การเปรียบเทียบระหว่างข้อมูลที่ยังไม่ผ่านกระบวนการ PCA และข้อมูลที่ผ่านกระบวนการ PCA โดยใช้ชุดข้อมูล Unsupervised Learning on Country Data ผ่านกระบวนการจัดกลุ่ม (Clustering) ด้วยโปรแกรม R

จัดทำโดย

นายสิทธัตกะ จรัสแสง 665020060-6

เสนอ

อ.ดร.เปรม จันทร์สว่าง

รายงานนี้เป็นส่วนหนึ่งของรายวิชา Multivariate

ปีการศึกษา 1/2566

มหาวิทยาลัยขอนแก่น

คำนำ

รายงานเล่มนี้เป็นส่วนหนึ่งของรายวิชา SC637703 Multivariate Analysis เพื่อให้ได้ความรู้ในเรื่องการทำข้อมูลด้วยวิธี Principal Component Analysis ควบคู่กับการจัดกลุ่ม Clustering และได้ศึกษาอย่างเข้าใจเพื่อเป็นประโยชน์กับการเรียน

ผู้จัดทำหวังว่ารายงานเล่มนี้จะเป็นประโยชน์กับผู้อ่าน หากมีข้อเสนอแนะหรือข้อผิดพลาดประการใด ขออภัยมา ณ ที่นี้ด้วย

นายสิทธัตกะ จรัสแสง

สารบัญ

1. Principal Component Analysis (PCA)

อะไรคือ Principal Component Analysis

Principal Component Analysis เป็นเทคนิคทางสถิติและคณิตศาสตร์ที่ใช้ในการลดมิติของข้อมูลหรือลดจำนวนตัวแปรในข้อมูลให้น้อยลงโดยไม่สูญเสียข้อมูลหลัก. PCA ถูกใช้อย่างแพร่หลายในการวิเคราะห์ข้อมูลและการประยุกต์ในหลายงานเช่นการปรับปรุงการระบุรูปร่าง, การจัดระเบียบข้อมูล, การลดรูปเมาท์และอื่น ๆ อีกมากมาย.

ลักษณะหลักของ PCA คือดัชนีหลักที่มีความสัมพันธ์กันน้อยที่สุด (principal components) ซึ่งเป็นคอมโพเนนต์ที่สามารถแสดงความแปรปรวนของข้อมูลในทิศทางที่แตกต่างกันได้มากที่สุด โดยไม่สูญเสียข้อมูลสำคัญ. PCA ทำงานโดยหา vector หรือแกนหลักเหล่านี้ที่ทำให้ข้อมูลกระจุดมุ่งหมุนข้างไปในทิศทางที่แตกต่างกันและมีความแปรปรวนมากที่สุด.

ข้อดีของการทำ PCA คือ

1. ลดมิติข้อมูล: PCA ช่วยลดจำนวนมิติของข้อมูลได้อย่างมีประสิทธิภาพ ซึ่งช่วยลดความซับซ้อนของข้อมูล และช่วยในการจัดเก็บและประมวลผลข้อมูลได้ดีขึ้น เช่นถ้าคุณมีข้อมูลที่มีหลายพันตัวแปร คุณสามารถลดมิติให้เหลือเพียงไม่กี่มิติที่สำคัญเท่านั้น ทำให้ข้อมูลมีขนาดเล็กลง และการทำงานกับข้อมูลเหล่านี้จะง่ายขึ้น.
2. การหาความสัมพันธ์: PCA ช่วยในการหาความสัมพันธ์ระหว่างตัวแปร แม้ว่ามันจะลดมิติของข้อมูล แต่มันยังคงรักษาความสัมพันธ์ระหว่างข้อมูล นี้ช่วยในการค้นพบโครงสร้างและความสัมพันธ์ในข้อมูล.
3. การแสดงข้อมูล: PCA ช่วยในการแสดงข้อมูลในรูปแบบที่อย่างง่ายที่จะเข้าใจ โดยลดมิติของข้อมูลลงมาที่สามารถพล็อตได้ในกราฟ และเป็นประโยชน์ในการวิเคราะห์ข้อมูล.
4. ทำให้การวิเคราะห์ข้อมูลเป็นไปได้: PCA ช่วยในการเตรียมข้อมูลสำหรับการวิเคราะห์อื่น ๆ โดยการลดมิติ และการลดความซับซ้อนของข้อมูล ทำให้การวิเคราะห์เช่น clustering หรือ classification ง่ายขึ้น.

ขั้นตอนการทำ Principal Component Analysis ใน R

1. ทำการหา PCA ด้วยคำสั่ง prcomp(data)
2. ตัดสินเลือกตัวแปรเมื่อค่า Cumulative มากกว่า 0.80 หรือ 80%
3. ทำการหา Correlation Circle ด้วย fviz_pca_var(res.pca)
4. และทำการ plot ด้วย individuals PCA จากสูตร fviz_pca_var(res.pca,target)

ในการทดสอบในครั้งนี้จะเป็นการเปรียบเทียบระหว่างข้อมูลที่ยังไม่ผ่านกระบวนการ PCA และข้อมูลที่ผ่านกระบวนการ PCA ด้วยวิธีการจัดกลุ่ม Clustering

2. Clustering

อะไรคือ Clustering

Clustering (การจัดกลุ่ม) เป็นกระบวนการที่ใช้ในการแบ่งข้อมูลหรือวัตถุในกลุ่มหรือคลัสเตอร์ที่มีความคล้ายคลึงกันในลักษณะหนึ่งหรือหลายลักษณะ โดยมุ่งเน้นที่จะทำให้วัตถุในคลัสเตอร์เดียวกันคล้ายกันมากและที่แตกต่างกันมากจากวัตถุในคลัสเตอร์อื่น ๆ หรือกลุ่มอื่น ๆ โดยการใช้ลักษณะหรือคุณสมบัติที่ระบุไว้ เช่น ค่าเลขคณิต, ลักษณะกราฟ, ความคล้ายคลึงในพื้นที่หรือการกระจาย, หรือคุณสมบัติอื่น ๆ ที่เกี่ยวข้อง. ในการทำ Clustering ครั้งนี้จะทำการสนใจอยู่ทั้งหมด 2 ส่วนหลักๆได้แก่

2.1 ประเภทของการวัดระยะห่าง

มีหลายประเภทของสูตรคำนวณระยะทาง (distance) ที่ใช้ในหลายแนวทางต่าง ๆ เช่น ในการคำนวณระยะทางระหว่างจุดหรือวัตถุต่าง ๆ ในพื้นที่หรือข้อมูลที่มีตัวแปรต่าง ๆ ทั้งนี้เราขอนำเสนอ 2 วิธีการได้แก่

1. ระยะยูคลิด (Euclidean Distance): สูตรคำนวณระยะยูคลิดคือวิธีที่ใช้บ่งบอกระยะทางระหว่างสองจุดในพื้นที่มิติหลาย ๆ มิติ โดยใช้ตรรกะสามเหลี่ยมของพื้นที่ เป็นสูตรที่คำนวณระยะทางในมิติสองมิติมักจะใช้สูตรพิทาโกรัสเพื่อคำนวณ.
2. ระยะเชิงมนุษย์ (Manhattan Distance): ระยะเชิงมนุษย์คำนวณระยะทางระหว่างสองจุดโดยการบวกระยะทางแนวแกน X และ Y โดยไม่คำนึงถึงการเคลื่อนที่แบบทแยงมุม ระยะเชิงมนุษย์มักใช้ในการคำนวณระยะทางในเมืองเมื่อคุณจำเป็นต้องเดินหรือขับรถตามถนนตาราง.

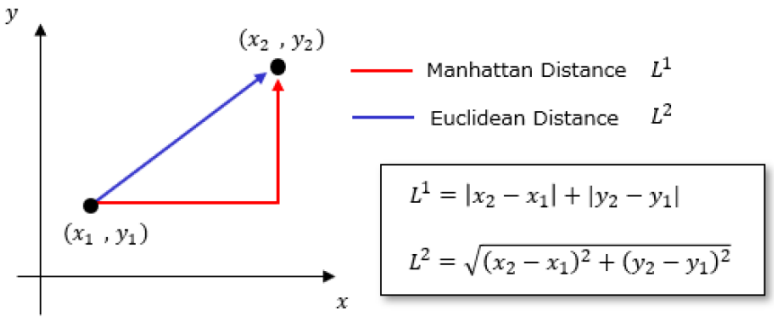

Figure 1 การหาระยะห่างระหว่าง Manhattan กับ Euclidean

2.2 ประเภทของการจัดกลุ่ม

ประเภทของการจัดกลุ่มที่นำมาใช้ในการทำรายงานประกอบไปด้วยดังนี้

1. **K-Means**

K-Means เป็นวิธีการแบ่งกลุ่มข้อมูล (clustering) ที่ใช้ใน Machine Learning และการประมวลผลข้อมูล เพื่อแบ่งข้อมูลให้อยู่ในกลุ่มที่มีคุณสมบัติคล้ายกัน ขั้นตอนหลัก K-Means ใน R ประกอบด้วยขั้นตอนดังนี้:

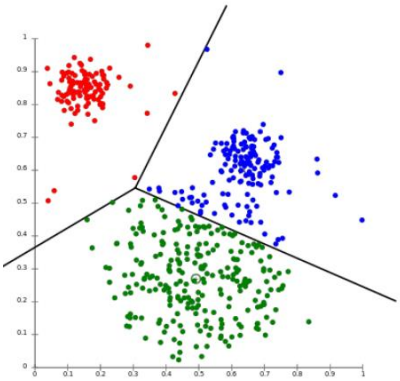

Figure 2 ตัวอย่าง K-Means

1. นำข้อมูลเข้า
2. กำหนด K หรือกลุ่มสำหรับการจัดกลุ่มด้วยวิธี Elbow Method ด้วยการหา kmeans$tot.withinss ผ่านการใช้ Looping โดยเงื่อนไขในการในการคัดเลือก K คือเมื่อค่า SSE ลดลงอย่างรวดเร็ว และหลังจากนั้นจึงค่อยๆ ลดลง
3. ใช้คำสั่ง kmeans(data,center = k) เพื่อทำการจัดกลุ่ม
4. ทำการ plot ลงกราฟ

2. Hierarchical Clustering

Hierarchical Clustering เป็นวิธีการจัดกลุ่มข้อมูลที่ช่วยให้ข้อมูลสามารถจัดเรียงเป็นโครงสร้างลำดับเฉลี่ยดั้งเดิมและแยกย่อยออกเป็นกลุ่มระดับที่ลึกลงไปได้ขั้นตอนหลัก Hierarchical Clustering ใน R ประกอบด้วยขั้นตอนดังนี้:

1. นำข้อมูลเข้า (ข้อมูลนั้นจะต้องไม่มากจนเกินไป)
2. ทำการหาระยะห่าง (Distance) ด้วย dist(data,method = ‘distance’)
3. ทำการ Heirarchical ด้วยคำสั่ง hclust(Distance)
4. plot ด้วย Dendrogram
5. ตัดสินใจกลุ่มที่ต้องการ และทำการตัด

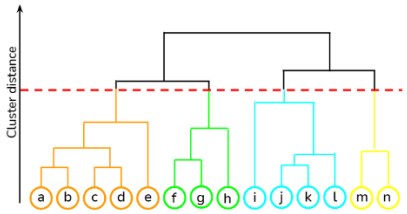

Figure 3 ตัวอย่าง Hierarchical

3. DBScan

เป็นวิธีการแบ่งกลุ่มข้อมูล (clustering) ที่ใช้ใน Machine Learning และการประมวลผลข้อมูลเพื่อค้นหากลุ่มข้อมูลที่มีความหนาแน่นสูงและความยากจะแยกจากกลุ่มอื่น ๆ โดยสนใจความหนาแน่นของข้อมูลแทนการกำหนดจำนวนของกลุ่มล่วงหน้า และสามารถค้นหากลุ่มข้อมูลที่มีรูปร่างและขนาดที่แตกต่างกันได้ดีขั้นตอนหลัก DBScan ใน R ประกอบด้วยขั้นตอนดังนี้:

1. นำข้อมูลเข้า
2. กำหนดการหาระยะห่างด้วย proxy::dist(data,method =’distance’)
3. กำหนด parameter 2 ตัวได้แก่ Eps และ Minpts
4. ใช้คำสั่ง dbscan เพื่อทำการสร้างกลุ่ม
5. จากนั้น plot dbscan

Figure 4 ตัวอย่าง DBScan

3. การทดสอบใน R

ในการทดสอบ R ประกอบไปด้วยดังนี้

1. Data Preparation
2. PCA
3. K-Means ระหว่างผ่าน PCA และยังไม่ผ่าน PCA
4. Hierarchical Clustering ระหว่างผ่าน PCA และยังไม่ผ่าน PCA
5. DBScan ระหว่างผ่าน PCA และยังไม่ผ่าน PCA

ทั้งนี้ในการเปรียบเทียบจะมีการเปรียบเทียบในทุกๆการจัดกลุ่มและการใช้การหาระยะทางที่แตกต่างกันทั้งหมด 6 คู่ได้แก่

1. K-Means ด้วยวิธีการ Euclidean Distance
2. K-Means ด้วยวิธีการ Manhattan Distance
3. Hierarchical Clustering ด้วยวิธีการ Euclidean Distance
4. Hierarchical Clustering ด้วยวิธีการ Manhattan Distance
5. DBScan ด้วยวิธีการ Euclidean Distance
6. DBScan ด้วยวิธีการ Manhattan Distance

ข้อมูลที่นำมาใช้ในการทดสอบครั้งนี้คือ Unsupervised Learning on Country Data จาก Kaggle ที่เป็นชุดข้อมูลสาธารณะสำหรับการศึกษาในครั้งนี้

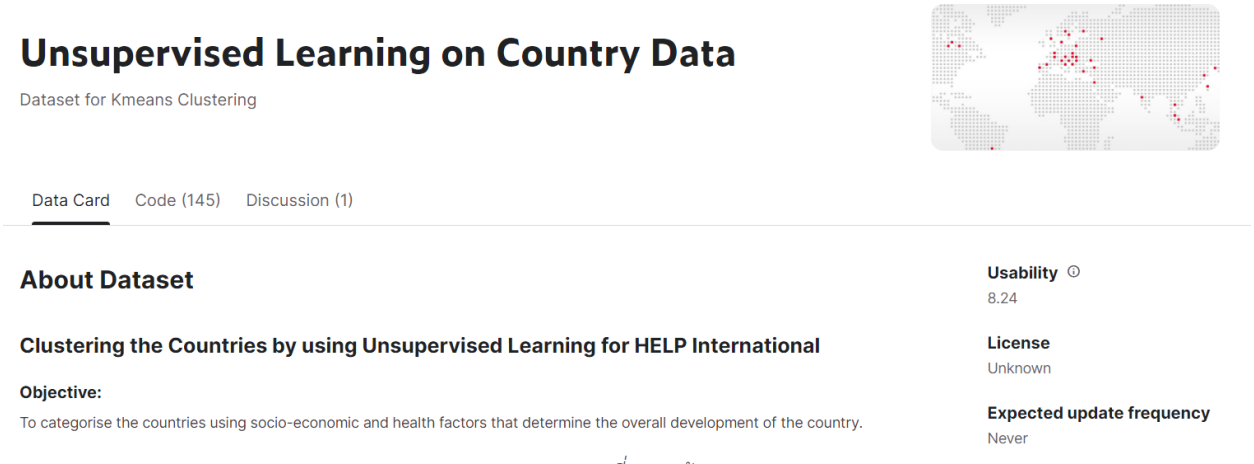

Figure 5 ที่มาของข้อมูล

4.1 Data Preparation

เมื่อเริ่มใช้งานเริ่มด้วยเปิดใช้งาน library แต่ละตัวประกอบด้วย

library(DataExplorer): โหลดแพ็กเกจ DataExplorer ซึ่งใช้สำหรับการสำรวจข้อมูล (data exploration) และการสร้างกราฟและสถิติพื้นฐานเพื่อทำความเข้าใจข้อมูล.

library(ClusterR): โหลดแพ็กเกจ ClusterR ซึ่งเป็นเครื่องมือสำหรับการวิเคราะห์และการจัดกลุ่มข้อมูล (clustering) โดยใช้วิธีต่าง ๆ เช่น K-means หรือ Hierarchical clustering.

library(cluster): โหลดแพ็กเกจ cluster ซึ่งมีฟังก์ชันที่เกี่ยวกับการทำ clustering และการวิเคราะห์กลุ่มข้อมูล.

library(ggfortify): โหลดแพ็กเกจ ggfortify ซึ่งช่วยในการทำการวิเคราะห์ข้อมูลและกราฟสำหรับข้อมูลที่มีโมเดลสถิติ, เช่น PCA (Principal Component Analysis) หรือ Clustering.

library(stats): โหลดแพ็กเกจ stats ซึ่งเป็นแพ็กเกจพื้นฐานของ R และมีฟังก์ชันสถิติพื้นฐาน.

library(fpc): โหลดแพ็กเกจ fpc ซึ่งใช้สำหรับการคำนวณค่า Fuzzy Partition Coefficient (FPC) ในการวิเคราะห์ข้อมูลที่มีการจัดกลุ่ม.

library(factoextra): โหลดแพ็กเกจ factoextra ซึ่งช่วยในการสร้างกราฟและผนวกข้อมูลหลายอย่างสำหรับการวิเคราะห์ข้อมูลหลายมิติ เช่น PCA.

library(corrplot): โหลดแพ็กเกจ corrplot ซึ่งใช้สำหรับการสร้างกราฟและการแสดงความสัมพันธ์ระหว่างตัวแปรในข้อมูลของคุณ.

```
library(DataExplorer)
library(ClusterR)
library(cluster)
library(ggfortify)
library(stats)
library(fpc)
library(factoextra)
library(corrplot)
```

Figure 6 เรียกใช้ package ด้วย library

หลังจากเรียกใช้งาน Library แล้วจะทำการเรียกข้อมูลเข้าสู่ R ด้วยวิธีการสร้าง path เข้าสู่ตัวแปร file และทำการอ่านค่าด้วยการ read.csv เก็บตัวแปรไว้ใน data

```
##import##
file = 'C:/Users/User/Documents/data/Country-data.csv'
data = read.csv(file)
data
```

country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
Afghanistan	90.2	10.000	7.58	44.9000	1610	9.440	56.2	5.82	553
Albania	16.6	28.000	6.55	48.6000	9930	4.490	76.3	1.65	4090
Algeria	27.3	38.400	4.17	31.4000	12900	16.100	76.5	2.89	4460
Angola	119.0	62.300	2.85	42.9000	5900	22.400	60.1	6.16	3530
Antigua and Barbuda	10.3	45.500	6.03	58.9000	19100	1.440	76.8	2.13	12200
Argentina	14.5	18.900	8.10	16.0000	18700	20.900	75.8	2.37	10300
Armenia	18.1	20.800	4.40	45.3000	6700	7.770	73.3	1.69	3220
Australia	4.8	19.800	8.73	20.9000	41400	1.160	82.0	1.93	51900
Austria	4.3	51.300	11.00	47.8000	43200	0.873	80.5	1.44	46900
Azerbaijan	39.2	54.300	5.88	20.7000	16000	13.800	69.1	1.92	5840
Bahamas	13.8	35.000	7.89	43.7000	22900	-0.393	73.8	1.86	28000
Bahrain	8.6	69.500	4.97	50.9000	41100	7.440	76.0	2.16	20700
Bangladesh	49.4	16.000	3.52	21.8000	2440	7.140	70.4	2.33	758
Barbados	14.2	39.500	7.97	48.7000	15300	0.321	76.7	1.78	16000
Belarus	5.5	51.400	5.61	64.5000	16200	15.100	70.4	1.49	6030
Belgium	4.5	76.400	10.70	74.7000	41100	1.880	80.0	1.86	44400
Belize	18.8	58.200	5.20	57.5000	7880	1.140	71.4	2.71	4340

Figure 7 วิธีการ import data และดูข้อมูลภายใน

หลังจากนั้นจะทำการตรวจสอบข้อมูลเบื้องต้นว่าเป็นยังไงด้วยคำสั่ง introduce

```
introduce(data)
```

```
  rows columns discrete_columns continuous_columns all_missing_columns total_missing_values
1  167      10                1                  9                   0                    0
  complete_rows total_observations memory_usage
1           167               1670        25112
```

Figure 8 ตรวจสอบข้อมูลเบื้องต้น

ผลลัพธ์ที่ได้พบว่ามีข้อมูลอยู่ 167 แถวและ 10 คอลัมน์แต่ละคอลัมน์ประกอบด้วยตัวแปรไม่ต่อเนื่อง 1 ตัวและตัวแปรต่อเนื่อง 9 ตัวทั้งนี้ยังคงพบอีกว่าข้อมูลของเราไม่มีสูญหายจึงสามารถใช้งานได้เลย

เราจึงเลือกคอลัมน์สำหรับการใช้งานโดยไม่เอาตัวแปรไม่ต่อเนื่องทั้งด้วยวิธีการเลือก data จากคอลัมน์ที่ 2 ถึงคอมลัมน์ที่ 10 ได้ออกมาดังนี้

```
new_df <- data[,2:10]
new_df
```

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
1	90.2	10.000	7.58	44.9000	1610	9.440	56.2	5.82	553
2	16.6	28.000	6.55	48.6000	9930	4.490	76.3	1.65	4090
3	27.3	38.400	4.17	31.4000	12900	16.100	76.5	2.89	4460
4	119.0	62.300	2.85	42.9000	5900	22.400	60.1	6.16	3530
5	10.3	45.500	6.03	58.9000	19100	1.440	76.8	2.13	12200
6	14.5	18.900	8.10	16.0000	18700	20.900	75.8	2.37	10300
7	18.1	20.800	4.40	45.3000	6700	7.770	73.3	1.69	3220
8	4.8	19.800	8.73	20.9000	41400	1.160	82.0	1.93	51900
9	4.3	51.300	11.00	47.8000	43200	0.873	80.5	1.44	46900
10	39.2	54.300	5.88	20.7000	16000	13.800	69.1	1.92	5840
11	13.8	35.000	7.89	43.7000	22900	-0.393	73.8	1.86	28000
12	8.6	69.500	4.97	50.9000	41100	7.440	76.0	2.16	20700
13	49.4	16.000	3.52	21.8000	2440	7.140	70.4	2.33	758
14	14.2	39.500	7.97	48.7000	15300	0.321	76.7	1.78	16000
15	5.5	51.400	5.61	64.5000	16200	15.100	70.4	1.49	6030
16	4.5	76.400	10.70	74.7000	41100	1.880	80.0	1.86	44400
17	18.8	58.200	5.20	57.5000	7880	1.140	71.4	2.71	4340
18	111.0	23.800	4.10	37.2000	1820	0.885	61.8	5.36	758

Figure 9 การสร้างตารางใหม่สำหรับการวิเคราะห์

จากนั้นนำข้อมูลทำให้เป็นมาตรฐานด้วยวิธีการ standardize data ด้วยคำสั่ง scale ของข้อมูลที่เราใช้

```
standardized_data <- scale(new_df)
standardized_data
```

