Probability and Statistic Full Report

เสนอ

ผศ.ดร. สุรินทร์ กิตติธรกุล

จัดทำโดย

นาย ภัทรพัทธิ์ ชัยอมรเวทย์ รหัสนักศึกษา : 62010684

นาย สหทัศน์ ลีวัฒนา รหัสนักศึกษา : 62010922

วิชา Probability and Statistic

รหัสวิชา : 01076253

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

King Mongkut’s Institute of Technology Ladkrabang

สารบัญ

Nutritional values for common foods (HW-1)

ชื่อคอลัมน์ **3** คอลัมน์ที่สนใจ

- Name of foods
- Calories
- Total_fat

Why is it interesting ?

- ในทุกๆวันนี้ผู้คนได้บริโภคอาหารหลากหลายชนิดต่างๆนานา อาหารเหล่านั้นถูกแบ่งออกเป็น 5 ประเภทหลักๆ ได้แก่ คาร์โบไฮเดรต วิตามิน เกลือแร่ โปรตีน และไขมัน ซึ่งอาหารเหล่านั้นมีทั้งข้อดีและข้อเสียตามมาด้วย ผู้คนส่วนใหญ่มักจะมองในหลายๆมุม ประเด็นที่ผมยกมานั้นจะโฟกัสไปที่การให้พลังงานของอาหารอย่างเหมาะสม เพื่อให้ผู้บริโภคได้รับรู้ว่าอาหารที่บริโภคไปนั้นสามารถควบคุมพลังงานให้กับตนเองได้อย่างไรบ้าง ตัวอย่าง เช่น การบริโภคอาหารจำพวกฟาสต์ฟู้ด (fast food) จะทำให้ร่างกายไม่แข็งแรงและอ้วนได้ง่าย การบริโภคอาหารจำพวกสลัดจะทำให้ร่างกายแข็งแรงและมีผิวเปล่งใส ซึ่งคำเหล่านี้ ก็เป็นเรื่องจริงอยู่ส่วนหนึ่ง แต่เราสามารถนำอาหารเหล่านี้มาประยุกต์ใช้และบริโภคได้ถูกต้องและเหมาะสม

แหล่งที่มาของข้อมูล

https://www.kaggle.com/trolukovich/nutritional-values-for-common-foods-and-products?select=nutrition.csv

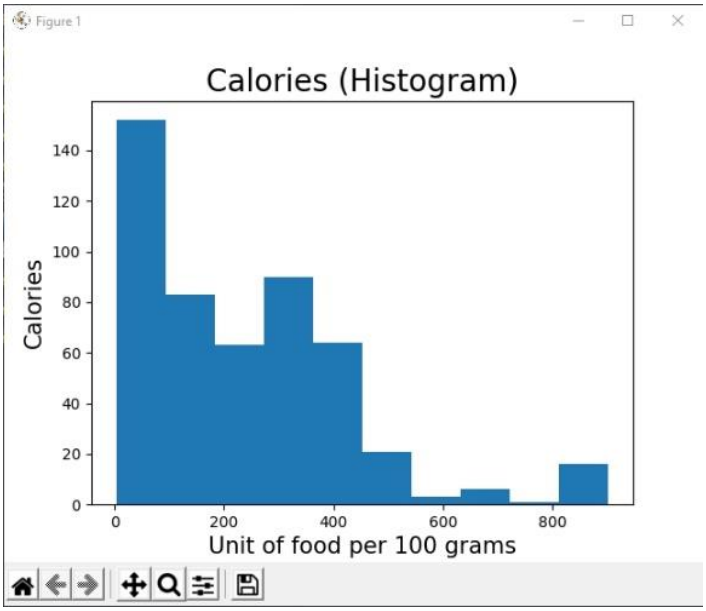
Nutritional values for common foods (HW-2)

ภาษาที่ใช้ **: Python**

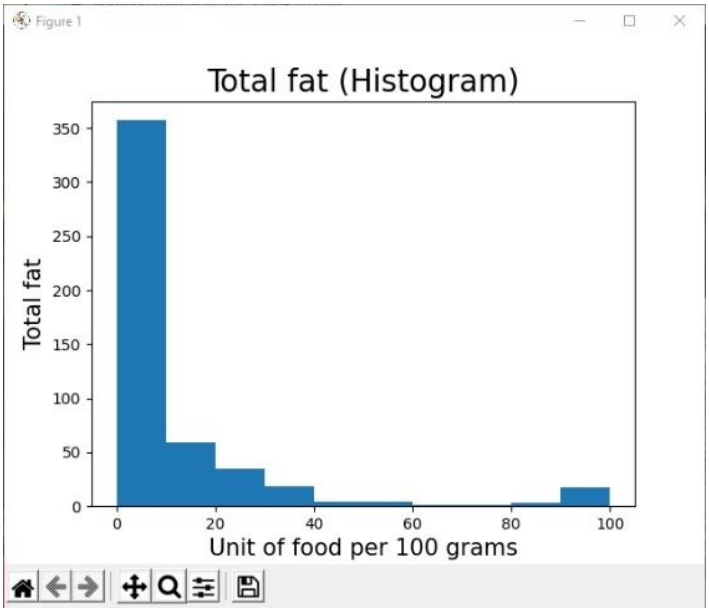
สิ่งที่นำมาเปรียบเทียบ : ปริมาณพลังงาน (kcal) ในอาหารชนิดนั้น 100 g และ ปริมาณไขมัน (g) ในอาหารชนิดนั้น 100 g

Picture 1

Scattered graph's outliers อยู่บริเวณตั้งแต่ calories > 580 kcal และ total fat > 40 units

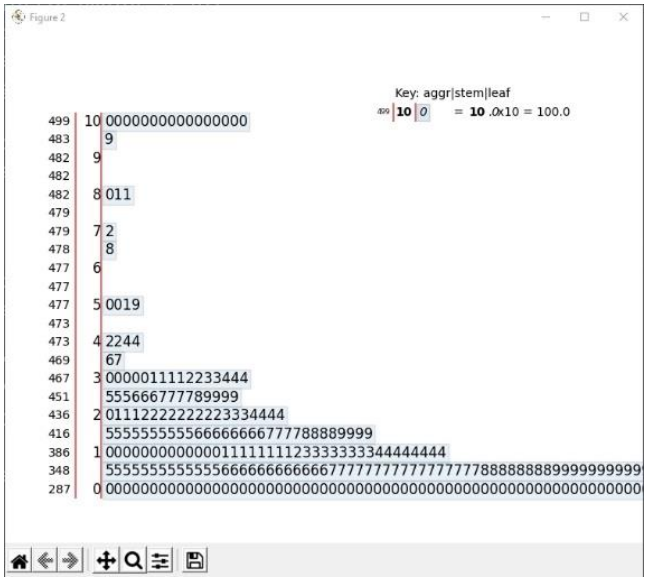


Picture 2

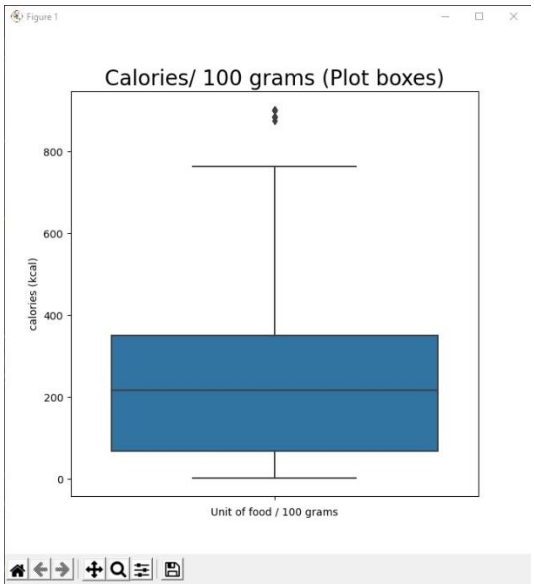


Picture 3

Picture 4 Stem and leaf (Calories)

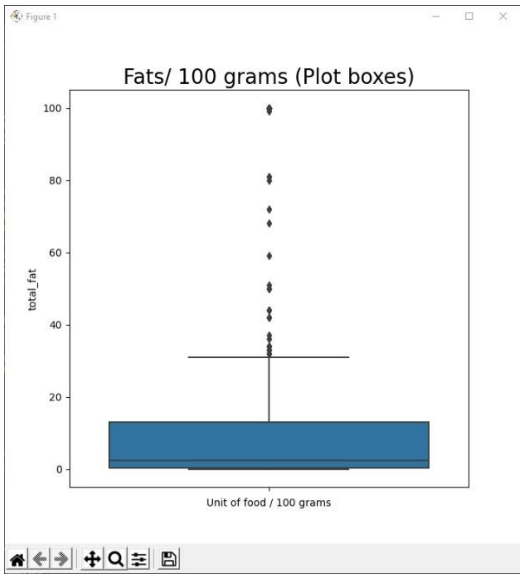


Picture 5 Stem and leaf (Total fat)



Picture 6

Calories plot box's outliers จะอยู่บริเวณอาหารที่มีพลังงานมากกว่า 800 kcal ขึ้นไป



Picture 7

Fats plot box's outliers จะอยู่บริเวณอาหารที่มีค่าไขมันมากกว่า 30 units ขึ้นไป

สถิติพื้นฐาน :

ในที่นี้ ผมได้ทำการ import library ของภาษา python ที่ชื่อว่า lib ‘statistic’ และ lib ‘ pandas ‘ ซึ่งเป็น library ที่สามารถช่วยคำนวณถึงสถิติพื้นฐานได้ โดยมคำสั่งดังนี้

Data.mean() -> หาค่าเฉลี่ยของข้อมูลนั้นๆ

Data.median() -> หามัธยฐานของข้อมูลนั้นๆ

Data.std() -> หาส่วนเบี่ยงแบนมาตรฐานของข้อมูลนั้นๆ

Data.mode() -> หาฐานนิยมของข้มูลนั้นๆ

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

df = pd.read_csv('nutrition.csv')
df = df[['calories']]

y = 'calories'

