODE





# Right join

|   | name    | age | job  |
|---|---------|-----|------|
| 0 | prawitt | 70  | chef |
| 1 | prayad  | 65  | PM   |
| 2 | manee   | NaN | VP   |
| 3 | mana    | NaN | CEO  |





# Reshaping

ແນ່ນອນຕາມຊື່ອເລຍ Pandas ກີ່ສາມາຮາດເປົ້າຢັ້ງແປ່ງປົງກຳນະນັດ

ลด

Dimension

Melting

เพิ่ม

Dimension

Pivoting

# Melting

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df



# Melting

```
pd.melt(df, id_vars="name",  
var_name="variable", value_name="value")
```



# Melting

|   | name  | variable | value |
|---|-------|----------|-------|
| 0 | alice | age      | 19    |
| 1 | john  | age      | 26    |
| 2 | eric  | age      | 22    |
| 3 | paul  | age      | 58    |
| 4 | peter | age      | 33    |
| 5 | julie | age      | 44    |
| 6 | alice | gender   | F     |
| 7 | john  | gender   | M     |
| 8 | eric  | gender   | M     |

Variable แต่เดิมเคยเป็น Dimension ต่างๆ แต่หลังจากการ Melt ก็ลด Dimension ลงมาเหลือเพียง Dimension เดียว โดยยังคงค่าของเดิมไว้ที่อีก Dimension นึงที่ชื่อ Value

จะเห็นว่า df จากเดิม มี 4 Dimension ตอนนี้เหลือเพียง 3 Dimension เท่านั้น แต่จำนวน Row ก็เปลี่ยนขึ้นมาตามด้วยเช่นกัน



# Pivoting

|   | name  | variable | value |
|---|-------|----------|-------|
| 0 | alice | age      | 19    |
| 1 | john  | age      | 26    |
| 2 | eric  | age      | 22    |
| 3 | paul  | age      | 58    |
| 4 | peter | age      | 33    |
| 5 | julie | age      | 44    |
| 6 | alice | gender   | F     |
| 7 | john  | gender   | M     |
| 8 | eric  | gender   | M     |



df



# Pivoting

```
df.pivot(index='name', columns='variable',  
         values='value')
```



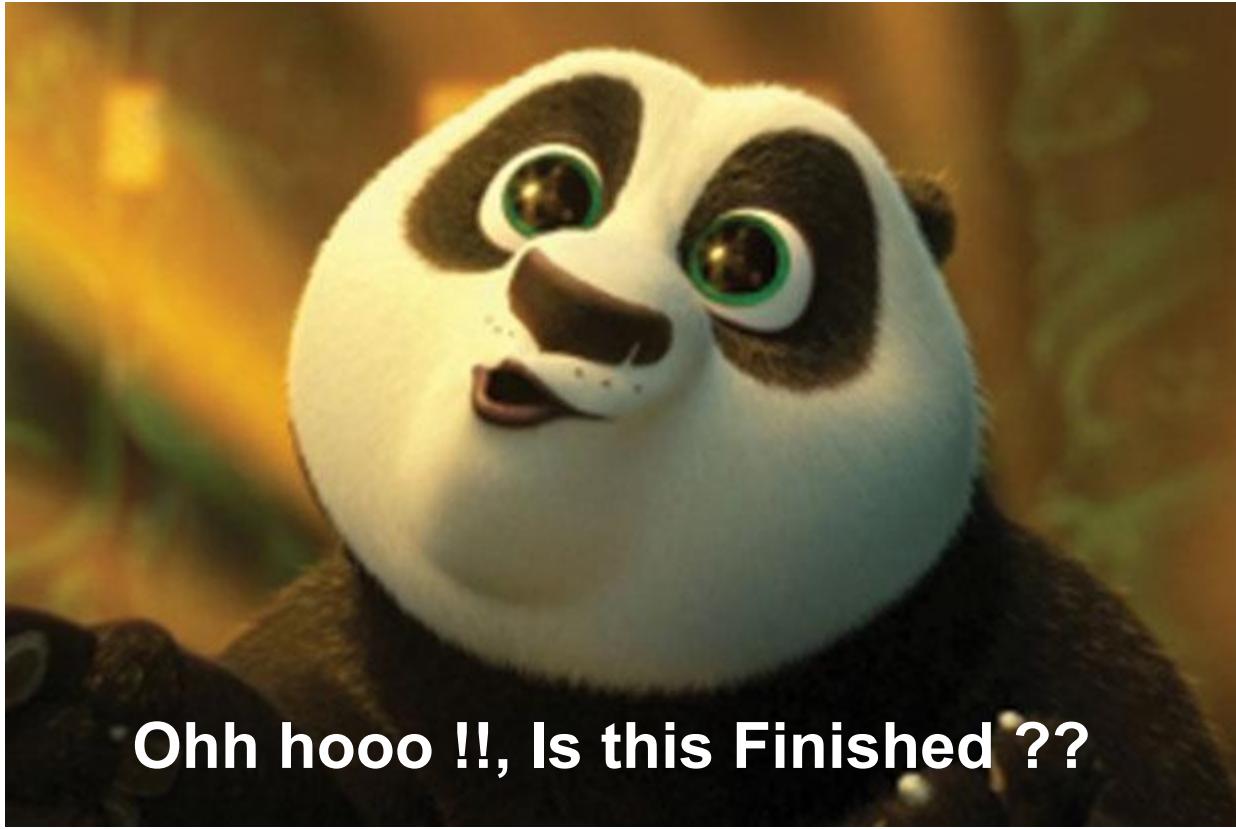
# Pivoting

| variable | age | gender | job       |
|----------|-----|--------|-----------|
| name     |     |        |           |
| alice    | 19  | F      | student   |
| eric     | 22  | M      | student   |
| john     | 26  | M      | student   |
| julie    | 44  | F      | scientist |
| paul     | 58  | F      | manager   |
| peter    | 33  | M      | engineer  |

Variable ตอนนี้กระจายออกมายแยกเป็น columns โดยค่าต่างๆที่เคยอยู่ใน Column Value ตอนนี้กระจายออกไปอยู่ตาม Dimension เดิมที่มันเคยอ้างถึงมาก่อน

และแน่นอนวิธีนี้ทำให้ Dimension กลับเพิ่มขึ้นมากกว่าเดิมจาก 3 กลายเป็น 4 Dimensions





Ohh hooo !!, Is this Finished ??





# Useful Method for Pandas

Check first 5 rows

`.head()`

Check last 5 rows

`.tail()`

Check  
Information of  
DataFrame

`.info()`

Check Index of  
DataFrame

`.index`

Check Shape of  
DataFrame

`.shape`

Check type of  
each column

`.dtypes`

Check Columns  
name

`.columns`

Transform to  
Array

`.values`



# Selection

เช่นเดียวกับ Numpy การเลือก Row และ Column ที่เราต้องการนำไปใช้งานจาก DataFrame ก็สามารถทำได้ อย่างง่ายดาย และดูเข้าใจง่ายกับรูปแบบของ Array เลียอีก !!



# Selection-01

pandas





# Selection

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df

ถ้าอยากได้ column  
age  
ต้องทำอย่างไร ?

# คำสั่งใช้งานง่ายมากๆ !!

## df.age หรือ df['age'] ก็ได้



# Selection-02

pandas





# Selection

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df

ถ้าอยากได้ column  
age และ job  
ต้องทำอย่างไร ?

# ลองเพิ่ม Array เข้าไปอีกชั้นหนึ่ง ก็ได้แล้ว !!

```
df[['age','job']]
```



# Selection-03

pandas





# Selection

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df

ถ้าอยากได้  
row ที่ 2-4  
ต้องทำอย่างไร ?

# คำสั่งใช้งานง่ายมากๆ !!

## df.iloc[1:5]

### ใช้ iloc เพื่อทำการเข้าถึงแบบ index



# Selection-04

pandas





# Selection

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df

ถ้าอยากได้  
row ที่ 2-4  
แต่เฉพาะ column  
gender กับ job ล่ะ  
ต้องทำอย่างไร ?

# คำสั่งใช้งานง่ายมากๆ !!!!!!

```
df.iloc[1:5, 1:3]
```

ใช้ iloc เพื่อทำการเข้าถึงแบบ index  
[row:column]



# Selection-05

pandas





# Selection

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df

ถ้าอยากได้  
ข้อมูลที่ age > 30  
ต้องทำอย่างไร ?

คำสั่งใช้งานง่ายมากๆ !!!!!!!

`df[df.age > 20]`

คุ้นๆ ใหม่เอี่ย คล้ายๆ Numpy เลย



# Selection-06

pandas





# Selection

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df

ถ้าอยากรู้  
ข้อมูลที่ age > 30  
และ gender เป็น  
ผู้ชายทั้งหมด  
ต้องทำอย่างไร ?