เมื่อทำการปรับข้อมูลแล้วจะได้ดังนี้

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
1	1.28765971	-1.1348666486	0.27825140	-0.08220771	-0.805821873	0.156864451	-1.614237166	1.897176463	-0.677143083
2	-0.53733286	-0.4782201668	-0.09672528	0.07062429	-0.374243349	-0.311410892	0.645923798	-0.857394179	-0.484167091
3	-0.27201464	-0.0988244217	-0.96317624	-0.63983800	-0.220182266	0.786907640	0.668412962	-0.038289240	-0.463980176
4	2.00178723	0.7730561847	-1.44372888	-0.16481961	-0.583289197	1.382894441	-1.175698472	2.121769753	-0.514720259
5	-0.69354825	0.1601861350	-0.28603389	0.49607554	0.101426731	-0.599944185	0.702146708	-0.540321300	-0.041691745
6	-0.58940465	-0.8101914437	0.46756001	-1.27594958	0.080677763	1.240992822	0.589700889	-0.381784860	-0.145354280
7	-0.50013871	-0.7408787595	-0.87944359	-0.06568534	-0.541791262	-0.001119352	0.308586341	-0.830971439	-0.531633620
8	-0.82992677	-0.7773591196	0.69691468	-1.07355044	1.258181668	-0.626432487	1.286864967	-0.672434999	2.124309640
9	-0.84232482	0.3717722236	1.52331959	0.03757953	1.351552022	-0.653582997	1.118196239	-0.996113564	1.851513496
10	0.02305888	0.4812133039	-0.34064215	-1.08181163	-0.059377768	0.569325158	-0.163686100	-0.679040684	-0.388688441
11	-0.60676192	-0.2228576461	0.39110846	-0.13177485	0.298541922	-0.773347964	0.364809250	-0.718674794	0.820344071
12	-0.73570161	1.0357147775	-0.67193222	0.16562797	1.242619943	-0.032337708	0.612190052	-0.520504245	0.422061700
13	0.27597905	-0.9159844880	-1.19981201	-1.03637509	-0.762767766	-0.060718032	-0.017506535	-0.408207600	-0.665958441
14	-0.59684348	-0.0586960256	0.42023286	0.07475488	-0.095688461	-0.705802793	0.690902126	-0.771520274	0.165633325
15	-0.81256951	0.3754202596	-0.43893700	0.72738885	-0.049003284	0.692306561	-0.017506535	-0.963085139	-0.378322187
16	-0.83736560	1.2874292622	1.41410308	1.14870951	1.242619943	-0.558319710	1.061973329	-0.718674794	1.715115424
17	-0.48278145	0.6234867083	-0.58819957	0.43824722	-0.480581808	-0.628324509	0.094939284	-0.157191570	-0.470527284

Figure 10 การทำข้อมูลให้เป็นมาตรฐาน

4.2 PCA

เริ่มจากใช้คำสั่ง prcomp ทำการหาค่า PCA สำหรับการคัดเลือกตัวแปรและเก็บไว้ใน res.pca

```
res.pca = prcomp(standardized_data, scale = TRUE)
print(summary(res.pca))
```

Figure 11 การคำนวณ PCA

ผลลัพธ์ที่ได้ค่าดังรูปภาพ ทั้งนี้เกณฑ์ในการตัดสินใจสำหรับการคัดเลือกตัวแปรคือต้องมี cumulative proportion อย่างน้อย 80% ดังนั้นจากภาพพบว่าจะคัดเลือกตัวแปรเข้าทั้งหมดเพียง 4 ตัวแปรได้แก่ PC1,PC2,PC3 และ PC4

```
Importance of components:
                          PC1    PC2    PC3    PC4    PC5     PC6    PC7     PC8     PC9
Standard deviation     2.0336 1.2435 1.0818 0.9974 0.8128 0.47284 0.3368 0.29718 0.25860
Proportion of Variance 0.4595 0.1718 0.1300 0.1105 0.0734 0.02484 0.0126 0.00981 0.00743
Cumulative Proportion  0.4595 0.6313 0.7614 0.8719 0.9453 0.97015 0.9828 0.99257 1.00000
> fviz_eig(res.pca,addlabels = TRUE)
>
```

Figure 12 ผลลัพธ์ในการคำนวณ PCA

ทั้งนี้ได้ใช้คำสั่ง fviz_eig เพื่อทำการ plot ค่า Eigenvalue เพื่อดูน้ำหนักของแต่ละตัวแปร ซึ่งพบว่าตัวแปร 4 ตัวแรกได้แก่ PC1 คือ 46% PC2 คือ17.2% PC3 คือ 13% และ PC4 คือ 11.1%

```
fviz_eig(res.pca,addlabels = TRUE)
```

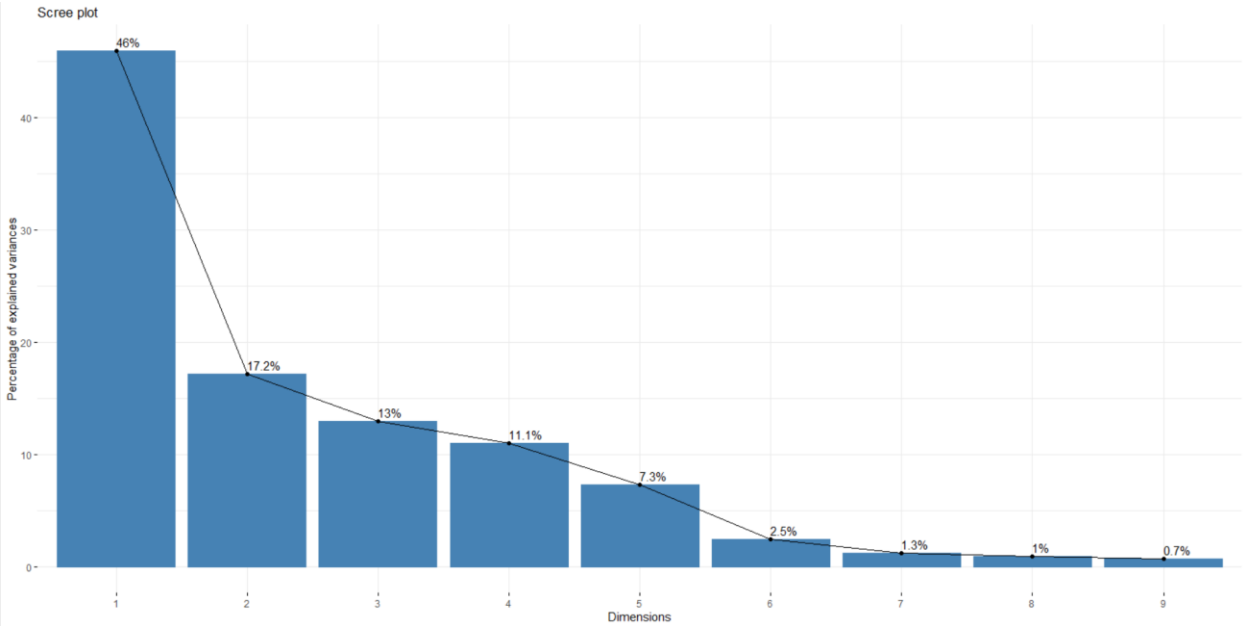

Figure 13 การ plot Eigenvalue

หลังจากนั้นทำการเก็บค่า PCA สำหรับการหา correlation circle ด้วยตัวแปร var

```
var = get_pca_var(res.pca)
```

Figure 14 เก็บค่า PCA ไว้ที่ตัวแปร var

จากนั้นใช้คำสั่ง fviz_pca_var ทำการ plot เพื่อดูความสัมพันธ์ของแต่ละตัวแปร

```
fviz_pca_var(res.pca, col.var = "black")
```

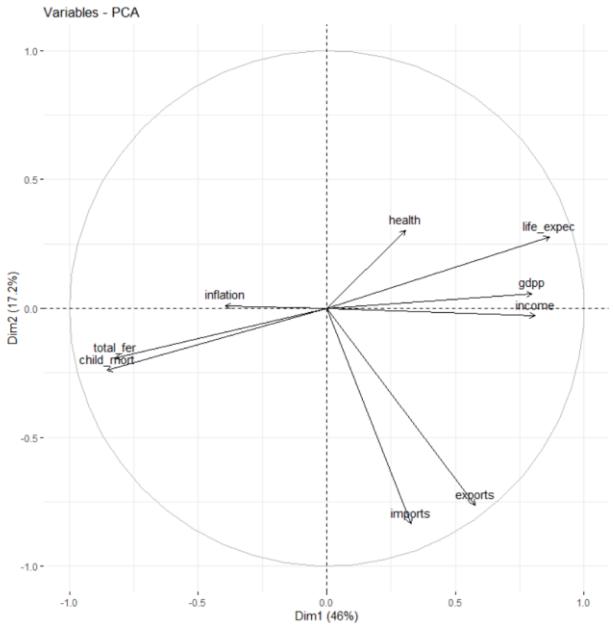

Figure 15 Correlation Circle แบบสีดำ

จากผลที่ได้โดยวิเคราะห์เพียง 4 ตัวแปรหลักพบว่า

1. อัตราการตายของเด็กที่ต่ำกว่า 5 ปี (Child_mort) พบว่าถ้าหากมีอัตราการตายที่เยอะ คุณภาพการพัฒนาในประเทศก็จะต่ำ
2. อัตราการส่งออกสินค้า (exports) พบว่าถ้ามีอัตราการส่งออกสูงจะทำให้ คุณภาพพัฒนาประเทศอยู่ในระดับที่ดี
3. การใช้จ่ายด้านสุขภาพ (health) พบว่าถ้ามีอัตราการใช้จ่ายด้านสุขภาพที่มาก คุณภาพการพัฒนาประเทศก็จะสูง
4. อัตราการนำเข้าสินค้า (import) พบว่าถ้ามีอัตราการนำเข้าสูงจะทำให้ คุณภาพพัฒนาประเทศอยู่ในระดับที่ดี

จากนั้นทำการค่าความสัมพันธ์ correlation ว่ามีความสัมพันธ์ระดับใดโดยที่หากมีสีเข้มจะมีความสัมพันธ์กันมาก และหากมีสีอ่อนจนถึงขาวหมายถึงไม่มีความสัมพันธ์เลย

```
corrplot(var$cos2, is.corr = FALSE)
```