Calories_mean = df['calories'].mean()
Calories_median = df['calories'].median()
Calories_Std = df['calories'].std()
Calories_mode = df['calories'].mode()

print('Calories mean is : ' + str(Calories_mean))
print('Calories meadian is : ' + str(Calories_median))
print('Calories standard deviation of salaries : ' + str(Calories_Std))
print('Calories mode is : ' + str(Calories_mode))
```

Calories Statistic Code

ผลลัพธ์ของ Code python คือ

Calories mean is : 235.3787

Calories median is : 216.0

Calories S.D. is 196.3723

Calories mode is : 0 and 884

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

df = pd.read_csv('nutrition.csv')
df = df[['total_fat']]

y = 'total_fat'

Fat_mean = df['total_fat'].mean()
Fat_median = df['total_fat'].median()
Fat_Std = df['total_fat'].std()
Fat_mode = df['total_fat'].mode()

print('Fat mean is : ' + str(Fat_mean))
print('Fat meadian is : ' + str(Fat_median))
print('Fat standard deviation of salaries : ' + str(Fat_Std))
print('Fat mode is : ' + str(Fat_mode))
```

Fat Statistic Code

<u>ผลลัพธ์ของ Code python คือ</u>

Fat mean is : 11.0801

Fat median is : 2.4

Fat S.D. is 20.6862

Fat mode is : 0 and 0.2

บทวิเคราห์และสรุปผล

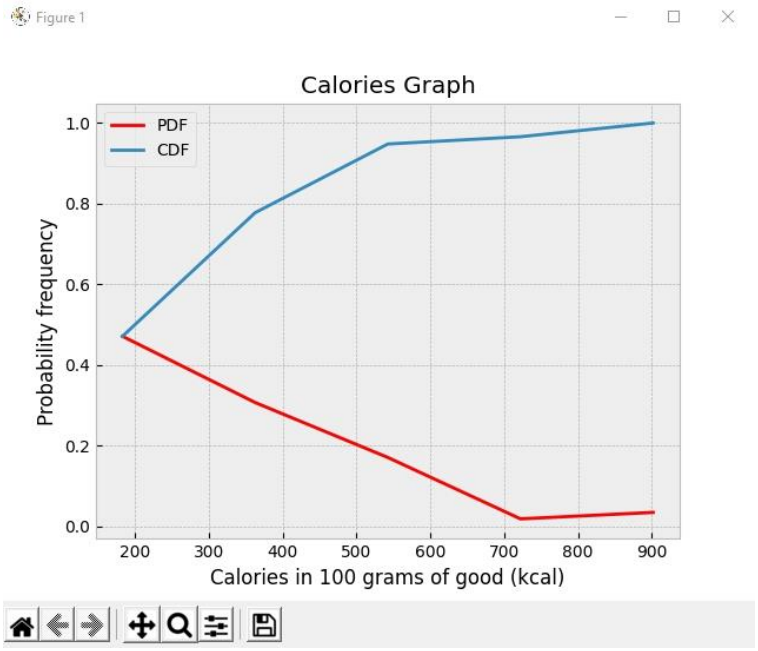
จากข้อมูลในกราฟที่นำมาเปรียบเทียบนั้น จะสังเกตได้ว่าข้อมูลของ Total fat และ Calories ในอาหารชนิดนั้นๆ ไม่ได้มีการแปรผันตรงและแปรผกผันซึ่งกันและกัน แต่จะแสดงว่าอาหารชนิดนี้มีปริมาณไขมันที่มากหรือน้อย รวมถึงพลังงานที่อาหารชนิดนี้ให้มาด้วย ตัวอย่างเช่น เบคอนให้พลังงาน 200 kcal / 100 grams และ มีไขมัน 60 unit / 100 grams ถั่วบดให้พลังงาน 220 kcal / 100 grams และ มีไขมัน 4 unit / 100 grams จะสังเกตุได้ว่าอาหารแต่ละชนิดให้พลังงานและมีไขมันที่ต่างกัน เนื่องจากอาหารแต่ละชนิดมีสารอาหารที่แตกต่างกันออกไปในอาหารชนิดนั้น ไขมันในอาหารแต่ละชนิดก็จะให้พลังงานที่แตกต่างกันออกไปด้วย เพราะไขมันในแต่ละเมนูนั้นมีส่วนผสมของไขมันที่แตกต่างกันออกไป และ มีการให้พลังงานที่แตกต่างกันออกไปด้วย

Nutritional values for common foods (HW-3)

ภาษาที่ใช้ : Python

Column ที่เลือกใช้ : Calories / Total_fat

1.Caloires with Probability Density Function / Cumulative Prob Function

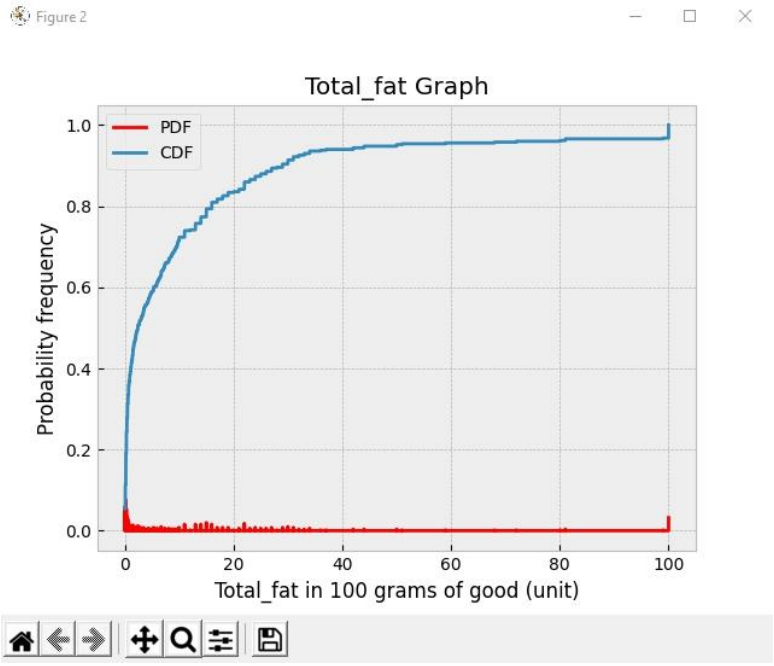


<u>ในแกน x ของกราฟ</u> -> จะบอกถึงจำนวน calories ในอาหารทุกๆ 100 กรัม มีหน่วยเป็น kcal หน่วยของพลังงานในอาหาร 500 ชนิด มีตั้งแต่ 0 - 900 kcal ในแกน x จึงมีเลขตั้งแต่ 0 - 900

<u>ในแกน y ของกราฟ</u> -> จะบอกถึงโอกาสที่จะมี Calories จำนวน … kcal ในอาหารทุกๆ 100 กรัม ไม่มีหน่วย (prob) จะคิดจาก จำนวนอาหารที่มี Calories เท่านั้น แล้วหารด้วยจำนวน Calories เช่น 0.48 = prob(อาหารที่มี 200 kcal ต่อ 100 g) ดังนั้น อาหารที่มี 200 kcal ต่อ 100 g จะมีค่าเท่ากับ 0.48 * 500 = 240 อย่าง เมื่อเราได้ความน่าจะเป็นนี้มาแล้ว เราสามารถหาจำนวนของอาหารให้ได้

หมายเหตุ : ในการวิเคราะห์ครั้งนี้ nutrition.scv มีข้อมูลอยู่ 500 rows หมายความว่า มีอาหารอยู่ 500 อย่าง

2.Total_fat with Probability Density Function / Cumulative Prob Function



<u>ในแกน x ของกราฟ</u> -> จะบอกถึงจำนวนไขมัน (total_fat) ในอาหารทุกๆ 100 กรัม ไม่มีหน่วย หน่วยไขมันของในอาหาร 500 ชนิด มีตั้งแต่ 0-100 หน่วย ในแกน x จึงมีเลขตั้งแต่ 0 - 100

<u>ในแกน y ของกราฟ</u> -> จะบอกถึงโอกาสที่จะมี Total fat จำนวน… หน่วย ในอาหารทุกๆ 100 กรัม ไม่มีหน่วย (prob) โดยคิดเหมือน calories แต่ข้อมูลจะต่างกันเนื่องจากช่วงของหน่วยมีค่าต่างกัน 0-100 และ 0-900

บทวิเคราะห์

จากกราฟที่ผ่านมา เราได้ทำการวิจัยข้อมูล 2 คอลัมน์ โดยแต่ละคอลัมน์จะมีกราฟอยู่ 2 ประเภท ได้แก่ Probability Density Function (PDF) และ Cumulative Prob Function (CPF) ซึ่งแต่ละกราฟจะบ่งบอกข้อมูลต่างกันไป PDF graph จะบ่งบอกว่าข้อมูลนี้แบ่งได้เป็นส่วนๆเท่าไหร่ และเมื่อข้อมูลขนาดเท่านี้จะมีความน่าจะเป็นเท่าไหร่ ส่วน CDF graph จะบ่งบอกว่าข้อมูลนี้มีการสะสมความน่าจะเป็นมากน้อยแค่ไหน การสะสมความน่าจะเป็นของข้อมูลนั้น จะขึ้นอยู่กับความชันของกราฟ ถ้าความชันมาก ข้อมูลในช่วงนั้น ก็จะมีมาก ถ้าความชันน้อย ข้อมูลในช่วงนั้น ก็จะมีน้อย