คำสั่งใช้งานง่ายมากๆ !!!!!!!

```
df[(df.age > 20) &&  
     (df.gender == 'M')]
```

นำ && มาต่อเพื่อใช้แทน and ได้ด้วยนะ  
หากต้องการ or ก็ใช้ | แทน



# Selection-07

pandas





# Selection

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |

df

ถ้าอยากได้ค่าเฉพาะลงกัน ขอให้ใน job มีค่า manager และ engineer ต้องทำอย่างไร ?

คำสั่งใช้งานง่ายมากๆ !!!!!!!!!!!!!!!

```
df[df.job.isin(['manager',  
                'engineer'])]
```

ใช้ iloc เพื่อทำการเข้าถึงแบบ index  
[row:column]





# Sort Values

ในเรื่องของข้อมูล หากเราต้องการข้อมูล ที่ต้องการแล้ว บางทีการเรียงลำดับข้อมูลก็เป็นเรื่องที่ทำให้เราเข้าใจ และเห็นภาพของข้อมูลได้ง่ายขึ้น และง่ายขึ้นไปอีกในการต่อยอดทำ Visualization หรือ Filter ในข้อมูลที่ต้องการไปใช้ต่อ



# สมมติว่าเรามีข้อมูลตั้งต้นแบบนี้

|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |



# อยากรีบค่า column ในนก็ใส่ไปได้เลย

```
df.sort_values(by='age')
```

เราก็จะได้ตารางที่เรียบค่าให้  
จากน้อยไปมาก ที่ column age



อยากเรียนค่าแบบมากไปน้อยล่ะ

```
df.sort_values(by='age',  
               ascending=False)
```

เท่านี้ก็เรียนจากมากไปน้อยแล้ว



อยากรจัดเรียงหลาย columns ละ ?? ก็ทำได้

`df.sort_values(`

`by=['age','height'])`

เท่านี้ก็เรียงจากมากไปน้อยแล้ว





# Statistical Description

ในทางสถิติการสำรวจค่าทางสถิติช่วยให้เราเข้าใจการกระจายตัวของข้อมูลรวมถึงสภาพของ outlier ได้อย่างง่ายๆ ก่อนการนำไปใช้งานต่อ

Pandas



# ค่าทางสถิติที่เราควรทราบคร่าวๆ

## df.describe()

ค่า default ที่ได้จะออกมาเฉพาะ  
Column ที่มีค่าเป็น numeric



|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |

**df.describe()**



|       | age       | height     |
|-------|-----------|------------|
| count | 6.000000  | 4.000000   |
| mean  | 33.666667 | 172.750000 |
| std   | 14.895189 | 6.344289   |
| min   | 19.000000 | 165.000000 |
| 25%   | 23.000000 | 169.500000 |
| 50%   | 29.500000 | 173.000000 |
| 75%   | 41.250000 | 176.250000 |
| max   | 58.000000 | 180.000000 |

มีข้อมูลเพียง 2 columns เท่านั้นที่สามารถ  
ดูข้อมูลแบบ Quartile ได้



|              | <b>age</b> | <b>height</b> |                               |
|--------------|------------|---------------|-------------------------------|
| <b>count</b> | 6.000000   | 4.000000      | จำนวน rows ของข้อมูล          |
| <b>mean</b>  | 33.666667  | 172.750000    | ค่าเฉลี่ยใน column นั้น       |
| <b>std</b>   | 14.895189  | 6.344289      | ค่า std ใน column นั้น        |
| <b>min</b>   | 19.000000  | 165.000000    | ค่าน้อยที่สุดใน column นั้น   |
| <b>25%</b>   | 23.000000  | 169.500000    | ข้อมูล ณ 25% แรกมีค่าเท่าไหร่ |
| <b>50%</b>   | 29.500000  | 173.000000    | ข้อมูล ณ 50% แรกมีค่าเท่าไหร่ |
| <b>75%</b>   | 41.250000  | 176.250000    | ข้อมูล ณ 75% แรกมีค่าเท่าไหร่ |
| <b>max</b>   | 58.000000  | 180.000000    | ค่ามากที่สุดใน column นั้น    |



# แล้วถ้าอยากได้ Stat ของ Column ที่มีชนิด ข้อมูลนอกเหนือจาก Numeric ล่ะ



# df.describe(include=['object'])

|        | gender | job     | name |
|--------|--------|---------|------|
| count  | 6      | 6       | 6    |
| unique | 2      | 4       | 6    |
| top    | F      | student | paul |
| freq   | 3      | 3       | 1    |



จำนวน rows ของข้อมูล



บอกจำนวน Unique



ค่าที่พบมากที่สุด



ค่าที่พบมากที่สุดพบกี่ครั้ง



แล้วถ้า !!  
อยากได้ข้อมูลที่เฉพาะเจ็บไปอีก



```
df.groupby("job")["age"].mean()
```

```
job
engineer      33.000000
manager       58.000000
scientist     44.000000
student        22.333333
Name: age, dtype: float64
```

ผลลัพธ์ Return ค่าอายุเฉลี่ย  
ในแต่ละกลุ่มของ Job ต่างๆกัน





# Duplicate

ข้อมูลที่เรามีบางทีก็ไม่ได้ Clean ไปซะหมดการ Check ว่ามีค่าไหน ซ้ำบ้างก็ช่วยได้  
เยอะเลย และช่วยในการคัดกรองก่อนนำไปทำอย่างอื่นๆต่อ



# ทำการแสดงค่า row ที่เกิดการซ้ำอว客家

**df[df.duplicated()]**

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |
| 6 | 33  | M      | engineer  | peter |



|   | age | gender | job      | name  |
|---|-----|--------|----------|-------|
| 6 | 33  | M      | engineer | peter |



# ทำการ ลบแถวที่ซ้ำออกไปง่ายๆโดยใช้ ..!

## df.drop\_duplicates()

|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |
| 6 | 33  | M      | engineer  | peter |



|   | age | gender | job       | name  |
|---|-----|--------|-----------|-------|
| 0 | 19  | F      | student   | alice |
| 1 | 26  | M      | student   | john  |
| 2 | 22  | M      | student   | eric  |
| 3 | 58  | F      | manager   | paul  |
| 4 | 33  | M      | engineer  | peter |
| 5 | 44  | F      | scientist | julie |





# Missing Value

Missing Value คือค่าที่ขาดหายไปจากตาราง ซึ่งทำให้ข้อมูลมีความน่าเชื่อถือน้อยลง และนำไปใช้งานได้ยากขึ้น หากเราสามารถจัดการ Missing Value ได้ก็จะทำให้ เราเข้าใจ ข้อมูลที่เรามี และความสัมพันธ์อื่นๆ ได้ง่ายขึ้น



# ทำการแสดงค่า row ที่เกิดการซ้ำอว客家



**df[df.height.notnull()]**

|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |



|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 5 | 44  | F      | scientist | julie | 171.0  |

BigData RPG  
 Boyd



คล้ายๆกับ Duplicate เลย  
หากเราไม่ต้องการใช้ Row  
ที่มี Missing Value  
เราก็สามารถทำได้โดยการ Drop Row  
ที่มี Missing Value ทั้งไปซะ



# ทำการ drop row ที่มี Missing Value



## df.dropna()

|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |



|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 5 | 44  | F      | scientist | julie | 171.0  |

BigData RPG  
 Boyd



แล้วถ้าอยากรอ  
Drop Row  
ที่ทุก column  
มี Missing value แทน  
ควรใช้คำสั่งอะไรดี !!!



# ง่ายๆแค่นี้เอง ^^



## df.dropna(how='all')

|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |



|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |



การจัดการกับ Missing Value  
เราไม่ได้มีเพียงแค่กำจัดออกนะ !!

ใจรဟน์ด้วยบ้าง !!





เพราะ ในชีวิตจริงข้อมูลที่เรามีคงไม่ได้สวยหรู  
 และมีมากมายให้เล่นกันอย่างหนำใจ  
 จะลบออกเท่าไหร่ก็ได้ จริงไหม !!

จริงๆแล้วข้อมูลเรามีแค่...  
 น้อยเดียวเอง !





# มาทำการแทนค่า Missing Value กันดีกว่า

เราเรียกว่า Imputation !