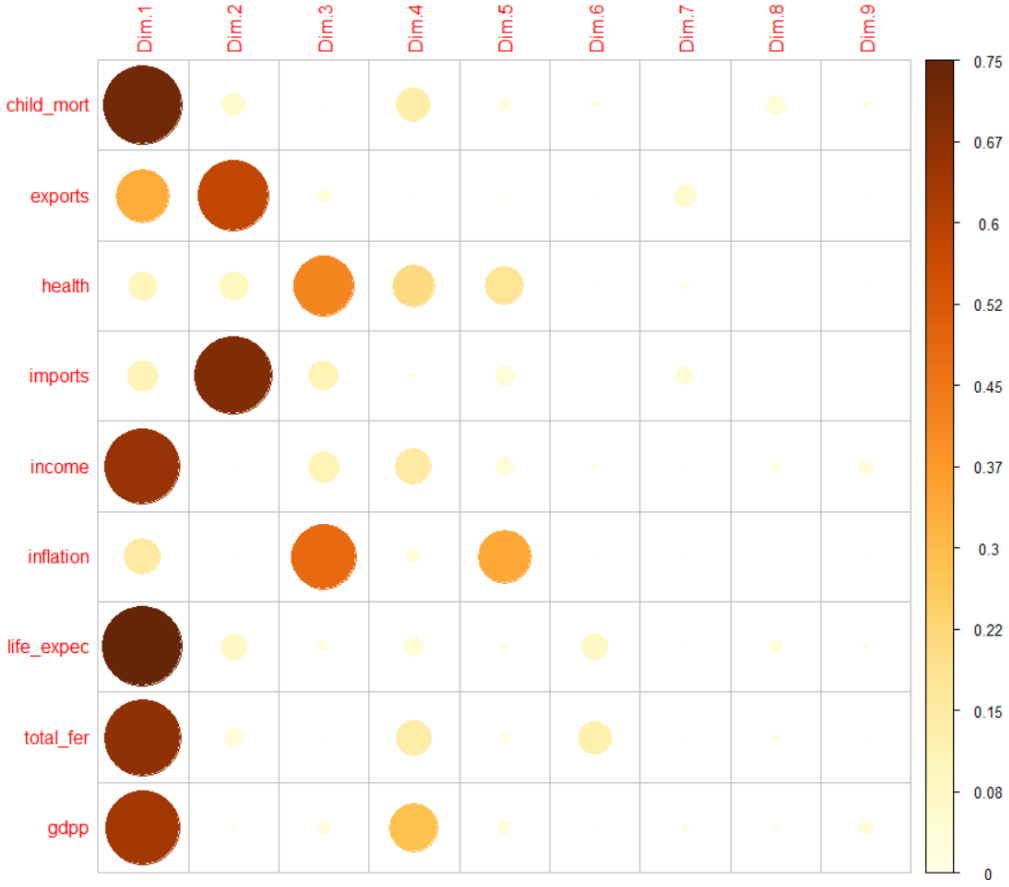

Figure 16 ระดับความสัมพันธ์ของแต่ละข้อมูล

ตัวอย่างโดยการใช้ Dimension 1 พบว่า

1. มีความสัมพันธ์กันมากกับตัวแปร อัตราการตายของเด็กที่ต่ำกว่า 5 ปี(child_mort), รายได้ต่อคน (income), อายุเฉลี่ย (life_expec),จำนวนลูกของแต่ละคน (total_fer) และค่า GDP เมื่อเทียบกับแต่ละหัวของแต่ละประเทศ (gdpp)
2. มีความสัมพันธ์ระดับปานกลางกับตัวแปร ส่งออกสินค้า(export)
3. มีความสัมพันธ์น้อยกับตัวแปร การใช้จ่ายสุดภาพ (health), การนำเข้าสินค้า(import) และอัตรา GDP (inflation)

จากนั้นใช้คำสั่ง fviz_pca_var ทำการ plot เพื่อดูความสัมพันธ์ของแต่ละตัวแปรและดูว่ามีความสัมพันธ์มากน้อยขนาดไหน

```
fviz_pca_var(res.pca, col.var = "cos2",
             gradient.cols = c("#00AFBB","#E7B880","#FC4E07"))
```

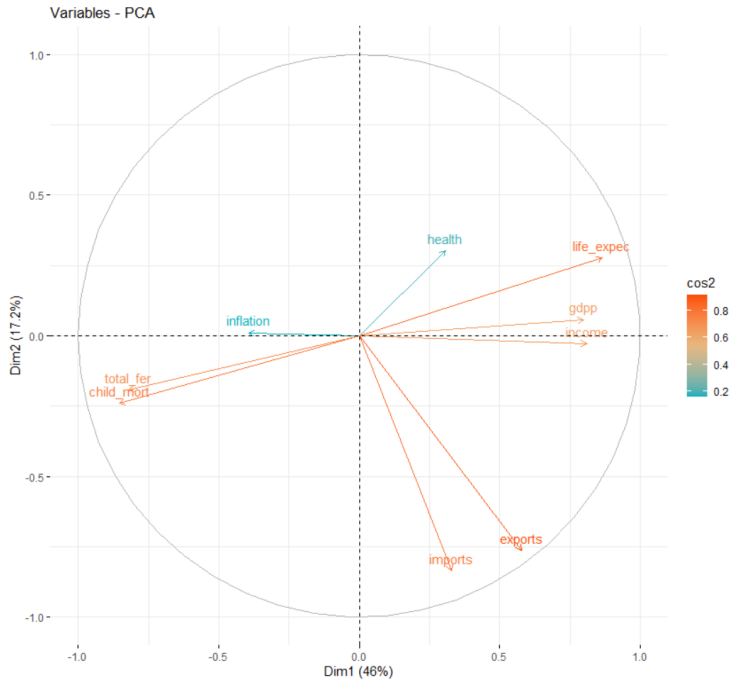

Figure 17 Correlation circle แบบสีตามระดับความสัมพันธ์

จากผลที่ได้โดยวิเคราะห์เพียง 4 ตัวแปรหลักพบว่า

1. อัตราการตายของเด็กที่ต่ำกว่า 5 ปี (Child_mort) พบว่าถ้าหากมีอัตราการตายที่เยอะ คุณภาพการพัฒนาในประเทศก็จะต่ำและมีความสัมพันธ์ที่สูง
2. อัตราการส่งออกสินค้า (exports) พบว่าถ้ามีอัตราการส่งออกสูงจะทำให้ คุณภาพพัฒนาประเทศอยู่ในระดับที่ดีและมีความสัมพันธ์ที่สูง
3. การใช้จ่ายด้านสุขภาพ (health) พบว่าถ้ามีอัตราการใช้จ่ายด้านสุขภาพที่มาก คุณภาพการพัฒนาประเทศก็จะสูงและมีความสัมพันธ์ที่น้อย
4. อัตราการนำเข้าสินค้า (import) พบว่าถ้ามีอัตราการนำเข้าสูงจะทำให้ คุณภาพพัฒนาประเทศอยู่ในระดับที่ดีและมีความสัมพันธ์ที่สูง

จากนั้นใช้คำสั่ง fviz_pca_var ทำการ plot เพื่อดูความสัมพันธ์ของแต่ละประเทศและดูว่ามีความสัมพันธ์มากน้อยขนาดไหน

```
ind <- get_pca_ind(res.pca)
fviz_pca_ind(res.pca, col.ind = "cos2",
             gradient.cols = c("#00AFBB","#E7B880","#FC4E07"))
```

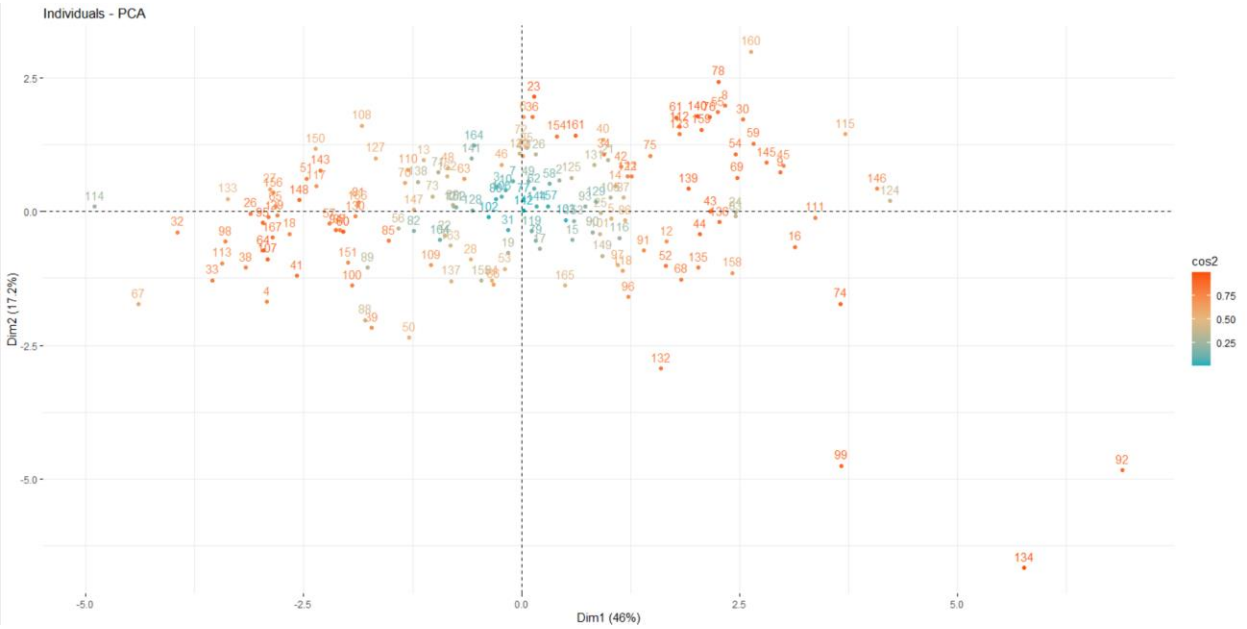

Figure 18 plot เป็น Individual ของแต่ละประเทศ

โดยผลที่ได้คือข้อมูลแต่ละประเทศจะถูก plot โดยแสดงเป็นตัวเลขและแสดงความสัมพันธ์หากมีสีส้มแสดงว่ามีความที่มาก และหากมีสีฟ้าแสดงความสัมพันธ์ที่น้อย

ทั้งนี้จากการทำ PCA จะกำหนดให้ตัวแปรที่ถูกคัดเลือกมีทั้งหมด 4 ตัวแปรเป็นตัวแปรสำหรับการทำกรณีที่คัดเลือกเรียบร้อยแล้วและกำหนดให้

Pca_components เป็นกรณีที่ดูเลือกผ่านกระบวนการ PCA เรียบร้อย

Standardized data เป็นกรณีที่ยังไม่ทำการผ่าน PCA.

```
pca_components <- res.pca$x[, 1:4]
pca_components # with PCA
standardized_data # without PCA
```

Figure 19 กำหนดข้อมูลสำหรับการนำไปใช้

4.3 K-Means ระหว่างผ่าน PCA และยังไม่ผ่าน PCA

4.3.1 K-Means ด้วยวิธีการ Euclidean Distance

ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA

ขั้นตอนแรกสำหรับ K-mean คือการเลือกจำนวนคลัสเตอร์ (K) โดยที่ใช้วิธี Elbow (Elbow method) เพื่อค้นหาค่า K ที่ดีที่สุด ดังนั้นจึงใช้การ Looping เพื่อค้นหาตัวเลือกที่ดีที่สุดโดยเริ่มจากตัวที่ 1 ถึงตัวที่ 20 และ การจัดเก็บตัวแปรในรูปแบบ sse และพล็อตจากตัวที่ 1 ถึงตัวที่ 20 เปรียบเทียบกับค่า sse ของแต่ละรายการ

```
sse <-numeric(20)
for (k in 1:20) {
  kmeans_model <- kmeans(standardized_data, centers = k)
  sse[k] <- kmeans_model$tot.withinss
}

ggplot(data.frame(K = 1:20, SSE = sse), aes(x = K, y = SSE)) +
  geom_line() +
  geom_point() +
  labs(x = "จำนวนกลุ่ม (K)", y = "SSE") +
  theme_minimal()
```