จากกราฟข้างต้น อาหารที่นำมาวิจัย 500 อย่าง และแต่ละอาหารจะมีขนาด 100 กรัม ความน่าจะเป็นที่อาหารมีพลังงาน (Calories) ที่มีค่าสูงที่สุดก็คือ 0.48 ซึ่ง ค่าพลังงานก็ประมาณ 170 kcal ทางฝั่งไขมันความน่าจะเป็นสูงที่สุดก็คือ 0.9 กว่าๆ โดยจะมีไขมันอยู่ที่ 90 กว่า หน่วย จะสรุปผลได้ว่า อาหารที่นำมาทำการวิจัย 500 อย่างนั้น เราได้พบความน่าจะเป็นของไขมัน มีค่ามากกว่าความน่าจะเป็นของค่าพลังงาน หมายความว่า เราสามารถค้นหาข้อมูลของคอลัมน์ไขมันได้ง่ายกว่าเมื่อเปรียบเทียบกับค่าพลังงานในอาหาร **Prob max (ไขมัน) 0.9 , Prob max (ค่าพลังงาน)** อย่างไรก็ตาม อาหารประกอบไปด้วยสารอาหารมากมาย ตัวความน่าจะเป็นอาจจะขึ้นอยู่กับสวนผสมของอาหารอย่างอื่นด้วย เช่น ค่าพลังงานอาจจะมาจากไขมัน โปรตีน และ คาร์โบรไฮเดรต

```python
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

plt.style.use('bmh')
pd.options.mode.chained_assignment = None
df = pd.read_csv('nutrition.csv')
df.dropna(inplace=True)

Calories_rate, Calories_rating = np.histogram(df["calories"], bins=5)
totalfat_review, totalfat_review = np.histogram(df["total_fat"], bins=1000000)
pdf_rating = Calories_rate / sum(Calories_rate)
pdf_review = Calories_rate / sum(Calories_rate)
cdf_rating = np.cumsum(pdf_rating)
cdf_review = np.cumsum(pdf_review)
plt.figure()
plt.plot(Calories_rating[1:], pdf_rating, color="red", label="PDF")
plt.plot(Calories_rating[1:], cdf_rating, label="CDF")
plt.title("Calories Graph")
plt.xlabel('Calories in 100 grams of good (kcal)')
plt.ylabel('Probability frequency')
plt.figure()
plt.plot(totalfat_review[1:], pdf_review, color="red", label="PDF")
plt.plot(totalfat_review[1:], cdf_review, label="CDF")
plt.title("Total_fat Graph")
plt.xlabel('Total_fat in 100 grams of good (unit)')
plt.ylabel('Probability frequency')
plt.show()
```

Source Code Python

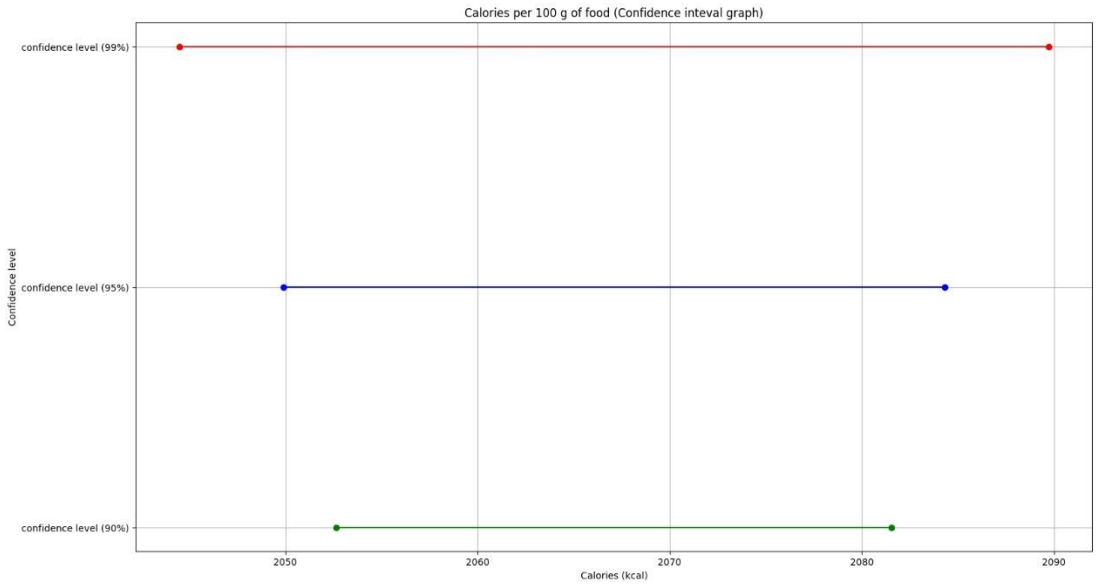
Matplotlib.pyplot -> plot graph และแสดง label ต่างๆ

Pandas -> อ่านข้อมูลและคอลัมน์จากไฟล์ csv

Nutritional values for common foods (HW-4)

ภาษาที่ใช้ : python

Column ที่เลือกใช้ : Calories หน่วย (kcal)



<u>ในแกน x ของกราฟ</u> -> จะบอกถึงจำนวนของ Calories ในช่วงระดับความมั่นใจแต่ละช่วง ซึ่งมีหน่วยเป็น kcal

<u>ในแกน y ของกราฟ</u> -> จะบอกถึงระดับความมั่นใจที่มาใช้ในการคำนวณ ซึ่งแบ่งเป็น 3 ระดับ ได้แก่ 90% , 95% และ 99%

[ที่มาของกราฟ] -> ก่อนที่เราจะได้ Confidence interval graph มานั้น เราต้องรู้ก่อนว่าการทดลองของเรามีองค์ประกอบอะไรบ้าง เช่น จำนวนประชากร, ค่าเออเร่อ, ค่าเฉลี่ย ฯลฯ ซึ่งระดับความเชื่อมั่น จะคำนวณได้จากสูตรข้างล่างดังนี้

$$\bar{X} - Z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}} < \mu < \bar{X} + Z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}}$$

เมื่อ μ เป็นค่าเฉลี่ยของประชากรที่ต้องการประมาณค่า
$\bar{X}$ เป็นค่าเฉลี่ยจากกลุ่มตัวอย่าง
σ เป็นค่าส่วนเบี่ยงเบนมาตรฐานของประชากร (และ σ^2 เป็นค่าความแปรปรวนของประชากร)
s เป็นค่าส่วนเบี่ยงเบนมาตรฐานของกลุ่มตัวอย่าง>
n เป็นขนาดของกลุ่มตัวอย่าง
$Z_{1-\alpha/2}$ เป็นค่าสถิติ Z ที่เปิดได้จากตาราง (บางตารางใช้ค่า $Z_{\alpha/2}$)
$t_{1-\alpha/2}$ เป็นค่าสถิติ t ที่เปิดได้จากตาราง ที่ df=n-1

ในการคำนวณและการหาค่าข้อมูลเชิงสถิตินั้น เราสามารถเขียนโปรแกรมภาษา Python เพื่อที่จะนำมาคำนวณหาช่วงระดับความมั่นใจ (Confidence interval) ได้ โดยใช้ lib ของภาษานี้ ซึ่ง lib ที่เรานำมาใช้ได้แก่

1.matplotlib -> ใช้เพื่อวาดกราฟจากไฟล์ข้อมูล และจำแนกได้กราฟได้หลายรูปแบบ

2.pandas -> ใช้เพื่ออ่านข้อมูลที่เป็นตัวเลข (numeric data) จากไฟล์ csv ได้

3.statistic -> ใช้เพื่อคำนวณหาค่าข้อมูลเชิงสถิติต่างๆ เช่น ค่าเฉลี่ย ส่วนเบี่ยงเบนมาตรฐาน เป็นต้น

4.math -> ใช้เพื่อคำนวณแล้วนำมาใช้กับสูตรระดับความมั่นใจ เช่น หารากที่สอง (sqrt)

Code : python

Confidence_Interval(CI).py > ...

```python
import matplotlib.pyplot as plt
import pandas as pd
import math as M
import statistics
df = pd.read_csv('nutrition.csv')
df = df[['calories']]
y = 'calories'

Calories_mean = df['calories'].mean()
Calories_Std = df['calories'].std()
Calories_max = df['calories'].max()

print('Calories mean is : ' + str('{:.2f}'.format(Calories_mean)))
print('Calories standard deviation of salaries : ' + str('{:.2f}'.format(Calories_Std)))
print('---------------------------------------------------')
print('Formula is (mean - z)*(S.D./sqrt(n)) for lower bound')
print('Formula is (mean + z)*(S.D./sqrt(n)) for upper bound')
lowerBoundlevel90 = (Calories_mean-1.645)*(Calories_Std/M.sqrt(500))
upperBoundlevel90 = (Calories_mean+1.645)*(Calories_Std/M.sqrt(500))
lowerBoundlevel95 = (Calories_mean-1.96)*(Calories_Std/M.sqrt(500))
upperBoundlevel95 = (Calories_mean+1.96)*(Calories_Std/M.sqrt(500))
lowerBoundlevel99 = (Calories_mean-2.576)*(Calories_Std/M.sqrt(500))
upperBoundlevel99 = (Calories_mean+2.576)*(Calories_Std/M.sqrt(500))

print('---------------------------------------------------')
print('The interval of Confidence level at 90% :',str('{:.2f}'.format(lowerBoundlevel90)),'-',str('{:.2f}'.format(upperBoundlevel90)))
print('The interval of Confidence level at 95% :',str('{:.2f}'.format(lowerBoundlevel95)),'-',str('{:.2f}'.format(upperBoundlevel95)))
print('The interval of Confidence level at 99% :',str('{:.2f}'.format(lowerBoundlevel99)),'-',str('{:.2f}'.format(upperBoundlevel99)))
data_dict = {}
data_dict['ConfidenceLevel'] = ['confidence level (90%)','confidence level (95%)','confidence level (99%)']
data_dict['lower'] = [lowerBoundlevel90,lowerBoundlevel95,lowerBoundlevel99]
data_dict['upper'] = [upperBoundlevel90,upperBoundlevel95,upperBoundlevel99]
dataset = pd.DataFrame(data_dict)

plt.plot((lowerBoundlevel90,upperBoundlevel90),(0,0),'ro-',color = 'green')
plt.plot((lowerBoundlevel95,upperBoundlevel95),(1,1),'ro-',color = 'blue')
plt.plot((lowerBoundlevel99,upperBoundlevel99),(2,2),'ro-',color = 'red')
plt.yticks(range(len(dataset)),list(dataset['ConfidenceLevel']))
plt.xlabel('Calories (kcal)')
plt.ylabel('Confidence level')
plt.title('Calories per 100 g of food (Confidence inteval graph)')
plt.xlim(2000,2150)
plt.grid()
plt.show()
```