# ค่ารายราน้อยนะ



**df.ix[df.height.isnull(),**

**"height"] =**

**df["height"].mean()**



# រូបីលេ Step នេះ



|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.0  |
| 1 | 26  | M      | student   | john  | 180.0  |
| 2 | 22  | M      | student   | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |

dataframe



|   | age | gender | job      | name  | height |
|---|-----|--------|----------|-------|--------|
| 3 | 58  | F      | manager  | paul  | NaN    |
| 4 | 33  | M      | engineer | peter | NaN    |

dataframe

**df.ix[df.height.isnull()]**



# Step ต่อมา..



|   | age | gender | job      | name  | height |
|---|-----|--------|----------|-------|--------|
| 3 | 58  | F      | manager  | paul  | NaN    |
| 4 | 33  | M      | engineer | peter | NaN    |

dataframe



3    NaN  
4    NaN

Name: height, dtype: float64

series

```
df.ix[df.height.isnull(), "height"]
```



# Step สุดท้าย



```
3      NaN  
4      NaN  
Name: height, dtype: float64
```

series



```
3      172.75  
4      172.75  
Name: height, dtype: float64
```

series

```
df.ix[df.height.isnull(), "height"]
```

```
= df["height"].mean()
```



# Check รอบสุดท้าย



|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.00 |
| 1 | 26  | M      | student   | john  | 180.00 |
| 2 | 22  | M      | student   | eric  | 175.00 |
| 3 | 58  | F      | manager   | paul  | 172.75 |
| 4 | 33  | M      | engineer  | peter | 172.75 |
| 5 | 44  | F      | scientist | julie | 171.00 |





# Rename

บางทีข้อมูลมาเพี้ยนๆ เราต้องการแปลงค่าก่อนเพื่อนำไปใช้ต่อได้ง่ายขึ้น โดยปกติแล้ว  
เราจะแปลงกันอยู่ประมาณ 2 ส่วนหลักๆ คือ

- ค่าใน Column
- ชื่อของ Column



# การแปลงค่าใน Column

|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.00 |
| 1 | 26  | M      | student   | john  | 180.00 |
| 2 | 22  | M      | student   | eric  | 175.00 |
| 3 | 58  | F      | manager   | paul  | 172.75 |
| 4 | 33  | M      | engineer  | peter | 172.75 |
| 5 | 44  | F      | scientist | julie | 171.00 |

สมมติ student  
อยากเป็น นายก กัน  
ทำอย่างไรดีน้า !?



# การแปลงค่าใน Column

ใช้ loc เพื่อเข้าถึง  
[row,column]

เข้าถึง row ที่มีคำว่า  
student

เข้าถึง column job

```
df.loc[df.job == 'student', 'job']
```

= 'PM'



Assign ค่าใหม่ชื่อ PM



# เปลี่ยนแล้ว !!

|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | PM        | alice | 165.0  |
| 1 | 26  | M      | PM        | john  | 180.0  |
| 2 | 22  | M      | PM        | eric  | 175.0  |
| 3 | 58  | F      | manager   | paul  | NaN    |
| 4 | 33  | M      | engineer  | peter | NaN    |
| 5 | 44  | F      | scientist | julie | 171.0  |



# อยากแปลงชื่อ Column Name บ้าง

|   | age | gender | job       | name  | height |
|---|-----|--------|-----------|-------|--------|
| 0 | 19  | F      | student   | alice | 165.00 |
| 1 | 26  | M      | student   | john  | 180.00 |
| 2 | 22  | M      | student   | eric  | 175.00 |
| 3 | 58  | F      | manager   | paul  | 172.75 |
| 4 | 33  | M      | engineer  | peter | 172.75 |
| 5 | 44  | F      | scientist | julie | 171.00 |

จาก Column ชื่อ  
name แปลงเป็น  
ชื่อ nickname



# วิธีเปลี่ยนชื่อ Column Name

ตารางที่เราต้องการแปลงค่า



เรียกใช้ method rename



**df.rename**

```
(columns={'name':'nickname'})
```



ชื่อ column name เก่า



ชื่อ column name ใหม่



# เปลี่ยนแล้ว !!

|   | age | gender | job        | nickname | height |
|---|-----|--------|------------|----------|--------|
| 0 | 19  | F      | etudiant   | alice    | 165.0  |
| 1 | 26  | M      | etudiant   | john     | 180.0  |
| 2 | 22  | M      | etudiant   | eric     | 175.0  |
| 3 | 58  | F      | manager    | paul     | NaN    |
| 4 | 33  | M      | ingenieur  | peter    | NaN    |
| 5 | 44  | F      | scientific | julie    | 171.0  |



# Homework Session 6

1. ทำการเลือก dataset จาก UCI Machine Learning มา 1 data set และทำการค้นข้อมูลดังต่อไปนี้
  - a. Column Name
  - b. Column ใดบ้างมีค่า Numeric
  - c. ทำการค้นหาค่า mean จาก Column ที่มีค่า Numeric
2. จากข้อมูลดังกล่าวจะงดตอบคำถามต่อไปนี้

|   | age  | gender | job       | name  | height |
|---|------|--------|-----------|-------|--------|
| 0 | NaN  | F      | student   | alice | 165.0  |
| 1 | 26.0 | None   | student   | john  | 180.0  |
| 2 | NaN  | M      | student   | eric  | 175.0  |
| 3 | 58.0 | None   | manager   | paul  | NaN    |
| 4 | 33.0 | M      | engineer  | peter | NaN    |
| 5 | 44.0 | F      | scientist | julie | 171.0  |

## Code Create Table

```
df_ex = users_outer.copy()  
df_ex.ix[[0, 2], "age"] = None  
df_ex.ix[[1, 3], "gender"] = None
```



# Homework Session 6

- a. จากข้อ 2 ทำการเขียน function เพื่อทำการเติม ค่า mean ให้กับค่าที่เกิด missing ไป โดยทำการเติมเฉพาะค่าที่เป็น Numeric data เท่านั้น
- b. จากข้อ 2 ทำการ save ไฟล์เป็นรูปแบบ excel ทำเป็น 2 sheets โดยที่ sheet ที่หนึ่ง เก็บรูปตารางแบบยังไม่เติมค่า mean และ sheet ที่สองเก็บตารางภายหลังการเติมค่า mean แล้ว



# Afternoon

---

Day-01



Boyd BigData RPG - Sorratat Sirirattanajakarin

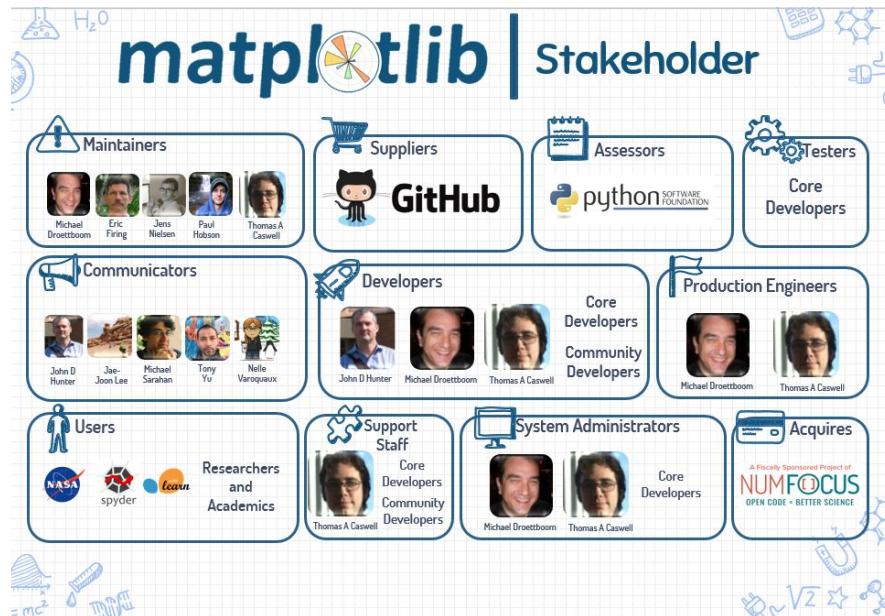
1. Input Data
2. Numpy
3. Pandas
- 4. Visualization**
5. Linear Regression Modeling
6. Project First Day