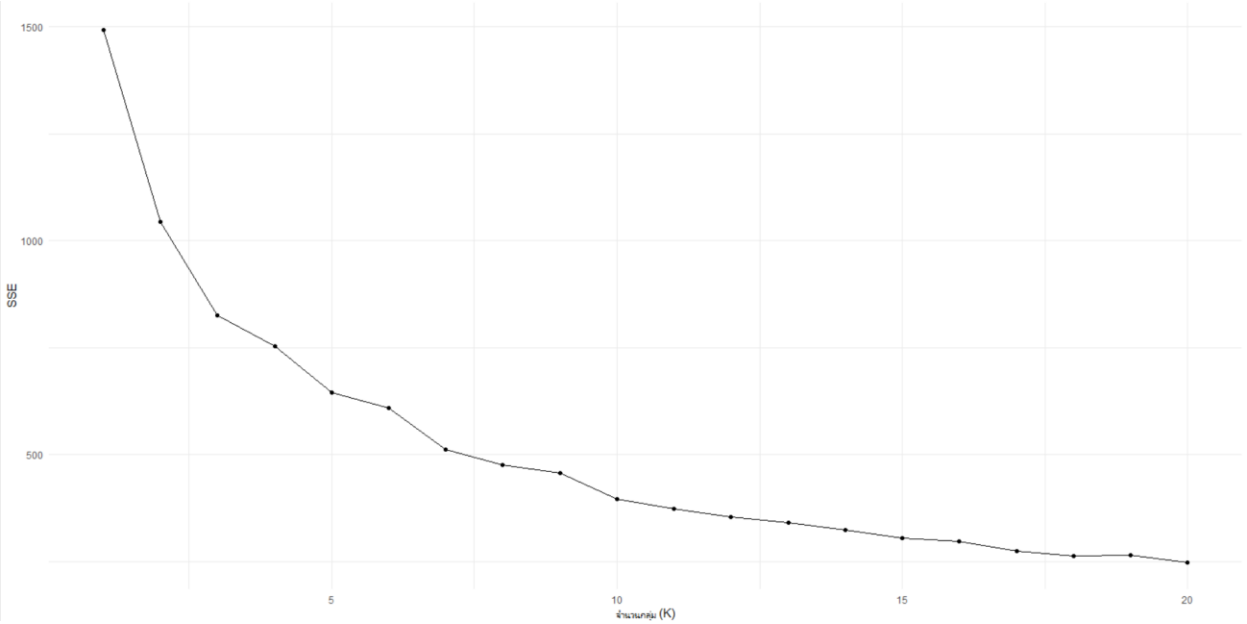

Figure 20 การคัดเลือก K ด้วยวิธี Elbow Method

ในการตัดสินใจ ให้หาค่า K ที่ทำให้ SSE ลดลงอย่างรวดเร็ว และหลังจากนั้นจึงค่อยๆ ลดลง เราเลยเลือก k = 8 เพื่อทำคลัสเตอร์ในครั้งนี้

ต่อมาคือการทำ K-means ด้วยการนำข้อมูลเป็นข้อมูลที่ยังไม่ผ่านกระบวนการ PCA และกำหนด k ด้วย center ในที่นี้ k=8

```
kmeans_result <- kmeans(standardized_data,center = 8)
kmeans_result
```

```
K-means clustering with 8 clusters of sizes 1, 14, 46, 30, 22, 6, 32, 16

Clustering vector:
  [1] 7 4 3 7 4 3 3 5 5 3 4 8 3 4 3 5 3 7 3 3 4 2 4 6 4 7 2 3 7 5 3 7 7 4 3 4 7 7 7 4 7 4 8
 [44] 8 5 3 4 3 4 7 7 8 3 5 5 7 7 4 5 7 5 4 3 7 2 3 2 8 5 3 3 3 3 8 5 5 3 5 4 3 7 2 6 3 3 4
 [87] 4 2 2 3 8 6 4 7 7 8 8 7 8 7 4 2 4 3 4 3 7 3 2 3 5 5 7 1 5 3 7 8 3 3 3 4 5 6 4 3 2 3 3
[130] 7 4 8 2 6 8 8 2 2 4 5 3 3 7 3 5 5 3 7 8 7 2 3 4 4 3 7 4 6 5 5 4 3 3 3 8 7 7

Within cluster sum of squares by cluster:
[1]   0.00000  66.20912 121.00813  40.76736  46.96944  85.92264  88.73087  48.77497
 (between_SS / total_SS =  66.6 %)
```

Figure 21 ผลลัพธ์จากการทำ K-means

ผลลัพธ์ที่ได้พบว่า K-means ที่ถูกแบ่งออกเป็น 8 กลุ่มประกอบด้วยสมาชิกดังนี้ 1,14,46,30,22,6,32 และ 16 ทั้งนี้ผลที่ได้มาพบว่ามีค่า Sum of square อยู่ที่ 66.6%

จากนั้นทำการ autoplot เพื่อทำการ cluster โดยที่เทียบ PC1 กับ PC2 พบว่าได้ข้อมูลออกมาเป็นดังนี้

```
autoplot(kmeans_result,standardized_data,frame=TRUE)
```

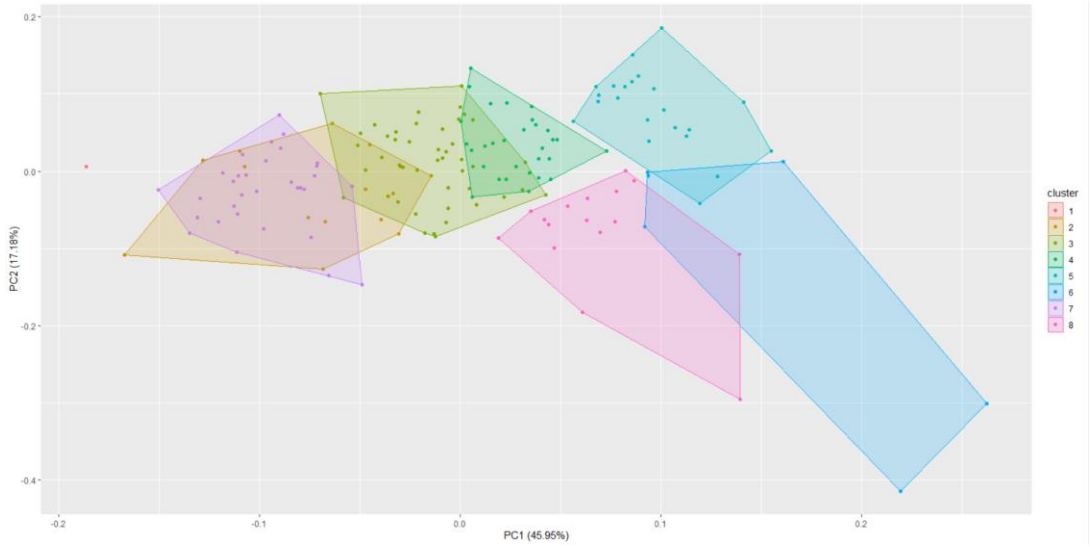

Figure 22 กราฟการจัดกลุ่มด้วย K-means

สรุปทั้งหมดของการทำ K-means ด้วยวิธีการ Euclidean Distance ผ่านข้อมูลที่ยังไม่ผ่านกระบวนการ PCA พบว่า จากการทำวิธี elbow พบว่าจำนวนกลุ่มที่เหมาะสมคือ 8 และผลลัพธ์การจัดกลุ่มพบว่า 8 กลุ่ม ประกอบด้วยสมาชิกดังนี้ 1,14,46,30,22,6,32 และ 16 และมีค่า sum of square คือ 66.6%

ข้อมูลที่ยังผ่านกระบวนการ PCA แล้ว

ขั้นตอนแรกสำหรับ K-mean คือการเลือกจำนวนคลัสเตอร์ (K) โดยที่ใช้วิธี Elbow (Elbow method) เพื่อค้นหาค่า K ที่ดีที่สุด ดังนั้นจึงใช้การ Looping เพื่อค้นหาตัวเลือกที่ดีที่สุดโดยเริ่มจากตัวที่ 1 ถึงตัวที่ 20 และ การจัดเก็บตัวแปรในรูปแบบ sse และพล็อตจากตัวที่ 1 ถึงตัวที่ 20 เปรียบเทียบกับค่า sse ของแต่ละรายการ

```
sse <-numeric(20)
for (k in 1:20) {
  kmeans_model <- kmeans(pca_components, centers = k)
  sse[k] <- kmeans_model$tot.withinss
}

ggplot(data.frame(K = 1:20, SSE = sse), aes(x = K, y = SSE)) +
  geom_line() +
  geom_point() +
  labs(x = "จำนวนกลุ่ม (K)", y = "SSE") +
  theme_minimal()
```

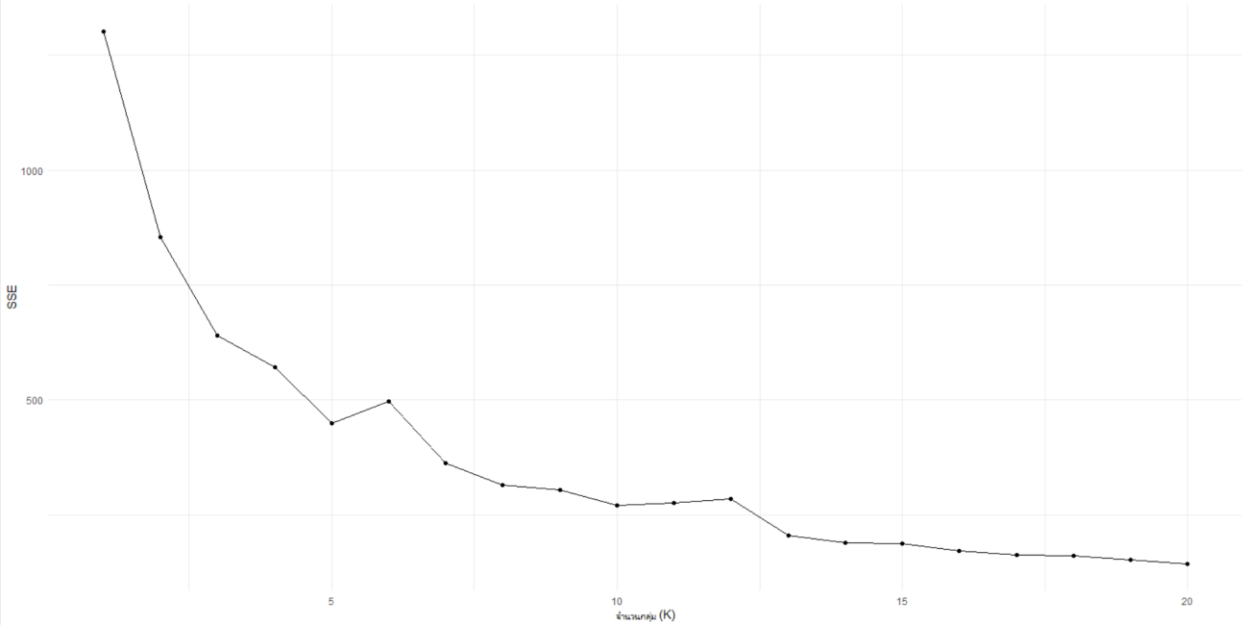

Figure 23 การคัดเลือก K ด้วยวิธี Elbow Method

ในการตัดสินใจ ให้หาค่า K ที่ทำให้ SSE ลดลงอย่างรวดเร็ว และหลังจากนั้นจึงค่อยๆ ลดลง เราเลยเลือก k = 8 เพื่อทำคลัสเตอร์ในครั้งนี้

ต่อมาคือการทำ K-means ด้วยการนำข้อมูลเป็นข้อมูลที่ผ่านกระบวนการ PCA แล้วและกำหนด k ด้วย center ในที่นี้ k=8

```
kmeans_result_pca <- kmeans(pca_components,center = 8)
kmeans_result_pca
```

```
K-means clustering with 8 clusters of sizes 43, 3, 25, 12, 22, 7, 28, 27

Clustering vector:
  [1] 5 1 7 4 1 7 7 8 8 7 1 1 7 1 1 8 1 5 3 3 1 3 7 8 1 5 5 3 5 8 3 5 5 7 7 7 3 5 4 1 5 1 1
 [44] 1 8 7 7 7 1 4 4 1 3 8 8 4 5 1 8 4 8 1 3 5 5 3 5 1 8 3 7 7 3 8 8 8 7 8 1 7 3 6 8 3 3 1
 [87] 1 6 6 7 1 2 1 3 5 1 1 5 2 4 1 6 1 4 1 3 5 7 3 7 8 8 5 4 8 7 4 1 1 7 3 1 8 8 1 7 6 3 7
[130] 3 1 1 5 2 1 1 6 3 1 8 7 1 4 1 8 8 3 5 1 5 6 3 1 7 3 5 1 8 8 8 7 7 3 7 1 4 5

Within cluster sum of squares by cluster:
[1] 60.45652 17.57130 27.81880 57.40151 26.90301 11.27856 46.01632 98.07230
 (between_SS / total_SS =  73.5 %)
```