บรรทัดที่ 1-4 -> จะเป็นการ import lib เพื่อนำมาใช้งานด้านต่างๆ

บรรทัดที่ 5-11 -> จะเป็นการดึงข้อมูลตัวเลขมาจากไฟล์ csv ซึ่งข้อมูลจะเป็นประเภทตัวเลข

บรรทัดที่ 13-28 -> จะเป็นการคำนวณเชิงสถิติ โดยเราใช้ statistics lib เพื่อนำมาช่วยคำนวณ CI ด้วย

บรรทัด 29-33 -> จะเป็นการกำหนดขอบล่างและขอบบนให้กับ CI แต่ละช่วงว่าถ้าระดับความมั่นใจช่วงนี้ควรมีจะมีขอบเขตเป็นเท่าไหร่ ใช้ค่า z เป็นเท่าไหร่ *ค่า z สามารถเปิดตารางในเว็บดูได้* โดย CI 90% จะใช้ค่า z = 1.645 / CI 95% จะใช้ค่า z = 1.96 / CI 99% จะใช้ค่า z = 2.576

บรรทัดที่ 35-44 -> จะเป็นการพลอตกราฟจากข้อมูลที่เราคำนวณมาในบรรทัดก่อนหน้า ซึ่งจะใช้ matplotlib มาช่วยในการสร้างกราฟ

ผลลัพธ์ของโค้ด

```
Calories mean is : 235.38
Calories standard deviation of salaries : 196.37
-----------------------------------------------------
Formula is (mean - z)*(S.D./sqrt(n)) for lower bound
Formula is (mean + z)*(S.D./sqrt(n)) for upper bound
-----------------------------------------------------
The interval of Confidence level at 90% : 2052.66 - 2081.55
The interval of Confidence level at 95% : 2049.89 - 2084.32
The interval of Confidence level at 99% : 2044.48 - 2089.73
```

บทวิเคราะห์

จากกราฟ เราคำนวณออกมาแล้วได้ว่า ขอบเขตล่างและขอบเขตบนของแต่ละช่วงแต่ระดับความมั่นใจมีกี่แคลรอลี่ ซึ่งจะสรุปได้ว่าถ้าระดับความมั่นใจอยู่ที่ 90% และเราสุ่มการทดลองไป 500 ครั้ง จะมีการครอบคลุมพารามิเตอร์อยู่ที่ 450 ครั้ง ก็คือ จำนวนแคลรอลี่จากการสุ่มอยู่ในช่วง 2052.66 – 2081.55 และไม่มีการครอบคลุมพารามิเตอร์อยู่ที่ 50 ครั้ง ก็คือ จำนวนแคลรอลี่จากการสุ่มไม่อยู่ในช่วง 2052.66-2081.55 ถ้าระดับความมั่นใจอยู่ที่ 95 % และเราสุ่มการทดลองไป 500 ครั้ง จะมีการคลอบคลุมพารามิเตอร์ จะมีการครอบคลุมพารามิเตอร์อยู่ที่475 ครั้ง ก็คือ จำนวนแคลรอลี่จากการสุ่มอยู่ในช่วง 2049.89 – 2084.32 และไม่มการรอบคลุมพารามิเตอร์อยู่ที่25 ครั้ง ก็คือ จำนวนแคลรอลี่จากการสุ่มไม่อยู่ในช่วง 2049.89 – 2084.32 ถ้าระดับความมั่นใจอยู่ที่ 99% และเราสุ่มการทดลองไป 500 ครั้ง จะมีการครอบคลุมพารามิเตอร์อยู่ที่ 495 ครั้ง ก็คือ จำนวนแคลรอลี่จากการสุ่มอยู่ในช่วง 2044.48 – 2089.73 และไม่มีการครอบคลุมพารามิเตอร์อยู่ที่ 5 ครั้ง ก็คือ จำนวนแคลรอลี่จากการสุ่มไม่อยู่ในช่วง 2044.48 – 2089.73 ซึ่งค่าที่เราสร้างขึ้นนั้นจะเป็นการทดลองที่เป็นแบบช่วงนั่นเอง

Diabetes (HW-1)

ชื่อคอลัมน์ 3 คอลัมน์ที่สนใจ

- Age มีหน่วยเป็น ปี
- Glucose มีหน่วยเป็น mmol/L
- Body Mass Index (BMI) มีหน่วยเป็น (กิโลกรัม/ส่วนสูง) ยกกำลังสอง

Why is it interesting?

- โรคเบาหวาน (Diabetes) เป็นโรคที่คนอินเดียมากอันดับต้นๆ และมีแนวโน้มว่าคนอินเดียจะเป็นมากขึ้น ทำให้เกิดความสนใจในการศึกษาเรื่องนี้

แหล่งที่มาข้อมูล

- Pima Indians Diabetes Database | Kaggle

วิธีการรวบรวมข้อมูล

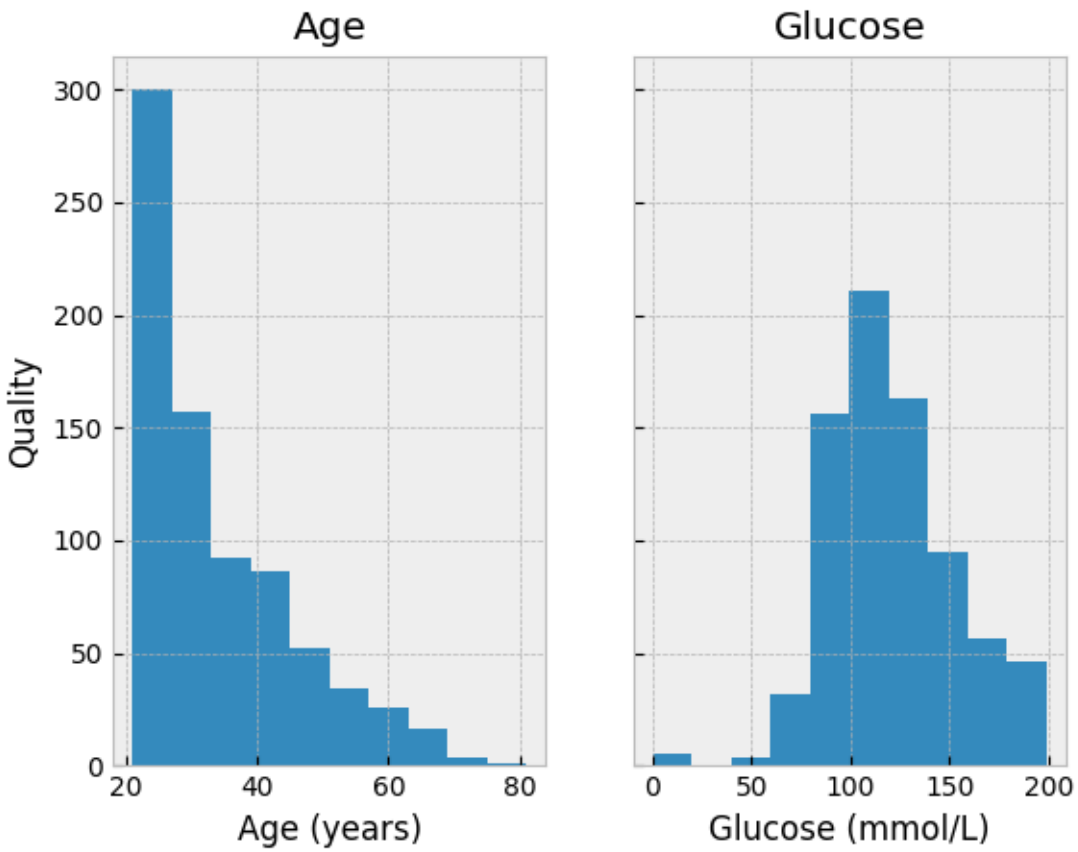
- ชุดข้อมูลนี้มาจากสถาบันโรคเบาหวานแห่งชาติและระบบทางเดินอาหารและโรคไตของอินเดีย

Diabetes (HW-2)