# The Creator of Matplotlib

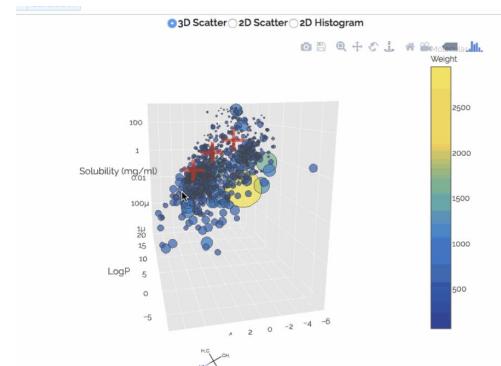
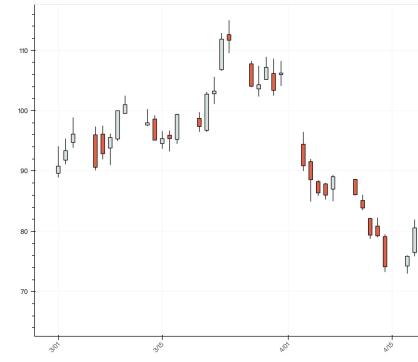
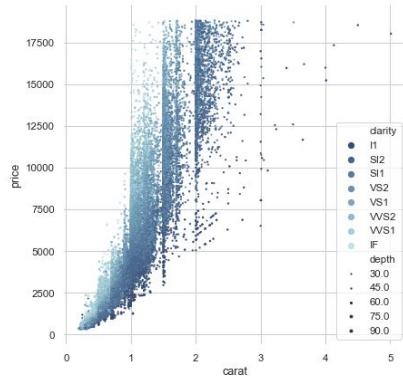
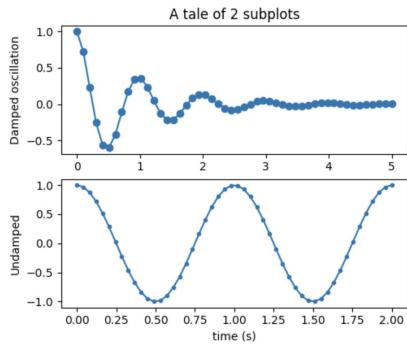


## John D. Hunter





# Visualization on Python



matplotlib

seaborn

bokeh

plotly



# Start with Matplotlib

---

เริ่มต้นด้วยความานี้ทุกครั้ง

```
import matplotlib.pyplot as plt
```

```
%matplotlib inline
```

# Matplotlib-01

Start with Pattern

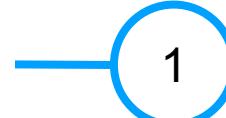




# PLT Structure

---

**plt.plot(x,y)**



รับค่า array แกน x, y

**plt.xlabel('this is x!')**



คำอธิบายแกน x

**plt.ylabel('this is y!')**



คำอธิบายแกน y

**plt.title('My First Plot')**



คำอธิบายกราฟ

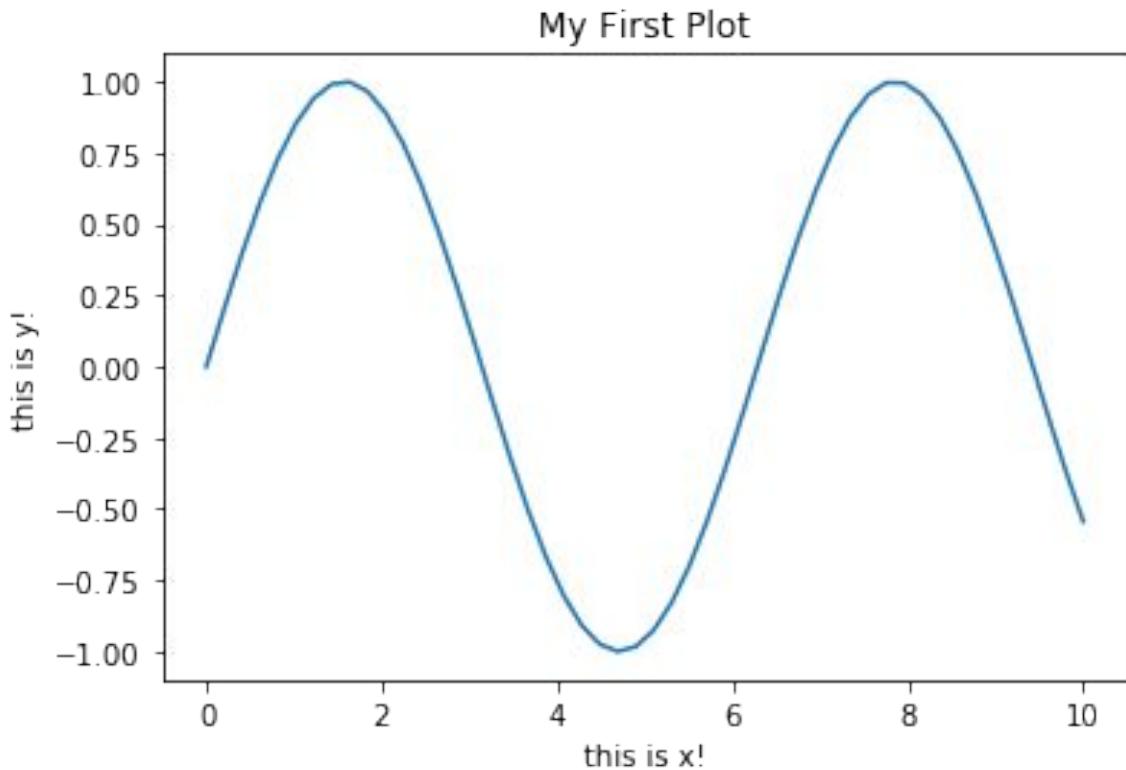
ถ้าเราใส่ค่า x, y ตามนี้

```
y = np.sin(np.linspace(0, 10, 50))
```

1. x มีค่าตั้งแต่ 0-10 จำนวน 50 ตัว จากคำสั่ง np.linspace()
2. นำ x ที่ได้มาเข้าฟังก์ชัน np.sin() และเก็บเป็น y



# แล้วนำไปสร้างกราฟดูตามโครงสร้าง



# Matplotlib-02

Customization





# PLT Structure Customization

```
plt.plot(x,y, 'go--', linewidth=2, markersize=5)
```

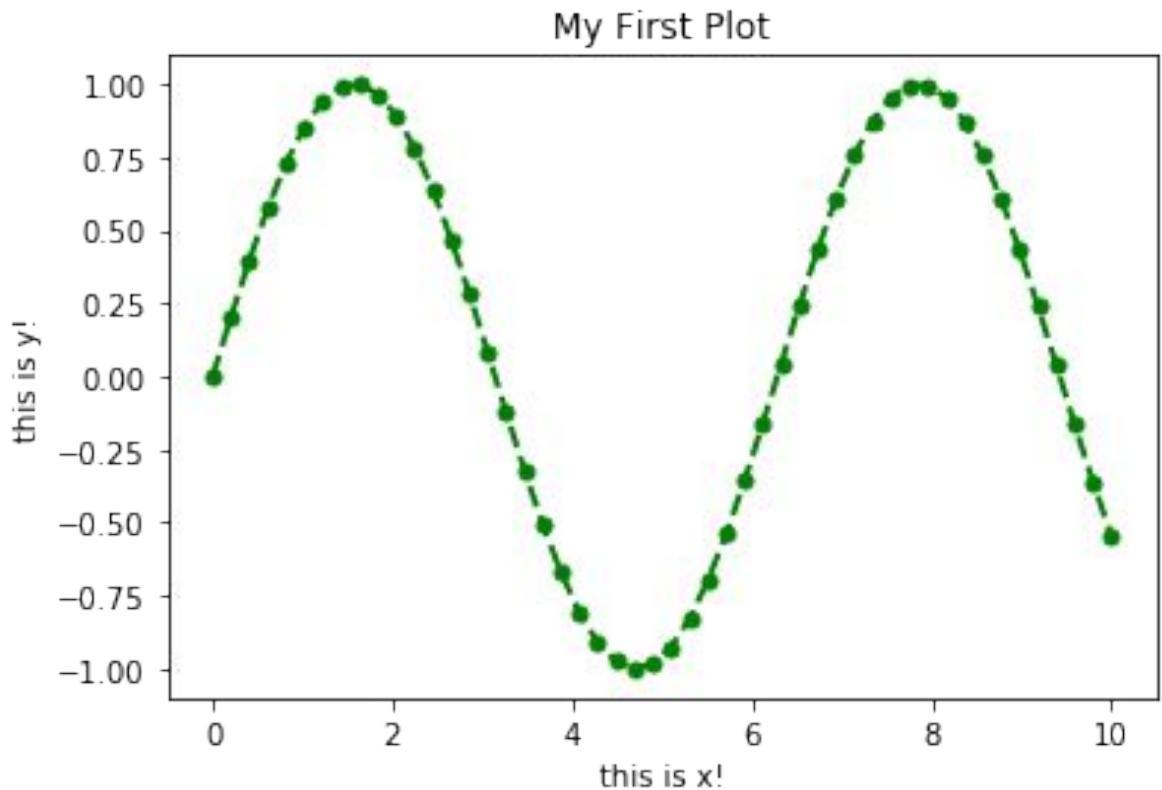
g คือสีเขียว และ o คือ  
mark เป็นเครื่องหมาย  
ของแต่ละจุด