Figure 24 ผลลัพธ์จากการทำ K-means

ผลลัพธ์ที่ได้พบว่า K-means ที่ถูกแบ่งออกเป็น 8 กลุ่มประกอบด้วยสมาชิกดังนี้ 43,3,25,12,22,7,28 และ 27ทั้งนี้ผลที่ได้มาพบว่ามีค่า Sum of square อยู่ที่ 73.5%

จากนั้นทำการ autoplot เพื่อทำการ cluster โดยที่เทียบ PC1 กับ PC2 พบว่าได้ค่าดังนี้

```
autoplot(kmeans_result_pca,pca_components,frame=TRUE)
```

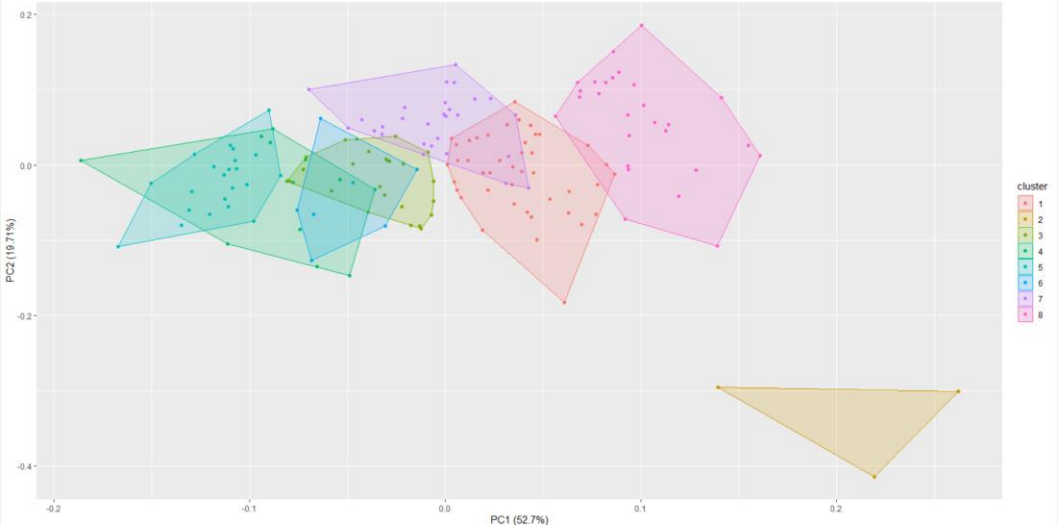

Figure 25 กราฟการจัดกลุ่มด้วย K-means

สรุปทั้งหมดของการทำ K-means ด้วยวิธีการ Euclidean Distance ผ่านข้อมูลที่ผ่านกระบวนการ PCA แล้วพบว่า จากการทำวิธี elbow พบว่าจำนวนกลุ่มที่เหมาะสมคือ 8 และผลลัพธ์การจัดกลุ่มพบว่า 8 กลุ่ม ประกอบด้วยสมาชิกดังนี้ 1,14,46,30,22,6,32 และ 16 และมีค่า sum of square คือ 73.5%

การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA กับข้อมูลที่ผ่านกระบวนการ PCA แล้วจากการทำ Clustering ด้วย K-Means ด้วยวิธีการ Euclidean Distance

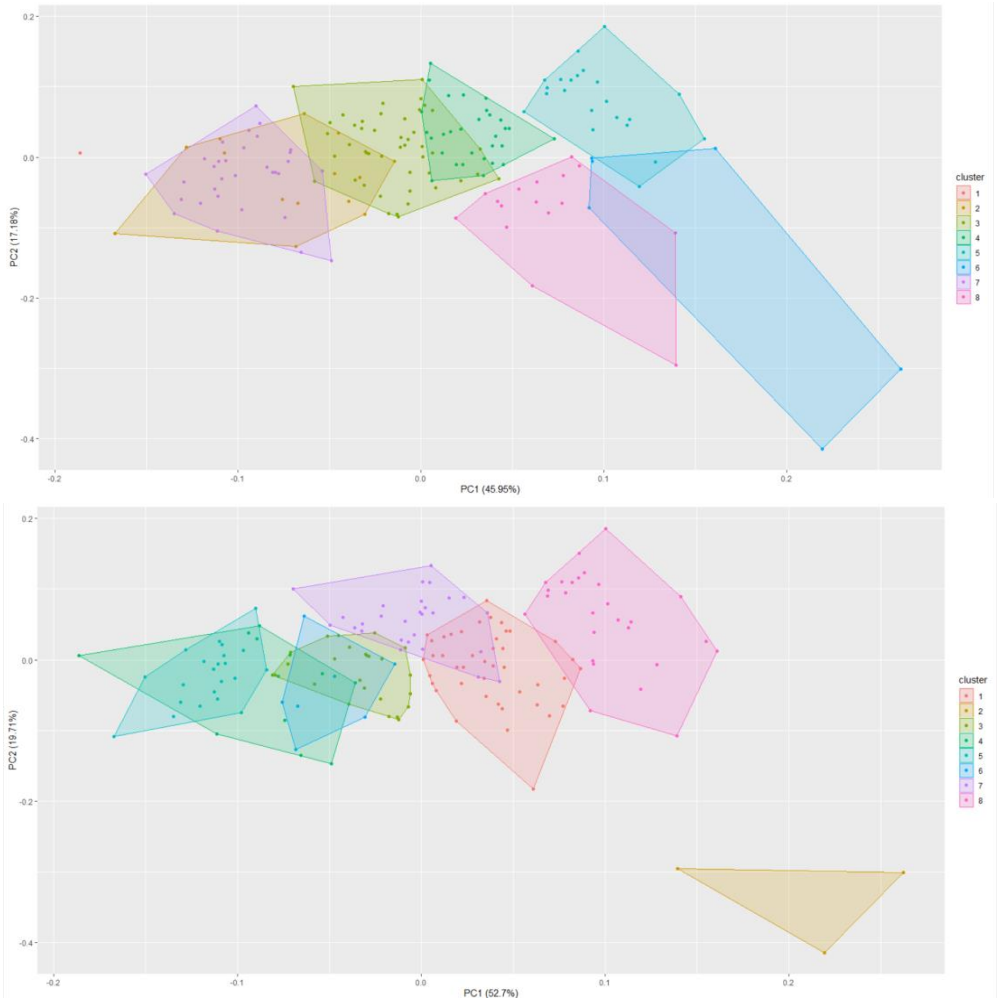

Figure 26 การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA (ด้านบน)กับข้อมูลที่ผ่าน PCA แล้ว(ด้านล่าง)

จากผลการเปรียบเทียบด้วย Visualization พบว่าข้อมูลทั้งกลุ่มนั้นแตกต่างกันในการจัดกลุ่มแต่ว่าจำนวนของกลุ่มที่ถูกเลือกนั้นมี 8 เท่ากันเหมือนเดิม ทั้งนี้ถ้าหากวัดที่ค่า sum of square จะทำให้สามารถพูดได้ว่ากรณีข้อมูลที่ผ่านกระบวนการ PCA แล้วนั้นดีกว่าด้วยค่า sum of square ที่มากกว่า

4.3.2 K-Means ด้วยวิธีการ Manhattan Distance

ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA

ขั้นตอนแรกสำหรับ K-mean คือการเลือกจำนวนคลัสเตอร์ (K) โดยที่ใช้วิธี Elbow (Elbow method) เพื่อค้นหาค่า K ที่ดีที่สุด ดังนั้นจึงใช้การ Looping เพื่อค้นหาตัวเลือกที่ดีที่สุดโดยเริ่มจากตัวที่ 1 ถึงตัวที่ 20 และ การจัดเก็บตัวแปรในรูปแบบ sse และพล็อตจากตัวที่ 1 ถึงตัวที่ 20 เปรียบเทียบกับค่า sse ของแต่ละรายการและกำหนด algorithm เป็น Lloyd สำหรับการทำ Manhattan distance

```
sse <-numeric(20)
for (k in 1:20) {
  kmeans_model <- kmeans(standardized_data, centers = k,algorithm = "Lloyd")
  sse[k] <- kmeans_model$tot.withinss
}

ggplot(data.frame(K = 1:20, SSE = sse), aes(x = K, y = SSE)) +
  geom_line() +
  geom_point() +
  labs(x = "จำนวนกลุ่ม (K)", y = "SSE") +
  theme_minimal()
```

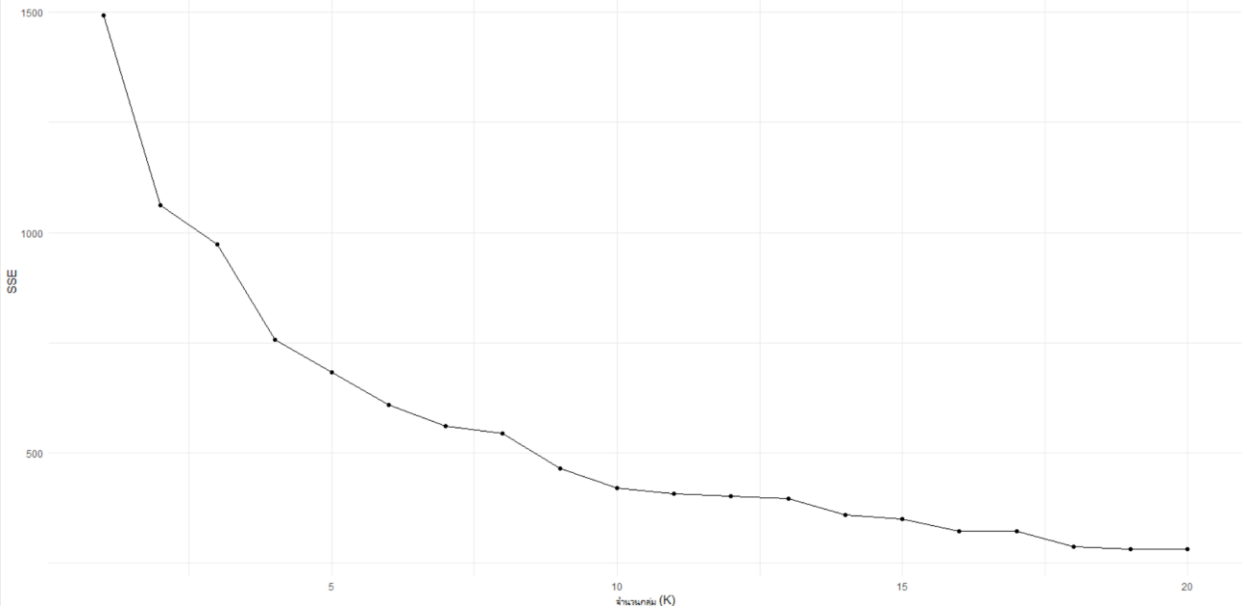

Figure 27 การคัดเลือก K ด้วยวิธี Elbow Method

ในการตัดสินใจ ให้หาค่า K ที่ทำให้ SSE ลดลงอย่างรวดเร็ว และหลังจากนั้นจึงค่อยๆ ลดลง เราเลยเลือก k = 10 เพื่อทำคลัสเตอร์ในครั้งนี้

ต่อมาคือการทำ K-means ด้วยการนำข้อมูลเป็นข้อมูลที่ยังไม่ผ่านกระบวนการ PCA กำหนด k ด้วย center ในที่นี้ k=10 และกำหนด algorithm เป็น Lloyd สำหรับการทำ Manhattan distance

```
kmeans_result_man <- kmeans(standardized_data,center = 10,algorithm = "Lloyd")
kmeans_result_man
```

```
K-means clustering with 10 clusters of sizes 25, 8, 22, 21, 6, 34, 15, 27, 8, 1

within cluster sum of squares by cluster:
 [1]  38.29021  31.31991  46.96944  59.97239  15.47046  61.74509  30.61631  51.29013
 [9] 115.75637   0.00000
 (between_SS / total_SS =  69.8 %)
```