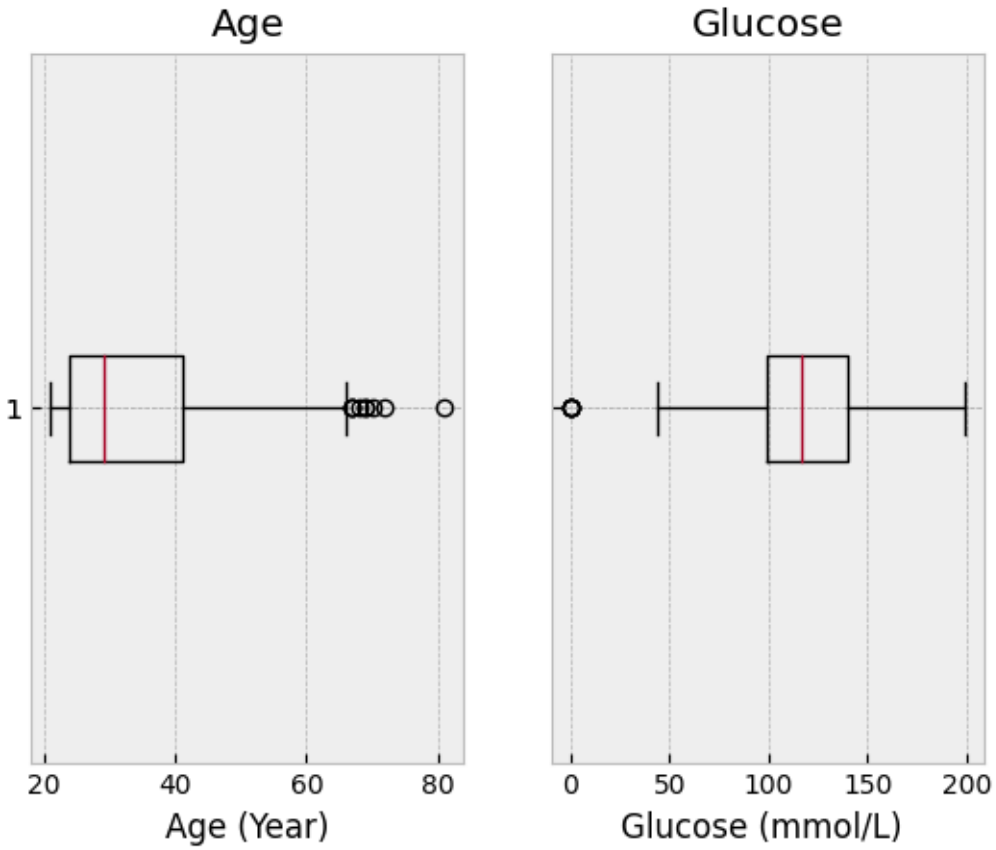
- ใช้ column : Age-อายุ (Year), Glucose-จำนวนน้ำตาลในเลือด (mmol/L)
- ค่าสถิติต่างๆ

```
Age mean : 33.241
Glucose mean : 120.895
Age mode : 22
Glucose mode : 100
Age median : 29.0
Glucose median : 117.0
Age sample standard deviatuon : 11.76
Glucose sample standard deviation : 31.973
```

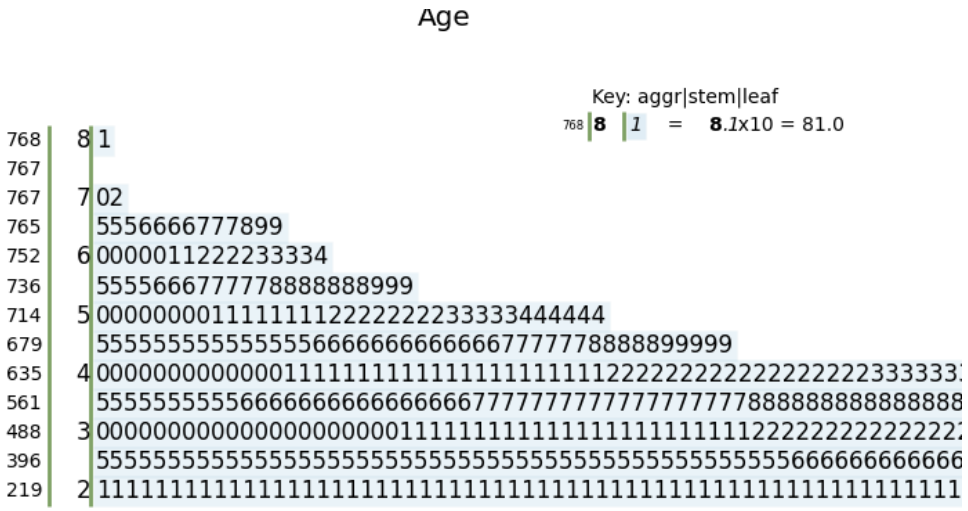
- กราฟต่างๆ
- Histogram



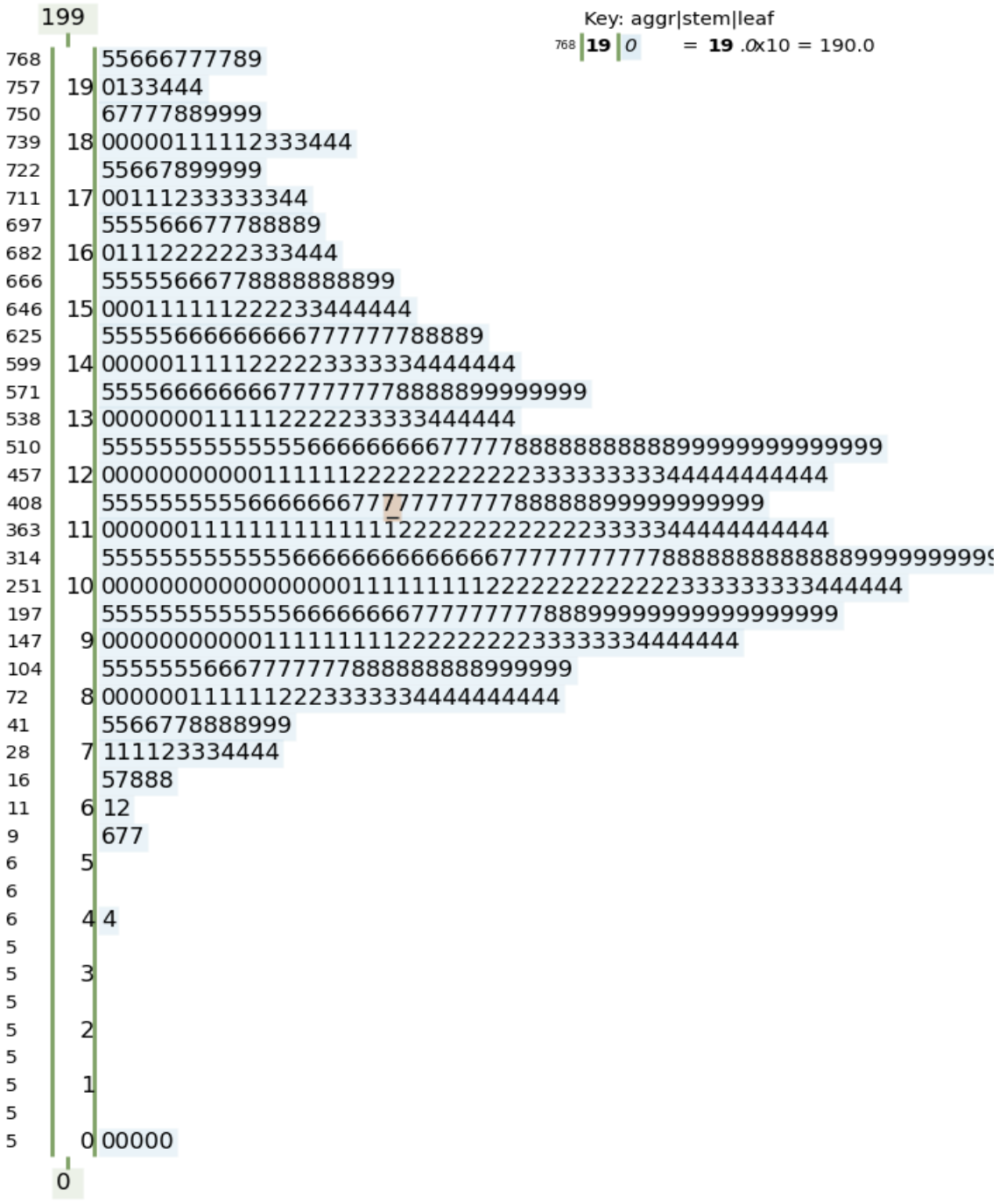
- Box plots


Age
Glucose
1
20
40
60
80
Age (Year)
0
50
100
150
200
Glucose (mmol/L)