-- คือสัญลักษณ์ของเส้นที่  
ใช้ในการ plot

ขนาดของเส้น

ขนาดของตัว mark

# แล้วนำไปสร้างกราฟดูตามโครงสร้าง



# Matplotlib-03

More Customization



# PLT Structure Customization

```
fmt = '[color][marker][line]'
```

```
plt.plot(x,y, fmt, linewidth=2, markersize=5)
```



# Color

| character | color   |
|-----------|---------|
| 'b'       | blue    |
| 'g'       | green   |
| 'r'       | red     |
| 'c'       | cyan    |
| 'm'       | magenta |
| 'y'       | yellow  |
| 'k'       | black   |
| 'w'       | white   |



# Marker

| character | character | character | character  |
|-----------|-----------|-----------|------------|
| ” ”       | “2”       | “H”       | TICKRIGHT  |
| ” ,       | “3”       | “+”       | TICKUP     |
| “o”       | “4”       | “x”       | TICKDOWN   |
| “v”       | “8”       | “D”       | CARETLEFT  |
| “^”       | “s”       | “d”       | CARETRIGHT |
| “<”       | “p”       | “ ”       | CARETUP    |
| “>”       | “*”       | “_”       | CARETDOWN  |
| “1”       | “h”       | TICKLEFT  | “None”     |



# Line

| character | color               |
|-----------|---------------------|
| '—'       | solid line style    |
| '--'      | dashed line style   |
| '-.'      | dash-dot line style |
| '.'       | dotted line style   |
| '—'       | solid line style    |
| '--'      | dashed line style   |
| '-.'      | dash-dot line style |
| '.'       | dotted line style   |



เอาล่ะ !! ลองปรับกราฟ  
แล้วมาใช้วัดกันหน่อยนะ

พร้อมแล้วจัดเลย !!



# Matplotlib-04

Multiplot



แล้วถ้า 1 ตาราง  
เราอยากเห็นหลายๆ ตารางล่ะ !?

เอ๊ จะเริ่มอย่างไรดีนะ !???



# PLT Multiple line in graph

กราฟที่ 1

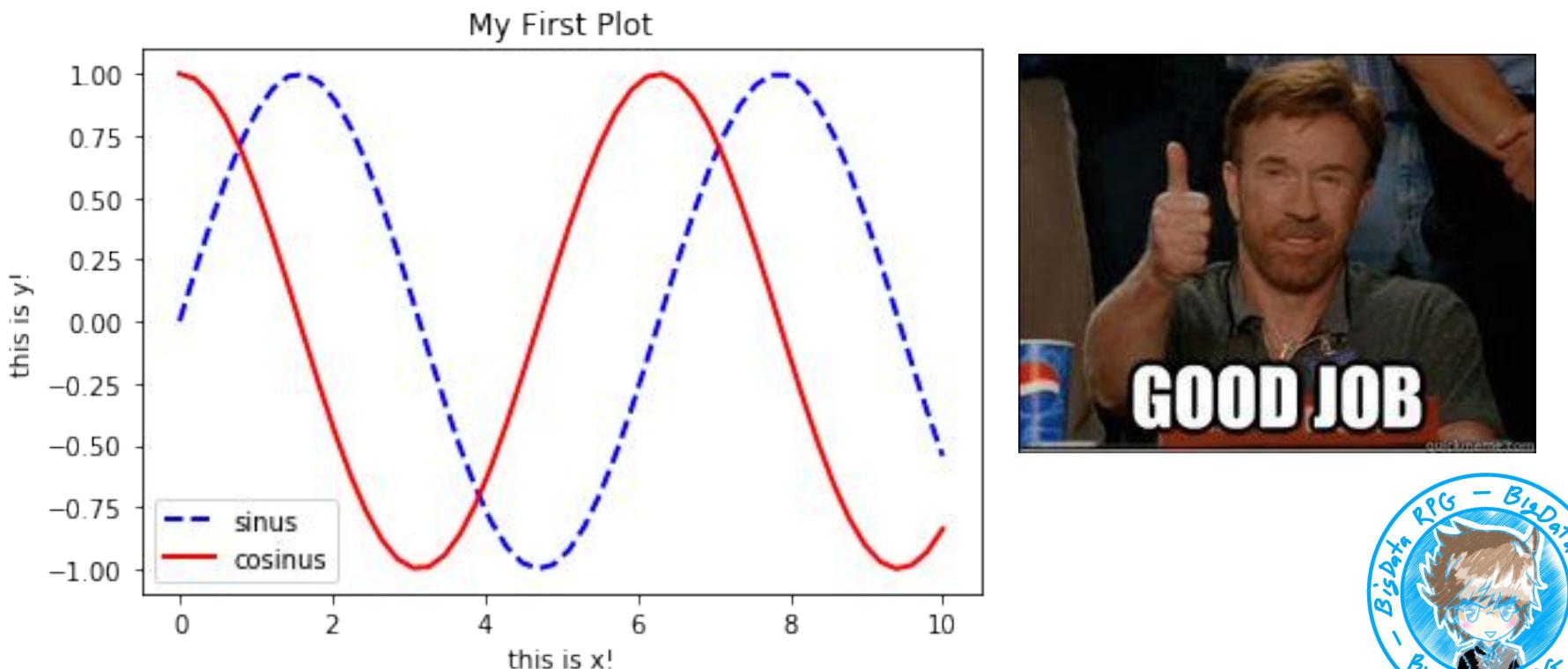
```
plt.plot(x, sinus, label='sinus', color='blue',  
         linestyle='--', linewidth=2)
```

กราฟที่ 2

```
plt.plot(x, cosinus, label='cosinus',  
         color='red', linestyle='-', linewidth=2)
```



# ไอเครเราได้กราฟแล้วนะ =]



# Matplotlib-05

Complexed Matplotlib



# เล่นใหญ่ขึ้นจากการอ่านไฟล์ แล้วเก็บใน df โดย pandas

## จากนั้นจะนำมา plot กราฟกัน



|   | salary | experience | education | management |
|---|--------|------------|-----------|------------|
| 0 | 13876  | 1          | Bachelor  | Y          |
| 1 | 11608  | 1          | Ph.D      | N          |
| 2 | 18701  | 1          | Ph.D      | Y          |
| 3 | 11283  | 1          | Master    | N          |
| 4 | 11767  | 1          | Ph.D      | N          |

# นี่คือ df.head() คร่าว



# เราจะมา plot ดูความสัมพันธ์ ของ year vs salary

โดยใช้สีแบ่งชื่อ民族 กองถึง  
กลุ่มของคนที่จบการศึกษา  
ว่าแตกต่างกันไหม !?



# PLT Multiple line in graph

```
colors =
```

```
{'Bachelor':'r', 'Master':'g', 'Ph.D':'blue'}
```

```
plt.scatter(df['experience'], df['salary'],
```

```
c=df['education'].apply(lambda x: colors[x]),
```

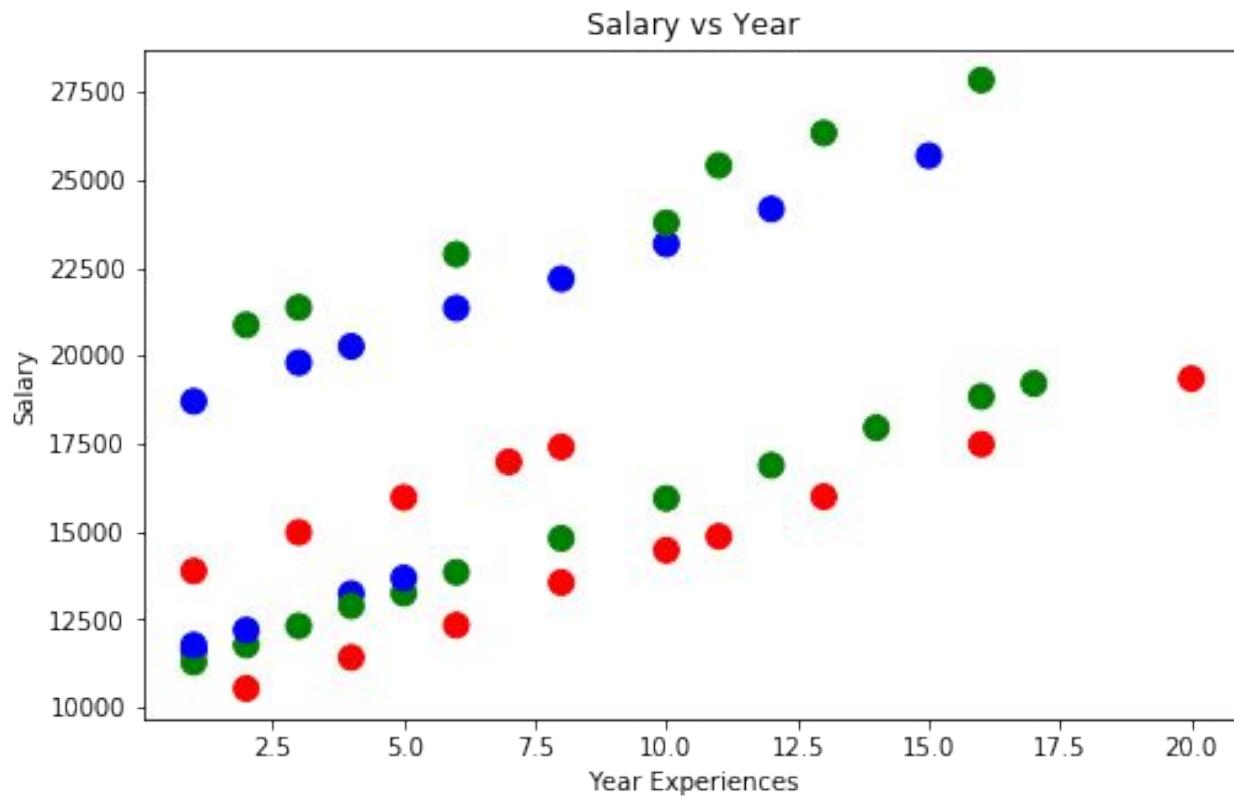
```
s=100)
```



s ຢ່າມາກຈາກ size ນີ້ !



# เรียนร้อย เสร็จแล้วอิว



# Matplotlib-06

Advanced Matplotlib



เราจะเพิ่มความแตกต่างขึ้น  
ระหว่างระดับ Manager

กับกลุ่มที่ไม่ใช่ Manager



# เริ่มจากสร้าง Dict 2 ตัว เพื่อทำสร้างกราฟให้ต่างกัน

```
symbols_manag = dict(Y='*', N='.')
```

```
colors_edu = {'Bachelor':'r', 'Master':'g',  
'Ph.D':'blue'}
```

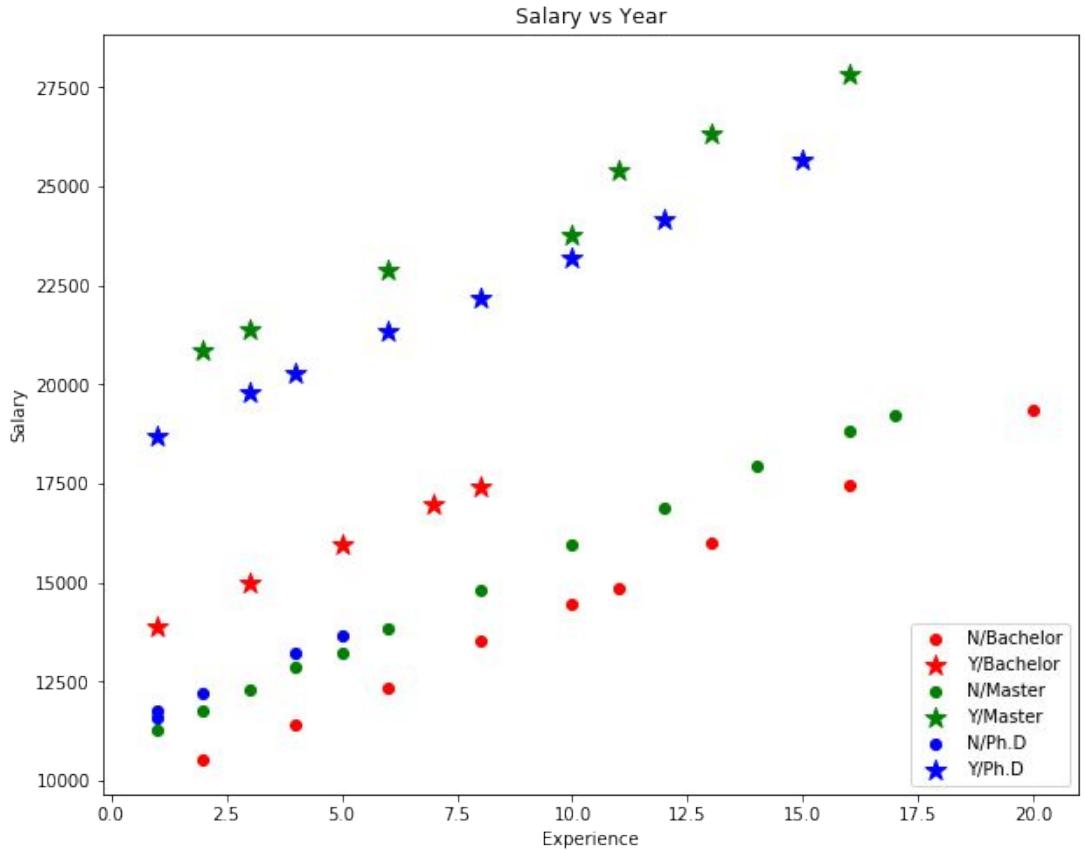


# นำค่า dict มาใส่ในส่วนของ marker และ color ตามลำดับ

```
plt.scatter(d['experience'], d['salary'],  
marker=symbols_manag[manager],  
color=colors_edu[edu], s=200,  
label=manager+"/"+edu)
```



# เรารีบมองเห็น Pattern มากขึ้นจาก Visualization





# Save Graph

หากเราวาดกราฟได้แล้ว ก็อย่าลืมเซฟเก็บไว้ด้วยล่ะ

```
plt.savefig("output-data/sinus.png")
```

bitmap

```
plt.savefig("output-data/sinus.svg")
```

vector

```
plt.savefig("output-data/sinus.pdf")
```

pdf

# Afternoon

---

Day-01



Boyd BigData RPG - Sorratat Sirirattanajakarin

1. Input Data
2. Numpy
3. Pandas
4. Visualization
- 5. Linear Regression Modeling**
6. Project First Day

# Afternoon

---

Day-01



Boyd BigData RPG - Sorratat Sirirattanajakarin

1. Input Data
2. Numpy
3. Pandas
4. Visualization
5. Linear Regression Modeling
6. **Project First Day**



# Final Project

## PROJECT: EXPLORING THE BITCOIN CRYPTOCURRENCY MARKET



# Package ที่ใช้ใน Project นี้

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

ครั้งต่อๆไปไม่ต้องใช้ plt.show() เพื่อแสดงกราฟ  
และ svg ทำการแสดงภาพแบบ vector

```
%matplotlib inline
```

```
%config InlineBackend.figure_format = 'svg'
```

ทำการเรียกใช้ Style กราฟชื่อ fivethirtyeight

```
plt.style.use('fivethirtyeight')
```



# อยากเปลี่ยน Style Graph ล่ะสิ !!

**print(plt.style.available)**

ทำการ Check ว่ามี Style ไหนให้ใช้งาน

```
['seaborn-dark', 'seaborn-paper', 'grayscale', 'bmh', 'classic',
'seaborn-deep', 'seaborn-muted', 'seaborn-colorblind',
'tableau-colorblind10', 'seaborn-whitegrid', 'seaborn-poster',
'seaborn-white', 'Solarize_Light2', 'dark_background', 'seaborn-bright',
'ggplot', 'seaborn', 'fivethirtyeight', 'seaborn-talk', 'seaborn-darkgrid', 'fast',
'seaborn-dark-palette', '_classic_test', 'seaborn-ticks', 'seaborn-pastel',
'seaborn-notebook']
```