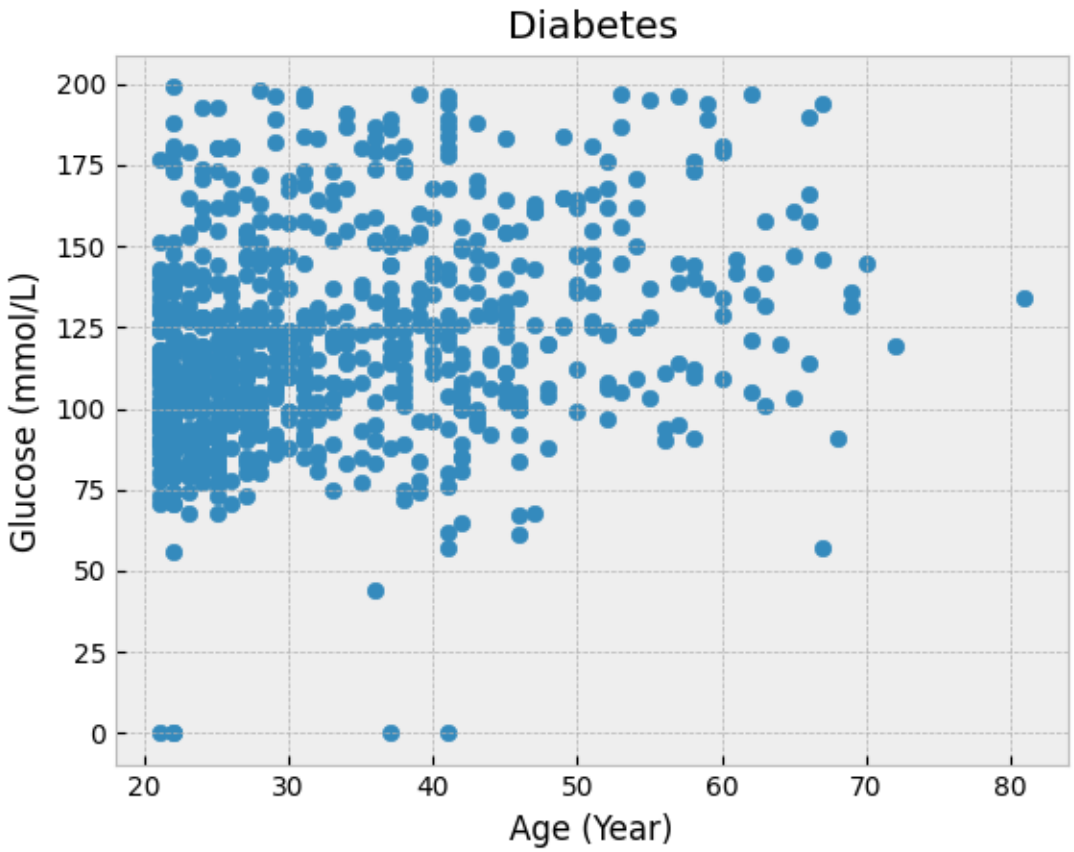
- Stem and Leaf



Glucose


199
Key: aggr|stem|leaf
768 19 0 = 19 .0x10 = 190.0
768 55666777789
757 19 0133444
750 67777889999
739 18 00000111112333444
722 55667899999
711 17 00111233333344
697 555566677788889
682 16 0111222222333444
666 55555666778888888899
646 15 000111111222233444444
625 555556666666667777777788889
599 14 00000111112222233333344444444
571 555566666666777777778888899999999
538 13 0000000111112222233333444444
510 555555555555556666666667777788888888888999999999999999
457 12 000000000001111112222222222223333333334444444444
408 55555555556666666777777777888888999999999999
363 11 00000011111111111112222222222223333334444444444
314 555555555555566666666666666677777777777888888888888899999999999
251 10 000000000000000001111111112222222222222333333333444444
197 55555555555556666666677777777788899999999999999999
147 9 000000000001111111112222222223333333444444
104 55555556667777777888888888999999
72 8 00000011111122233333444444444444
41 5566778888999
28 7 111123334444
16 57888
11 6 12
9 677
6 5
6
6 4 4
5
5 3
5
5 2
5
5 1
5
5 0 00000
0

- Scatter



- ตัวแปรแกน x : อายุของผู้ป่วย
- ตัวแปรแกน y : ปริมาณน้ำตาลในเลือด
- เหตุผล
    - เหตุผลที่ใช้ อายุ และ ปริมาณน้ำตาลในเลือดเป็น ตัวแปรเพราะต้องการศึกษาว่า ปริมาณน้ำตาลในเลือดเท่าไรถึงมีโอกาสเป็นโรคเบาหวาน
- Outlier
    - ตามแนวแกน x : Age -อายุ 80 ปี
    - ตามแนวแกน y : Glucose -ปริมาณน้ำตาลในเลือด 0 mmol/L มีอยู่ 5 จุด

บทวิเคราะห์ข้อมูล

- จากข้อมูลที่ได้มาจะเห็นได้ว่า ช่วงที่เป็นโรคเบาหวานมากที่สุดจะอยู่ในช่วง 21-42 ปี และจำนวนน้ำตาลในร่างกายที่ผู้ป่วยส่วนมากมีคือ 100-140 mmol/L นี้แสดงให้เห็นว่าในช่วงอายุ 21-40ปี มีโอกาสเป็นโรคเบาหวานสูง ส่วนคนที่มีน้ำตาลในเส้นเลือดมี น้ำตาลในเส้นเลือดอยู่ระหว่าง 100-120 mmol/L ก็มีโอกาสเป็นโรคเบาหวานสูงเช่นเดียวกัน

จึงวิเคราะห์ได้ว่าอายุไม่มีผลต่อการเป็นโรคเบาหวาน ถ้าน้ำตาลในเส้นเลือดนั้นมีมาก จะทำให้มีอาการเป็นโรคเบาหวานมากขึ้นไปด้วย

- Source Code

```
import statistics as stc
import matplotlib.pyplot as plt
import pandas as pd

plt.style.use('bmh')
df = pd.read_csv('diabetes.csv')

# age glucose BMI
x = df['Age']
y = df['Glucose']
z = df['BMI']

# convert to list
age = x.to_list()
glucose = y.to_list()
bmi = z.to_list()

def mean(data):
    print(stc.mean(data))

def mode(data):
    print(stc.multimode(data))

def median(data):
    print(stc.median(data))

# # mean
# age_mean = stc.mean(age)
# print('Age mean : ' + str(age_mean))

# glucose_mean = stc.mean(glucose)
# print('Glucose mean : ' + str(glucose_mean))

# #mode
# age_mode = stc.multimode(age)
# print('Age mode : ' + str(*age_mode))

# glucose_mode = stc.mode(glucose)
# print('Glucose mode : ' + str(glucose_mode))

# # median
# age_median = stc.median(age)
# print('Age median : ' + str(age_median))
```

```
•
•   # glucose_median = stc.median(glucose)
•   # print('Glucose median : ' + str(glucose_median))
•
•   # Sample standard deviation
•   age_ssd = stc.stdev(age)
•   print('Age sample standard deviatuon : ' + str(age_ssd))
•
•   glucose_ssd = stc.stdev(glucose)
•   print('Glucose sample standard deviation : ' + str(glucose_ssd))
•
•   # histogram
•
•   # plt.xlabel('Age')
•   # plt.ylabel('Interval')
•   # plt.title('Age')
•   # plt.hist(age, bins=10)
•   # plt.show()
•
•   # fig, ax = plt.subplots(1, 2, sharey=True)
•
•   # ax[0].set_title('Age (years)')
•   # ax[0].hist(age)
•   # ax[0].set_title('Age')
•   # ax[0].set_xlabel('Age (years)')
•   # ax[0].set_ylabel('Quality')
•
•   # ax[1].set_title('Glucose')
•   # ax[1].hist(glucose)
•   # ax[1].set_xlabel('Glucose (mmol/L)')
•
•   # plt.show()
•
•   # box plot
•   # fig, ax = plt.subplots(1, 2, sharey=True)
•   # ax[0].set_title('Age')
•   # ax[0].boxplot(age, vert=False)
•   # ax[0].set_xlabel('Age (Year)')
•
•   # ax[1].set_title('Glucose')
•   # ax[1].boxplot(glucose, vert=False)
•   # ax[1].set_xlabel('Glucose (mmol/L) ')
•
•   # plt.show()
•
```

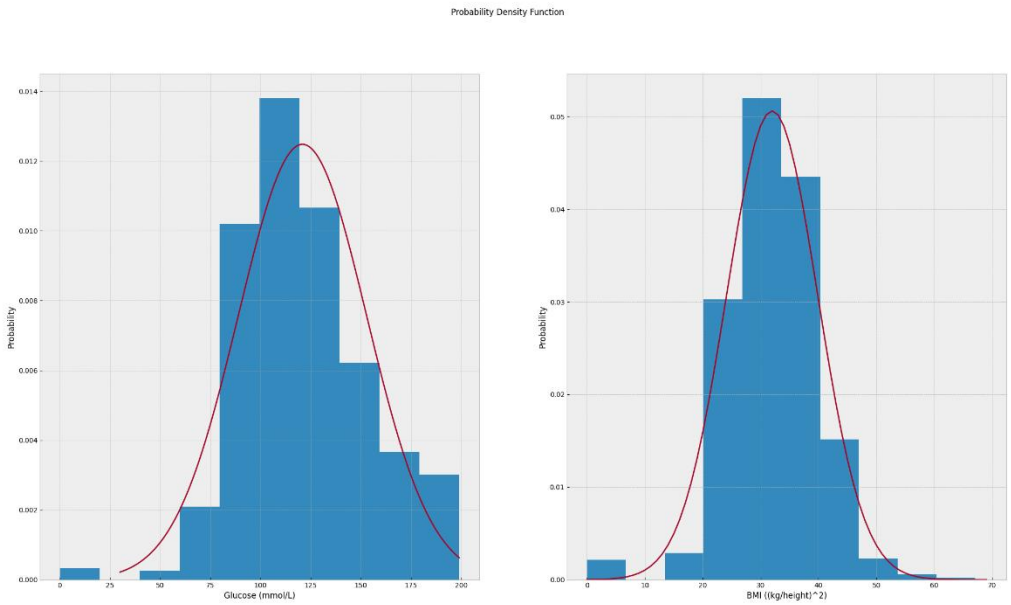
```
# stem and leave
# ls = [i for i in range(1,10)]
# print(ls)
# fig, ax = plt.subplots(2)
# ax[0].set_title('Age')
# ax[0].stem(age, ls)

# ax[1].set_title('Glucose')
# ax[1].stem(glucose)

# plt.show()

# import stemgraphic

# fig, ax = stemgraphic.stem_graphic(df['Age'])
# plt.title('Age')

# plt.show()

# fig, ax = stemgraphic.stem_graphic(df['Glucose'])
# plt.title('Glucose')

# plt.show()

# scatter
# fig, ax = plt.subplots(2)
# ax[0].set_title('Age')
# ax[0].scatter(x, y)

# ax[1].set_title('Glucose')
# ax[1].scatter(x, y)

# plt.xlabel('Age (Year)')
# plt.ylabel('Glucose (mmol/L)')
# plt.title('Diabetes')
# plt.scatter(x, y)

# plt.show()
```

Diabetes (HW-3)

Column ที่เลือกใช้คือ

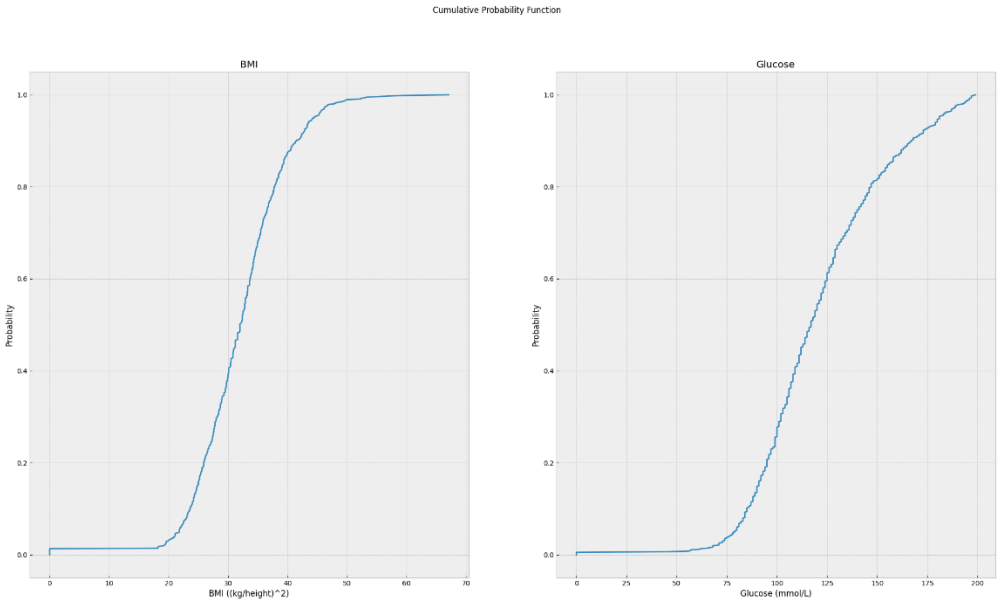
- Glucose -จำนวนน้ำตาลในเลือด หน่วย mmol/L
- BMI -ค่าดัชนีมวลกาย หน่วย $(\frac{kg}{height(m)})^2$

Probability Density Function



จากกราฟฟังก์ชันความหนาแน่นของความน่าจะเป็นของกราฟ Glucose จะเห็นได้ว่าจุดสูงสุดของกราฟอยู่ที่ 121.0 mmol/L ซึ่งทำให้เห็นว่าจำนวนน้ำตาลในเลือดของผู้ป่วยที่เป็นโรคเบาหวานมากที่สุดอยู่ที่ 121.0 mmol/L และ จากกราฟฟังก์ชันความหนาแน่นของความน่าจะเป็นของกราฟ BMI จะเห็นได้ว่าจุดสูงสุดของกราฟอยู่ที่ 32.0 $(\frac{kg}{height(m)})^2$ซึ่งทำให้เห็นว่าดัชนีมวลกายของผู้ป่วยที่เป็นโรคเบาหวานมากที่สุดอยู่ที่ 32.0 $(\frac{kg}{height(m)})^2$

Cumulative Probability Function



จากกราฟ CPF ของ BMI จะเห็นได้ว่าช่วงแรกของกราฟยังไม่ชันมาก จะชันขึ้นอย่างมากในช่วง 19 ถึง 50 $(\frac{kg}{height(m)})^2$เพราะข้อมูลส่วนใหญ่จะสะสมอยู่ในช่วงนี้ และในช่วง 50 ขึ้นไปจะเห็นได้ว่าความชันลดลงเรื่อย ๆ และในส่วนกราฟ CPF ของ Glucose จะเห็นได้ว่าช่วงแรกของกราฟยังไม่ชันมาก จะชันขึ้นอย่างมากในช่วง 60 mmol/L ขึ้นไปและหยุดในจุด 200 mmol/L

Source code

PDF

```
import matplotlib.pyplot as plt
import pandas as pd
from scipy.stats import norm
import statistics as stc

plt.style.use('bmh')
df = pd.read_csv('diabetes.csv')

# age glucose BMI
x = df['Age']
y = df['Glucose']
z = df['BMI']

# convert to list
age = x.to_list()
glucose = y.to_list()
bmi = z.to_list()

data = glucose
data2 = bmi

# calculate parameters
sample_mean = stc.mean(data)
sample_std = stc.stdev(data)
print('Mean=%.3f, Standard Deviation=%.3f' % (sample_mean, sample_std))

sample_mean2 = stc.mean(data2)
sample_std2 = stc.stdev(data2)

# define the distribution
dist = norm(sample_mean, sample_std)
dist2 = norm(sample_mean2, sample_std2)

# sample probabilities for a range of outcomes
values = [value for value in range(30, 200)]
probabilities = [dist.pdf(value) for value in values]

values2 = [value2 for value2 in range(0, 70)]
probabilities2 = [dist2.pdf(value2) for value2 in values2]

fig, ax = plt.subplots(1, 2)
```

```
fig.suptitle('Probability Density Function')

# plot the histogram and pdf
ax[0].hist(data, bins=10, density=True)
ax[0].plot(values, probabilities)
ax[0].set_xlabel('Glucose (mmol/L)')
ax[0].set_ylabel('Probability')

ax[1].hist(data2, bins=10, density=True)
ax[1].plot(values2, probabilities2)
ax[1].set_xlabel('BMI ((kg/height)^2)')
ax[1].set_ylabel('Probability')

plt.show()
```

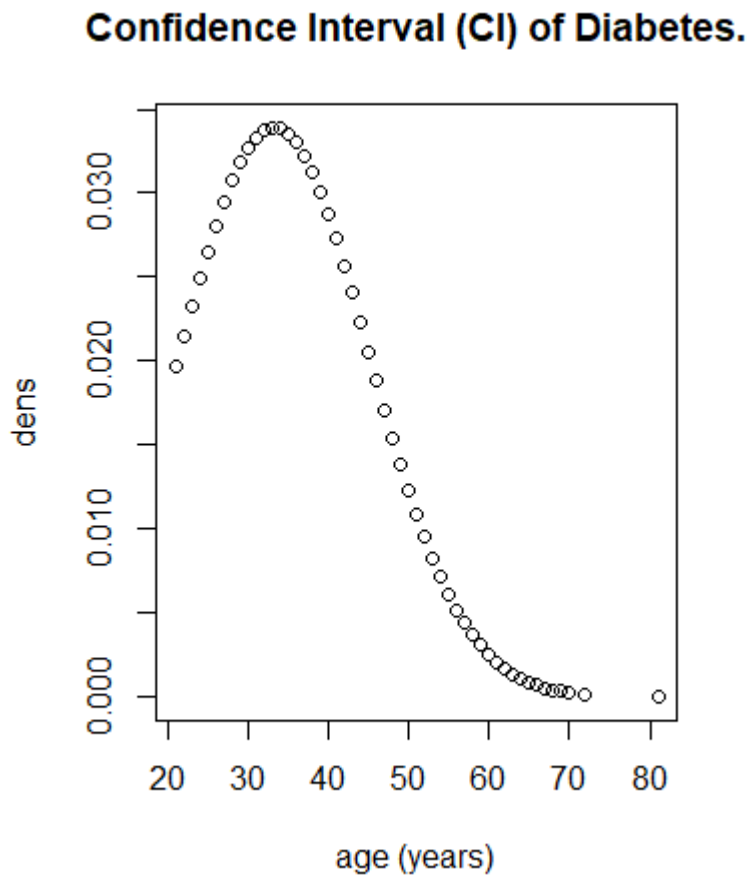
CFP

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

plt.style.use('bmh')
df = pd.read_csv('diabetes.csv')

# read data
x = df['Age']
y = df['Glucose']
z = df['BMI']


# to list
age = x.to_list()
glucose = y.to_list()
bmi = z.to_list()

data = np.array(bmi)
data2 = np.array(glucose)
data.sort()
data2.sort()

# https://www.youtube.com/watch?v=fQ0Iy0Sew_U

# yvals = np.zeros(len(data))
# for i in range(len(data)):
```

```
#     yvals[i] = (i+1)/len(yvals)
# plt.plot(data, yvals, 'k')

# https://stackoverflow.com/questions/24788200/calculate-the-cumulative-
distribution-function-cdf-in-python

p = 1. * np.arange(len(data)) / (len(data) - 1)
p2 = 1. * np.arange(len(data2)) / (len(data) - 1)

fig, ax = plt.subplots(1, 2)
fig.suptitle('Cumulative Probability Function')

ax[0].plot(data, p)
ax[0].set_title('BMI')
ax[0].set_xlabel('BMI ((kg/height)^2)')
ax[0].set_ylabel('Probability')

ax[1].plot(data2, p2)
ax[1].set_title('Glucose')
ax[1].set_xlabel('Glucose (mmol/L)')
ax[1].set_ylabel('Probability')


plt.show()
```

Diabetes (HW-4)

- เลือก column : Age-อายุ(Years)
- กราฟ Confidence Interval of Diabetes.



- Confidence Interval 90% อยู่ในช่วง 32.54203 ถึง 33.93974 ปี

```
> t.test(age, conf.level = .90)

        One Sample t-test

data:  age
t = 78.332, df = 767, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 32.54203 33.93974
sample estimates:
mean of x
 33.24089
```

- Confidence Interval 95% อยู่ในช่วง 32.40784 ถึง 34.073 ปี

```
> t.test(age, conf.level = .95)

        One Sample t-test

data:  age
t = 78.332, df = 767, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 32.40784 34.07393
sample estimates:
mean of x
 33.24089
```

- Confidence Interval 99% อยู่ในช่วง 32.14508 ถึง 34.336 ปี

```
> t.test(age, conf.level = .99)

        One Sample t-test

data:  age
t = 78.332, df = 767, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
99 percent confidence interval:
 32.14508 34.33669
sample estimates:
mean of x
 33.24089
```

บทวิเคราะห์ข้อมูล

- จากการคำนวณช่วงความเชื่อมั่นทั้ง 3 ระดับ ได้แก่ 90%, 95% และ 99% ได้ค่าของความเชื่อมั่นตามนี้ ในช่วง 90% จะคลอบคลุมข้อมูลในช่วง 32.542 ถึง 33.939 ปี, 95% จะคลอบคลุมข้อมูลในช่วง 32.407 ถึง 34.073 ปี และ 99% จะคลอบคลุมข้อมูลในช่วง 32.145 ถึง 34.336 ปี
- สรุปได้ว่า จากข้อมูลชุดนี้คนที่เป็นโรคเบาหวานส่วนใหญ่ 99% อยู่ในช่วงอายุ 32.145 ถึง 34.336 ปี คนที่เป็นโรคเบาหวานส่วนใหญ่ 95% อยู่ในช่วงอายุ 32.407 ถึง 34.073 ปี และ คนที่เป็นโรคเบาหวานส่วนใหญ่ 90% อยู่ในช่วงอายุ 32.542 ถึง 33.939 ปี

Source Code