# Start ..... !

```
[  
 {  
   "id": "bitcoin",  
   "name": "Bitcoin",  
   "symbol": "BTC",  
   "rank": "1",  
   "price_usd": "6465.16414237",  
   "price_btc": "1.0",  
   "24h_volume_usd": "4105107395.16",  
   "market_cap_usd": "111637137781",  
   "available_supply": "17267487.0",  
   "total_supply": "17267487.0",  
   "max_supply": "21000000.0",  
   "percent_change_1h": "-0.23",  
   "percent_change_24h": "-0.56",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "ethereum",  
   "name": "Ethereum",  
   "symbol": "ETH",  
   "rank": "2",  
   "price_usd": "350.00000000",  
   "price_btc": "0.05585424",  
   "24h_volume_usd": "1000000000.0",  
   "market_cap_usd": "35000000000.0",  
   "available_supply": "100000000.0",  
   "total_supply": "100000000.0",  
   "max_supply": "100000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "ripple",  
   "name": "Ripple",  
   "symbol": "XRP",  
   "rank": "3",  
   "price_usd": "0.25000000",  
   "price_btc": "0.00425000",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "400000000.0",  
   "total_supply": "400000000.0",  
   "max_supply": "400000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "eos",  
   "name": "EOS",  
   "symbol": "EOS",  
   "rank": "4",  
   "price_usd": "0.35000000",  
   "price_btc": "0.00625000",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "285714285.7142857",  
   "total_supply": "285714285.7142857",  
   "max_supply": "285714285.7142857",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "cardano",  
   "name": "Cardano",  
   "symbol": "ADA",  
   "rank": "5",  
   "price_usd": "0.10000000",  
   "price_btc": "0.00187500",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "10000000000.0",  
   "total_supply": "10000000000.0",  
   "max_supply": "10000000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "dash",  
   "name": "Dash",  
   "symbol": "DASH",  
   "rank": "6",  
   "price_usd": "100.00000000",  
   "price_btc": "0.01875000",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "10000000.0",  
   "total_supply": "10000000.0",  
   "max_supply": "10000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "monero",  
   "name": "Monero",  
   "symbol": "XMR",  
   "rank": "7",  
   "price_usd": "100.00000000",  
   "price_btc": "0.01875000",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "10000000.0",  
   "total_supply": "10000000.0",  
   "max_supply": "10000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "stellar",  
   "name": "Stellar",  
   "symbol": "XLM",  
   "rank": "8",  
   "price_usd": "0.05000000",  
   "price_btc": "0.00093750",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "20000000000.0",  
   "total_supply": "20000000000.0",  
   "max_supply": "20000000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "iotex",  
   "name": "IoTeX",  
   "symbol": "IOTX",  
   "rank": "9",  
   "price_usd": "0.05000000",  
   "price_btc": "0.00093750",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "20000000000.0",  
   "total_supply": "20000000000.0",  
   "max_supply": "20000000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 },  
 {  
   "id": "digibyte",  
   "name": "Digibyte",  
   "symbol": "DGB",  
   "rank": "10",  
   "price_usd": "0.05000000",  
   "price_btc": "0.00093750",  
   "24h_volume_usd": "100000000.0",  
   "market_cap_usd": "1000000000.0",  
   "available_supply": "20000000000.0",  
   "total_supply": "20000000000.0",  
   "max_supply": "20000000000.0",  
   "percent_change_1h": "-0.01",  
   "percent_change_24h": "-0.01",  
   "percent_change_7d": "0.01",  
   "last_updated": "1536936441"  
 }]
```

เริ่มจากเข้าไปดู API ของ Cryptocurrency กัน

<https://api.coinmarketcap.com/v1/ticker/>

# ทำการดูดมาแล้วแปลงเป็น dataframe

```
pd.read_json("https://api.coinmarketcap.com/v1/ticker/")
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 15 columns):
24h_volume_usd      100 non-null float64
available_supply     100 non-null int64
id                  100 non-null object
last_updated         100 non-null int64
market_cap_usd       100 non-null int64
max_supply           32 non-null float64
name                100 non-null object
percent_change_1h    100 non-null float64
percent_change_24h   100 non-null float64
percent_change_7d    100 non-null float64
price_btc            100 non-null float64
price_usd            100 non-null float64
rank                100 non-null int64
symbol               100 non-null object
total_supply          100 non-null int64
dtypes: float64(7), int64(5), object(3)
memory usage: 11.8+ KB
```



# ข้อมูลน้อยไป ไม่เป็นไฟร์เซ็ต csv

```
pd.read_csv("datasets/coinmarketcap_06122017.csv")
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1326 entries, 0 to 1325
Data columns (total 16 columns):
Unnamed: 0           1326 non-null int64
24h_volume_usd      1270 non-null float64
available_supply     1031 non-null float64
id                  1326 non-null object
last_updated         1326 non-null int64
market_cap_usd      1031 non-null float64
max_supply          215 non-null float64
name                1326 non-null object
percent_change_1h    1273 non-null float64
percent_change_24h   1270 non-null float64
percent_change_7d    1283 non-null float64
price_btc            1326 non-null float64
price_usd            1326 non-null float64
rank                1326 non-null int64
symbol               1326 non-null object
total_supply         1211 non-null float64
dtypes: float64(10), int64(3), object(3)
memory usage: 165.8+ KB
```



# เนื่องจาก Market Cap บางตัวไม่มี ! Filter ออกซะ

```
cap = df.query('market_cap_usd > 0')
```

|               | <b>id</b> | <b>market_cap_usd</b> |
|---------------|-----------|-----------------------|
| <b>count</b>  | 1031      | 1.031000e+03          |
| <b>unique</b> | 1031      | NaN                   |
| <b>top</b>    | plncoin   | NaN                   |
| <b>freq</b>   | 1         | NaN                   |
| <b>mean</b>   | NaN       | 3.630503e+08          |
| <b>std</b>    | NaN       | 6.844947e+09          |
| <b>min</b>    | NaN       | 1.000000e+01          |
| <b>25%</b>    | NaN       | 1.880625e+05          |
| <b>50%</b>    | NaN       | 1.488564e+06          |
| <b>75%</b>    | NaN       | 1.546756e+07          |
| <b>max</b>    | NaN       | 2.130493e+11          |

```
cap.describe(include='all')
```



# Set index ใหม่ด้วย columns id แทน

|              | market_cap_usd | market_cap_perc |
|--------------|----------------|-----------------|
| id           |                |                 |
| bitcoin      | 2.130493e+11   | 56.918669       |
| ethereum     | 4.352945e+10   | 11.629410       |
| bitcoin-cash | 2.529585e+10   | 6.758088        |
| iota         | 1.475225e+10   | 3.941238        |
| ripple       | 9.365343e+09   | 2.502063        |
| dash         | 5.794076e+09   | 1.547956        |
| litecoin     | 5.634498e+09   | 1.505323        |
| bitcoin-gold | 4.920065e+09   | 1.314454        |
| monero       | 4.331688e+09   | 1.157262        |
| cardano      | 3.231420e+09   | 0.863312        |

**cap10 =**

**cap[:10].set\_index('id')**

**cap10 = cap10.assign(**

**market\_cap\_perc = lambda**

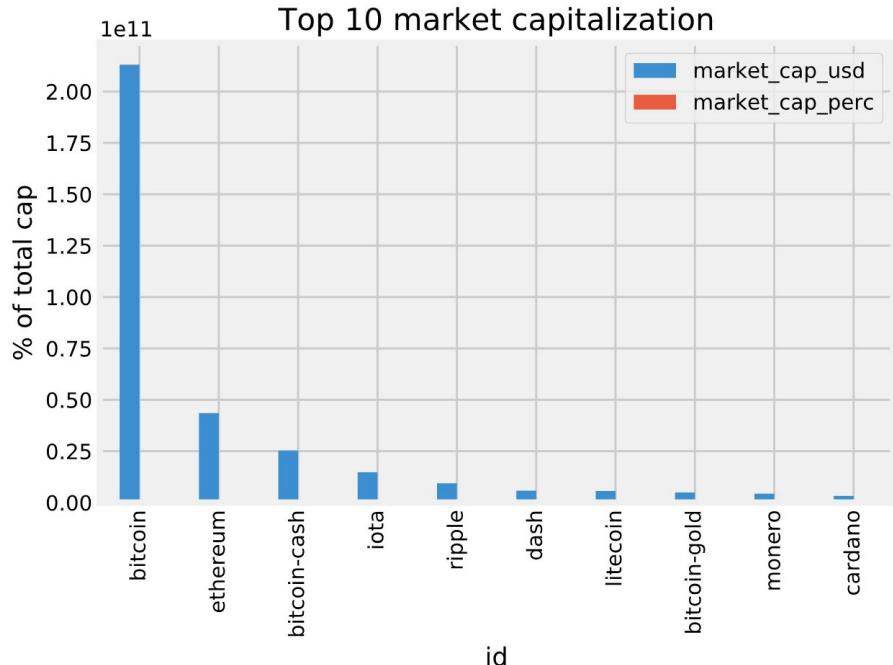
**x: (x.market\_cap\_usd /**

**cap.market\_cap\_usd.sum())**

**\* 100)**



# สร้างกราฟใหม่เพื่อดู Market Cap



`ax =`

```
cap10.plot.bar(title=TOP_CA  
P_TITLE)
```

```
ax.set_ylabel(TOP_CAP_YL  
ABEL)
```

Marketcap % มองไม่เห็นเลย !!??