```
x <- diabetes$Age

t.test(x, conf.level = .99)

t.test(x, conf.level = .95)

t.test(x, conf.level = .90)
```

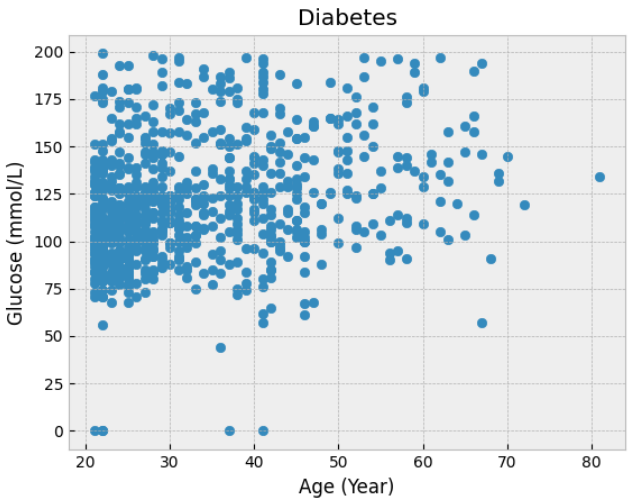
```
        age_mean <- mean(x)

        age_sd <- sd(x)

        dens <- dnorm(x, age_mean, age_sd)

plot(age, dens, xlab = "age (years)", main = "Confidence Interval (CI) of Diabetes.")
```

Diabetes (Homework 5)

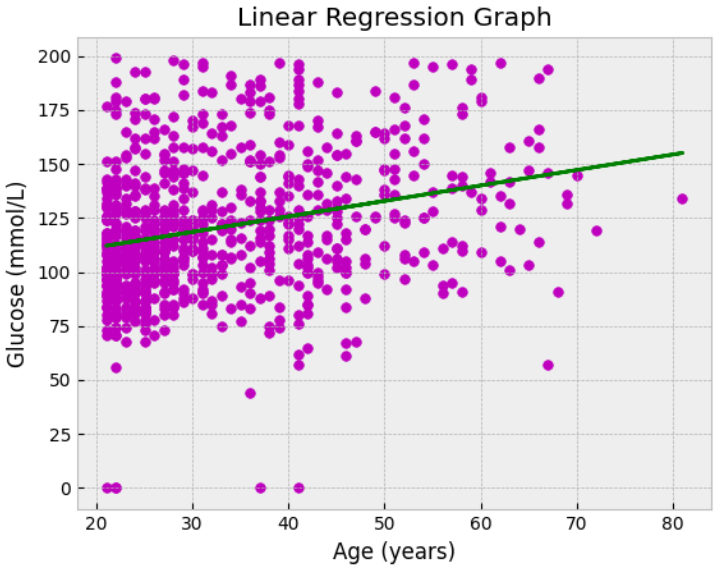
ภาษาที่ใช้ : Python 3

หัวข้อที่เลือกใช้ : Diabetes (หัวข้อของ สหทัศน์)

Column ที่เลือกใช้ : Age (อายุ) , Glucose (ระดับน้ำตาลในเลือด)



Scatter graph



Linear Regression Graph

<u>ในแกน x ของกราฟ</u> -> แสดงช่วงอายุของผู้คน (Age) มีหน่วยเป็นปี

<u>ในแกน y ของกราฟ</u> -> แสดงถึงปริมาณความเข้มข้นของน้ำตาลในเลือด (Glucose) มีหน่วยเป็นโมลลาร์

Code : Python 3

linear_regression.py ×

Glucose-age > Stat-main > linear_regression.py > ...

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

plt.style.use('bmh')
df = pd.read_csv('diabetes.csv')


def estimate_coef(x, y):
    # number of observations/points
    n = np.size(x)

    # mean of x and y vector
    m_x = np.mean(x)
    m_y = np.mean(y)

    # calculating cross-deviation and deviation about x
    SS_xy = np.sum(y*x) - n*m_y*m_x
    SS_xx = np.sum(x*x) - n*m_x*m_x

    # calculating regression coefficients
    b_1 = SS_xy / SS_xx
    b_0 = m_y - b_1*m_x

    return (b_0, b_1)
```

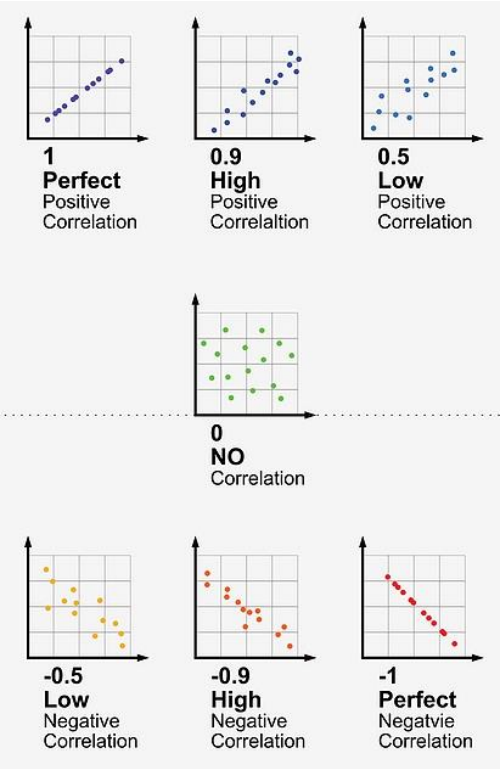
```python
def plot_regression_line(x, y, b):
    # plotting the actual points as scatter plot
    plt.scatter(x, y, color="m",
                marker="o", s=30)

    # predicted response vector
    y_pred = b[0] + b[1]*x

    # plotting the regression line
    plt.plot(x, y_pred, color="g")

    # putting labels
    plt.xlabel('Age (years)')
    plt.ylabel('Glucose (mmol/L)')

    plt.title('Linear Regression Graph')

    # function to show plot
    plt.show()

def main():
    # observations / data
    x = df['Age']
    y = df['Glucose']

    # estimating coefficients
    b = estimate_coef(x, y)
    print("Estimated coefficients:\nb_0 = {} nb_1 = {}".format(b[0], b[1]))

    # plotting regression line
    plot_regression_line(x, y, b)

if __name__ == "__main__":
    main()
```

บทวิเคราะห์

จากกราฟ linear regression ที่ได้แสดงออกมานั้น จะเห็นได้ว่า กราฟมีส่วนประกอบอยู่สองส่วนก็คือ แกน x แสดงอายุของผู้คน (years) และ แกน y แสดงระดับปริมาณน้ำตาลในเลือด (glucose) ซึ่งเส้นการถดถอยนั้น มีค่าความชันอยู่ประมาณที่ 0.711 ซึ่งก็คือค่าบวก นั่นคือการบ่งบวกว่าตัวแปรทั้งสองตัวมีความสันพันธ์ค่อนข้างมาก (แปรผันตรง) เช่น ถ้าผู้คนอายุ x ก็จะมีระดับน้ำตาลอยู่ที่ y ถ้าผู้คนอายุ x + 1 ก็จะมีระดับน้ำตาลอยู่

ที่ y + 0.7 เป็นต้น แต่ทั้งนี้ตัวแปรทั้งสองนั้นไม่ได้เป็นปัจจัย หรือ เป็นเหตุผลให้กันและกัน เนื่องจาก ผู้คนอายุสูงและผู้คนอายุต่ำนั้นมีร่างกายที่ไม่เหมือนกัน นี่เป็นแค่การทดลองและการรีเสิชจากข้อมูลส่วนนึงเท่านั้น ว่าข้อมูลนี้มีข้อมูลเป็นอย่างไรบ้าง จะมีแนวโน้มในรูปแบบไหน

รูปแบบของ Coefficient of Correlation



จากรูปแบบกราฟ เรามีค่า Coefficient of Correlation ที่มีค่าความชันอยู่ที่ประมาณ 0.7
จึงสรุปได้ว่า กราฟที่ได้จากการทดลอง มีความสัมพันธ์อยู่ใกล้กลับ High Positive Correlation

เหตุผลที่นำหัวข้อ “โรคเบาหวาน (Diabetes)” **มาทำการทดลอง**

ในการใช้ชีวิตของพวกเรานั้น พวกเราควรหมั่นดูและสุขภาพร่างกายให้แข็งแรง เพื่อที่จะให้ใช้ชีวิตอย่างสะดวกสบาย ไม่เจ็บป่วยได้ง่าย โรคเบาหวาน เป็นโรคหนึ่งที่ก่อให้เกิดอันตรายกับผู้ที่มีอายุสูงเป็นจำนวนอย่างมาก เช่น ก่อให้เกิดภาวะเฉียบพัน ก่อให้เกิดโรคความดันโลหิต และอีกมากมาย ที่พวกเราเลือกวิเคราะห์ข้อมูลเหล่านี้ ก็เพราะว่า เราอยากให้ผู้คนได้รับรู้ว่า เรามีจำนวนผู้คนที่เสี่ยงต่อการเป็นโรคเบาหวานอย่างมาก จากการทดลองครั้งนี้ เราควรแนะนำผู้คนมากขึ้นเพื่อไม่ให้ผู้คนเสี่ยงต่อการมีโรคเบาหวานนี้

สรุปผลการศึกษาและเสนอแนะแนวทางการศึกษา

ในการศึกษาครั้งนี้ เราได้ทราบเกี่ยวกับการวิเคราะห์ข้อมูลเชิงกราฟหลากหลายรูปแบบ ได้แก่ histogram, scatter, box plot, bar graph และอื่นๆ อีกมากมาย ทั้งนี้ อยากจะแนะนำให้ หมั่นรีเสิชข้อมูลเพิ่มเติมจากเว็บไซต์ให้มากขึ้น เพื่อที่จะเป็นความรู้รอบตัวได้ อีกเรื่องหนึ่งที่จะกล่าวถึงคือควรที่จะใช้หลักสูตรในการสอนเดียวกันครับ