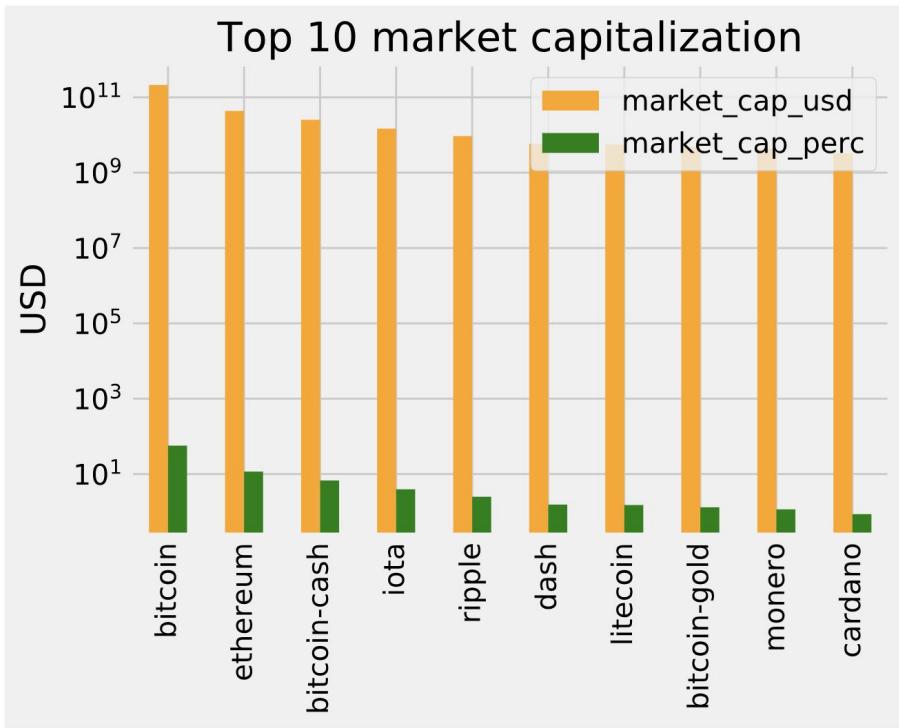
เพื่อความชัดเจนขึ้น ยิ่งขึ้น

เราลองเปรียบเทียบกับ  
Marketcap % vs Marketcap USD

อีกครั้ง คราวนี้ปรับ Scale แกน Y ใหม่



# ปรับ Scale ใน Market Cap



```
ax.set_ylabel('USD')
```

```
ax.set_yscale('log')
```



# ถ้าอยาก Trade ได้เงินเยอะๆ เข้า-ออกบ่อยๆ ก็ต้องดูที่ **Volatility in cryptocurrencies**



# เนื่องจาก Market Cap บางตัวไม่มี ! Filter ออกซะ

```
volatility = dec6[['id', 'percent_change_24h',  
'percent_change_7d']]
```

```
volatility = volatility.set_index('id').dropna()
```

```
volatility =  
volatility.sort_values(by=['percent_change_24h'])
```



# ໂອັນໂຫວວາ !! ວັນເດືອນເກືອນ 100 %

|                      | percent_change_24h | percent_change_7d |
|----------------------|--------------------|-------------------|
| <b>id</b>            |                    |                   |
| <b>flappycoin</b>    | -95.85             | -96.61            |
| <b>credence-coin</b> | -94.22             | -95.31            |
| <b>coupecoin</b>     | -93.93             | -61.24            |
| <b>tyrocoin</b>      | -79.02             | -87.43            |
| <b>petrodollar</b>   | -76.55             | 542.96            |



# โฉมหน้าเจ้า Flappycoin



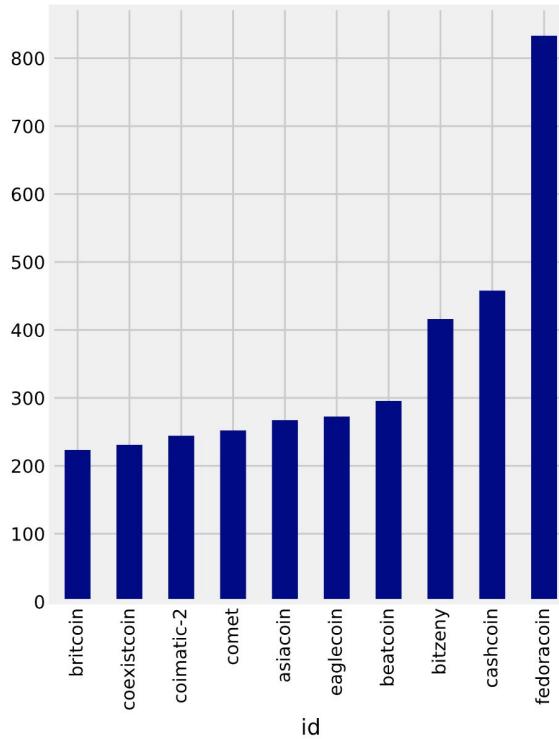
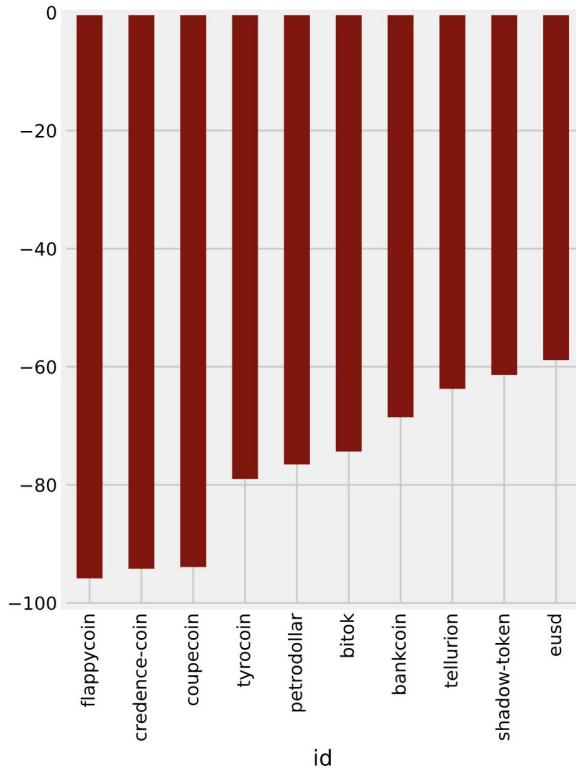
มาดู 10 อันดับตัวที่

นำเสนอสุด กับ ดีสุด

10 อันดับแรกกันดีกว่า



# 10 Loser และ 10 Winner



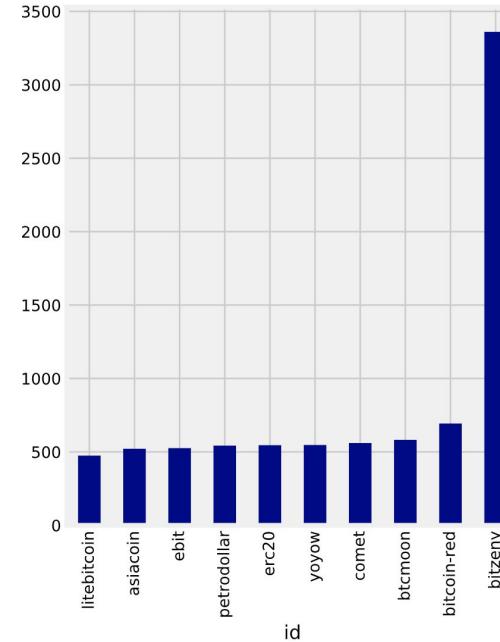
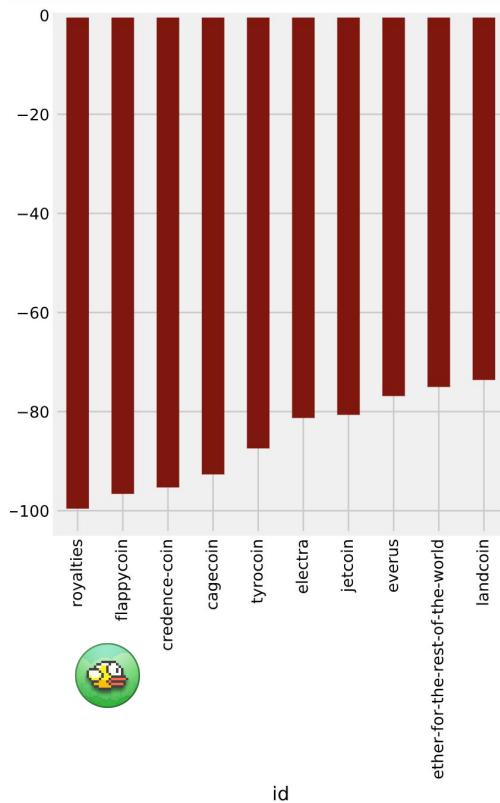
จริงๆ แล้วเราลองดู

**Volatility**

ในรอบ Week กันบ้างดีกว่า



# 10 Loser และ 10 Winner



# จริงๆแล้วในการเล่น Cryptocurrencies เรายังต้องดูปัจจัยอื่นๆอีกมากมาย ควบคู่กับ สายแคลงซ์ข้อมูล







*It's done.*

# Python Data Science for Developer

Thank you