Figure 28 ผลลัพธ์จากการทำ K-means

ผลลัพธ์ที่ได้พบว่า K-means ที่ถูกแบ่งออกเป็น 10 กลุ่มประกอบด้วยสมาชิกดังนี้25,8,22,21,6,34 ,15,27,8 และ 16 ทั้งนี้ผลที่ได้มาพบว่ามีค่า Sum of square อยู่ที่ 69.8%

จากนั้นทำการ autoplot เพื่อทำการ cluster โดยที่เทียบ PC1 กับ PC2 พบว่าได้ค่าดังนี้

```
autoplot(kmeans_result_man,standardized_data,frame=TRUE)
```

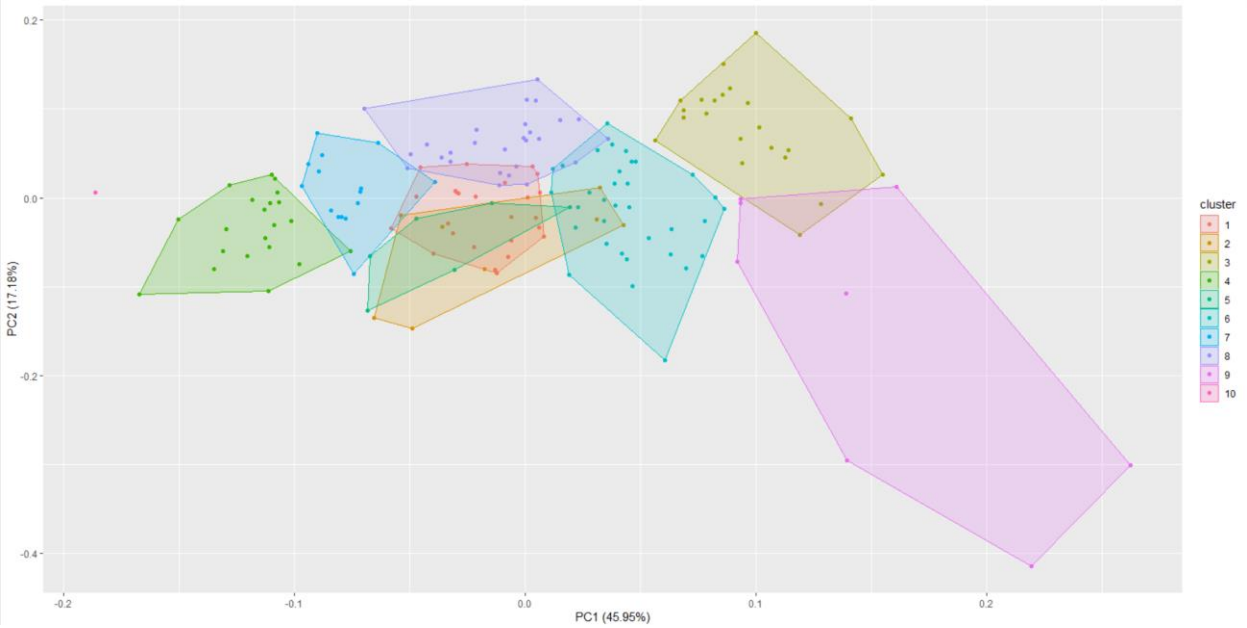

Figure 29 กราฟการจัดกลุ่มด้วย K-means

สรุปทั้งหมดของการทำ K-means ด้วยวิธีการ Manhattan Distance ผ่านข้อมูลที่ยังไม่ผ่านกระบวนการ PCA พบว่า จากการทำวิธี elbow พบว่าจำนวนกลุ่มที่เหมาะสมคือ 10 และผลลัพธ์การจัดกลุ่มพบว่า 10 กลุ่ม ประกอบด้วยสมาชิกดังนี้ 25,8,22,21,6,34,15,27,8 และ 16 มีค่า sum of square คือ 69.8%

ข้อมูลที่ผ่านกระบวนการ PCA แล้ว

ขั้นตอนแรกสำหรับ K-mean คือการเลือกจำนวนคลัสเตอร์ (K) โดยที่ใช้วิธี Elbow (Elbow method) เพื่อค้นหาค่า K ที่ดีที่สุด ดังนั้นจึงใช้การ Looping เพื่อค้นหาตัวเลือกที่ดีที่สุดโดยเริ่มจากตัวที่ 1 ถึงตัวที่ 20 และการจัดเก็บตัวแปรในรูปแบบ sse และพล็อตจากตัวที่ 1 ถึงตัวที่ 20 เปรียบเทียบกับค่า sse ของแต่ละรายการและกำหนด algorithm เป็น Lloyd สำหรับการทำ Manhattan distance

```
sse <-numeric(20)
for (k in 1:20) {
  kmeans_model <- kmeans(pca_components, centers = k,algorithm = "Lloyd")
  sse[k] <- kmeans_model$tot.withinss
}

ggplot(data.frame(K = 1:20, SSE = sse), aes(x = K, y = SSE)) +
  geom_line() +
  geom_point() +
  labs(x = "จำนวนกลุ่ม (K)", y = "SSE") +
  theme_minimal()
```

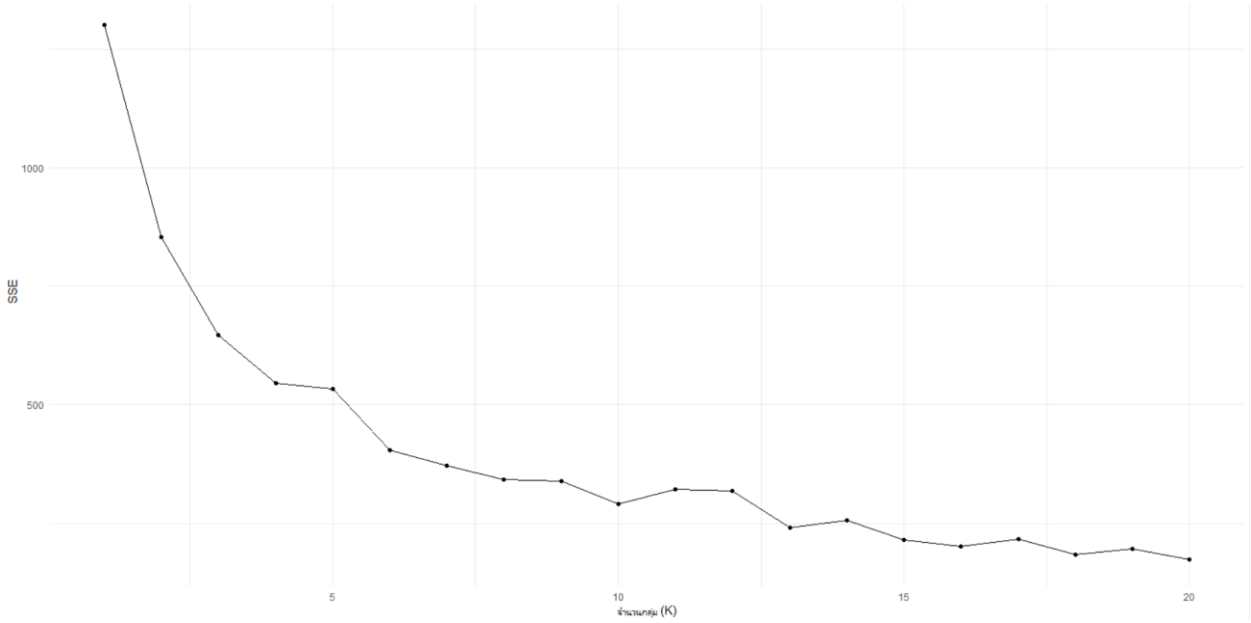

Figure 30 การคัดเลือก K ด้วยวิธี Elbow Method

ในการตัดสินใจ ให้หาค่า K ที่ทำให้ SSE ลดลงอย่างรวดเร็ว และหลังจากนั้นจึงค่อยๆ ลดลง เราเลยเลือก k = 10 เพื่อทำคลัสเตอร์ในครั้งนี้

ต่อมาคือการทำ K-means ด้วยการนำข้อมูลเป็นข้อมูลที่ผ่านกระบวนการ PCA แล้วกำหนด k ด้วย center ในที่นี้ k=10 และกำหนด algorithm เป็น Lloyd สำหรับการทำ Manhattan distance

```
kmeans_result_pca_man <- kmeans(pca_components,center = 10,algorithm = "Lloyd")
kmeans_result_pca_man
```

```
K-means clustering with 10 clusters of sizes 19, 17, 8, 6, 24, 19, 14, 20, 17, 23

Within cluster sum of squares by cluster:
 [1] 33.105167 14.366783 98.154846  9.178227 34.010911 33.412367  8.465316 68.813831
 [9] 13.759989 29.574193
 (between_SS / total_SS =  73.7 %)
```

Figure 31 ผลลัพธ์จากการทำ K-means

ผลลัพธ์ที่ได้พบว่า K-means ที่ถูกแบ่งออกเป็น 10 กลุ่มประกอบด้วยสมาชิกดังนี้ 19,17,8,6,24,19,14,20,17 และ 23 ทั้งนี้ผลที่ได้มาพบว่ามีค่า Sum of square อยู่ที่ 73.7%

จากนั้นทำการ autoplot เพื่อทำการ cluster โดยที่เทียบ PC1 กับ PC2 พบว่าได้ค่าดังนี้

```
autoplot(kmeans_result_pca_man,pca_components,frame=TRUE)
```

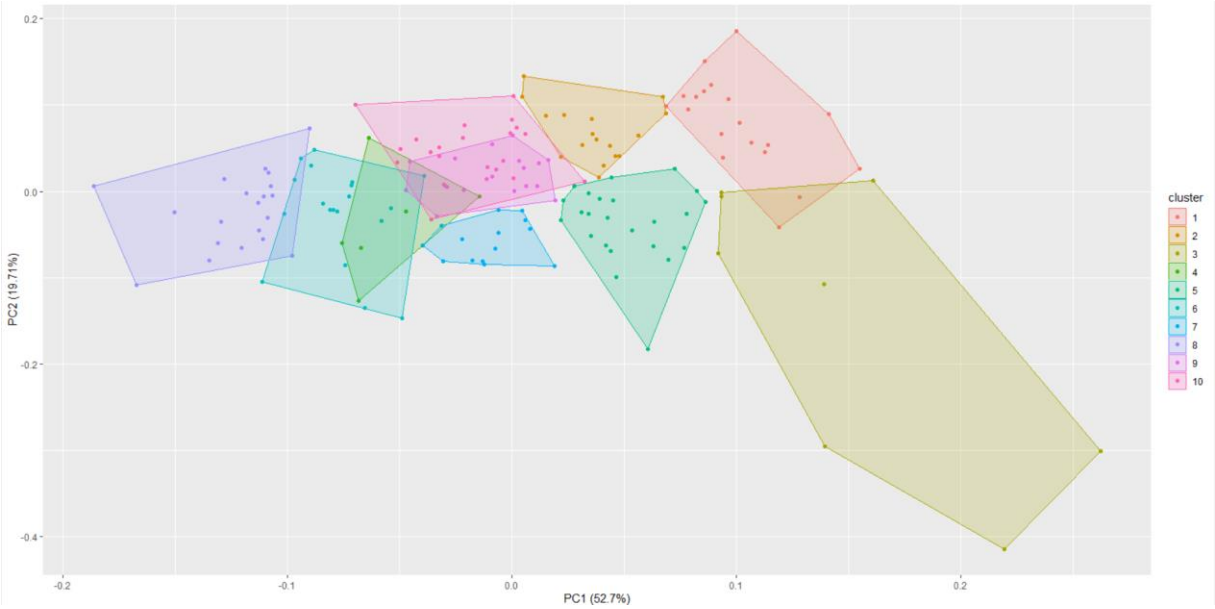

Figure 32 กราฟการจัดกลุ่มด้วย K-means

สรุปทั้งหมดของการทำ K-means ด้วยวิธีการ Manhattan Distance ผ่านข้อมูลที่ยังไม่ผ่านกระบวนการ PCA พบว่า จากการทำวิธี elbow พบว่าจำนวนกลุ่มที่เหมาะสมคือ 10 และผลลัพธ์การจัดกลุ่มพบว่า 10 กลุ่ม ประกอบด้วยสมาชิกดังนี้ 19,17,8,6,24,19,14,20,17 และ 23 มีค่า sum of square คือ 73.7%

การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA กับข้อมูลที่ผ่านกระบวนการ PCA แล้วจากการทำ Clustering ด้วย K-Means ด้วยวิธีการ Manhattan Distance

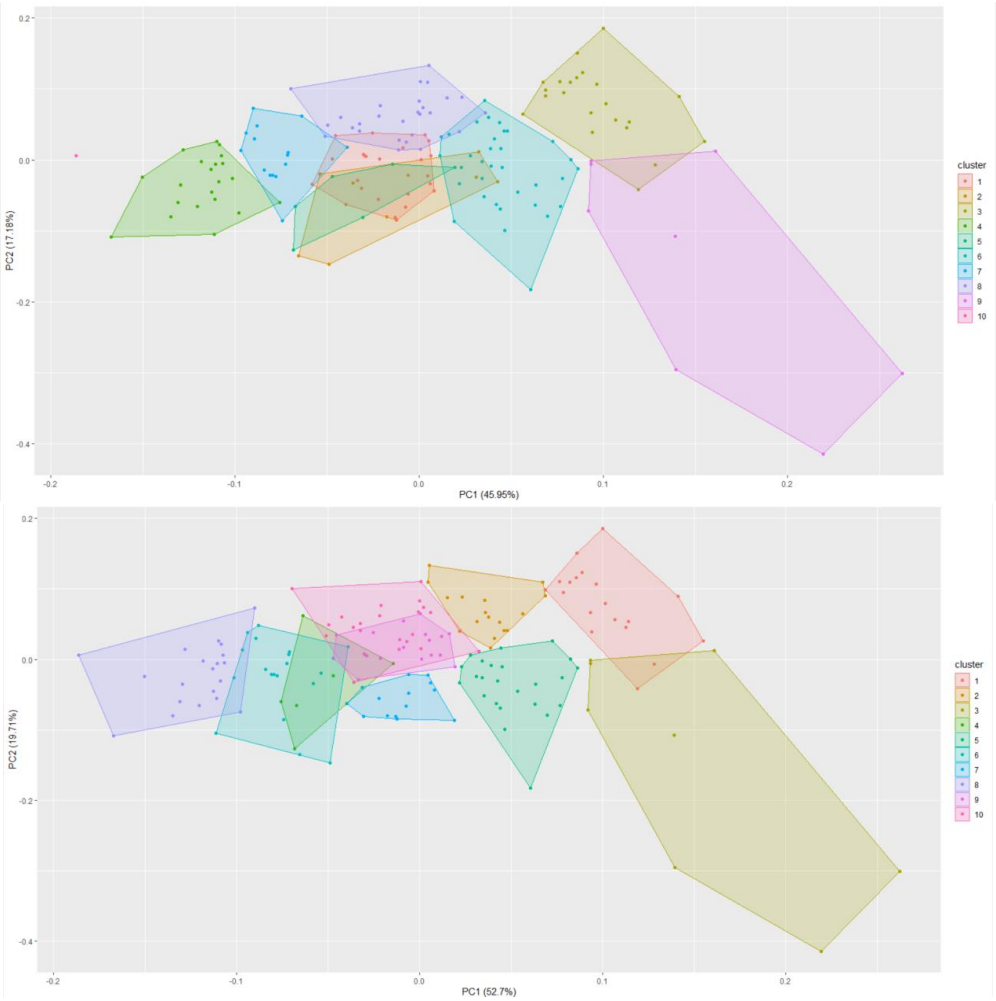

Figure 33 การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA (ด้านบน)กับข้อมูลที่ผ่าน PCA แล้ว(ด้านล่าง)

จากผลการเปรียบเทียบด้วย Visualization พบว่าข้อมูลทั้งกลุ่มนั้นแตกต่างกันในการจัดกลุ่มแต่จำนวนของกลุ่มที่ถูกเลือกนั้นมีขนาดเท่ากับ 8 ทั้ง 2 กลุ่มทั้งนี้ถ้าหากวัดที่ค่า sum of square จะทำให้สามารถพูดได้ว่ากรณีข้อมูลที่ผ่านกระบวนการ PCA แล้วนั้นดีกว่าด้วยค่า sum of square ที่มากกว่า

4.4 Hierarchical Clustering ระหว่างผ่าน PCA และยังไม่ผ่าน PCA

ในการทำ Hierarchical Clustering จะกำหนดข้อมูลที่นำมาใช้ทั้งหมด 50 ข้อมูลทั้งข้อมูลที่ยังไม่ผ่านกระบวนการการทำ PCA และผ่านกระบวนการทำ PCA แล้ว

```
test = (standardized_data[1:50,])
test
test_pca = (pca_components[1:50,])
test_pca
```

Figure 34 ข้อมูลที่ใช้เฉพาะ Hierarchical

4.4.1 Hierarchical Clustering ด้วยวิธีการ Euclidean Distance

ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA

ขั้นแรกคำนวณระยะทางด้วยวิธี Euclidean Distance และใช้ข้อมูลของตัวแปร test ที่เป็นข้อมูลยังไม่ทำ PCA พร้อมคำสั่ง dist และเก็บไว้ที่ Distance_mat จากนั้นคุณใช้คำสั่ง hclust เพื่อจัดกลุ่มโดยเลือก Distance_mat และเก็บไว้ในตัวแปร Heirar_cl

```
distance_mat <- dist(test,method = 'euclidean')
distance_mat
Hierar_cl <- hclust(distance_mat)
Hierar_cl
```

Figure 35 กำหนดวิธีหาระยะห่างและจัดกลุ่ม

เมื่อคุณประมวลผลพบจะพบว่าใช้วิธีหาระยะห่างด้วย Euclidean distance และมีจำนวนข้อมูล 50 ข้อมูล

```
Call:
hclust(d = distance_mat)

Cluster method   : complete
Distance         : euclidean
Number of objects: 50
```

Figure 36 ข้อมูลทั้งหมดของการจัดกลุ่ม

จากนั้นทำการ plot ด้วย Dendrogram

```
plot(Hierar_cl)
```

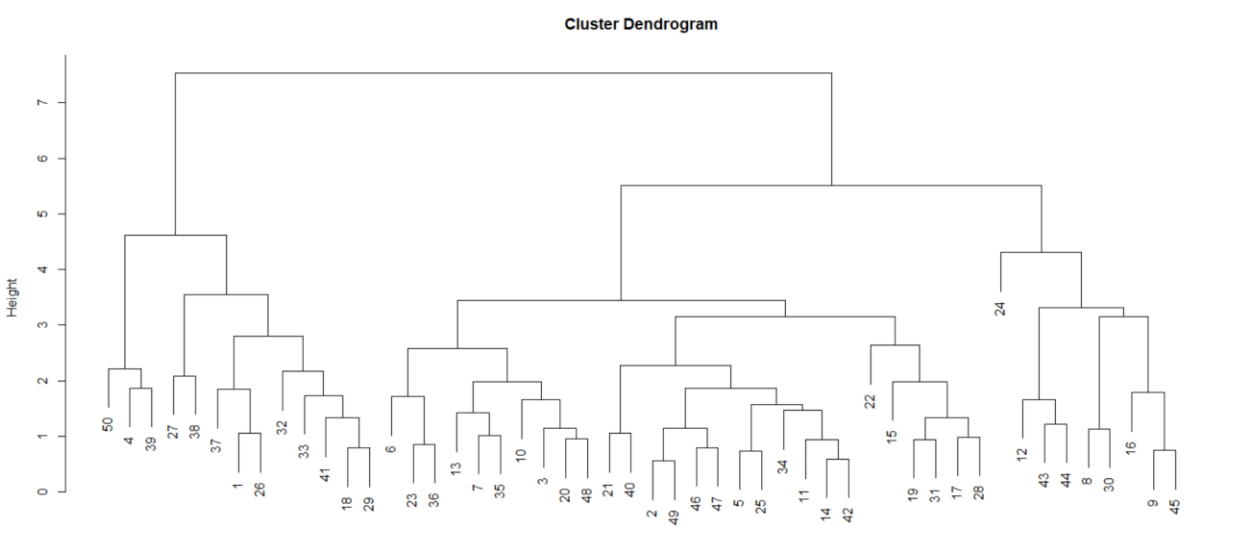

Figure 37 plot dendrogram

จะได้ตาราง Dendrogram ดังนี้ทั้งนี้เราสามารถตัดสินเลือกกลุ่มได้โดยการตีเส้น เราจึงตัดสินใจเลือกกลุ่มทั้งหมด 3 กลุ่ม

ในการเลือกกลุ่ม 3 กลุ่มจะตัดความสูงของระยะห่างอยู่ที่ 5

```
plot(Hierar_cl)
abline(h=5,col = "green")
```

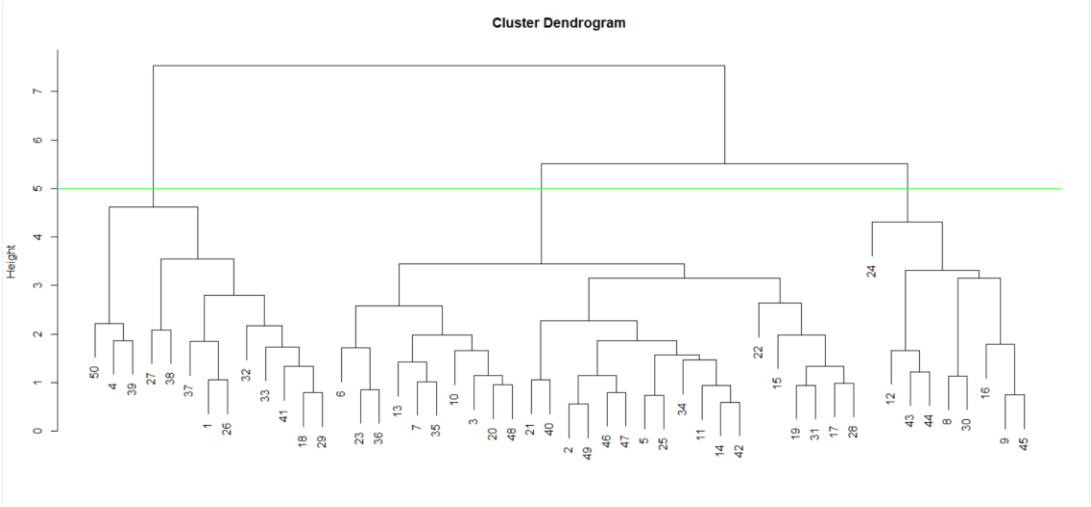

Figure 38 เส้นการตัดสินใจเลือกกลุ่ม

จากนั้นเราจึงทำการแบ่งกลุ่มเท่ากับ 3 กลุ่มโดยที่ใช้ fit = 3 เพื่อแสดงผลของแต่ละกลุ่มจากนั้น plot ด้วยคำสั่ง plot และตามด้วย rect.hclust เพื่อดูสมาชิกของแต่ละกลุ่ม

```
fit<- cutree(Hierar_cl,k=3)
fit
plot(Hierar_cl)
table(fit)
rect.hclust(Hierar_cl,k=3,border = "green")
```

```
fit
 1  2  3
13 28  9
```

Figure 39 จำนวนสมาชิกในแต่ละกลุ่ม

หากเราแบ่งกลุ่มเท่ากับ 3 พบว่ากลุ่มที่ 1 มีสมาชิกทั้งหมด 13 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 28 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 9 ตัว

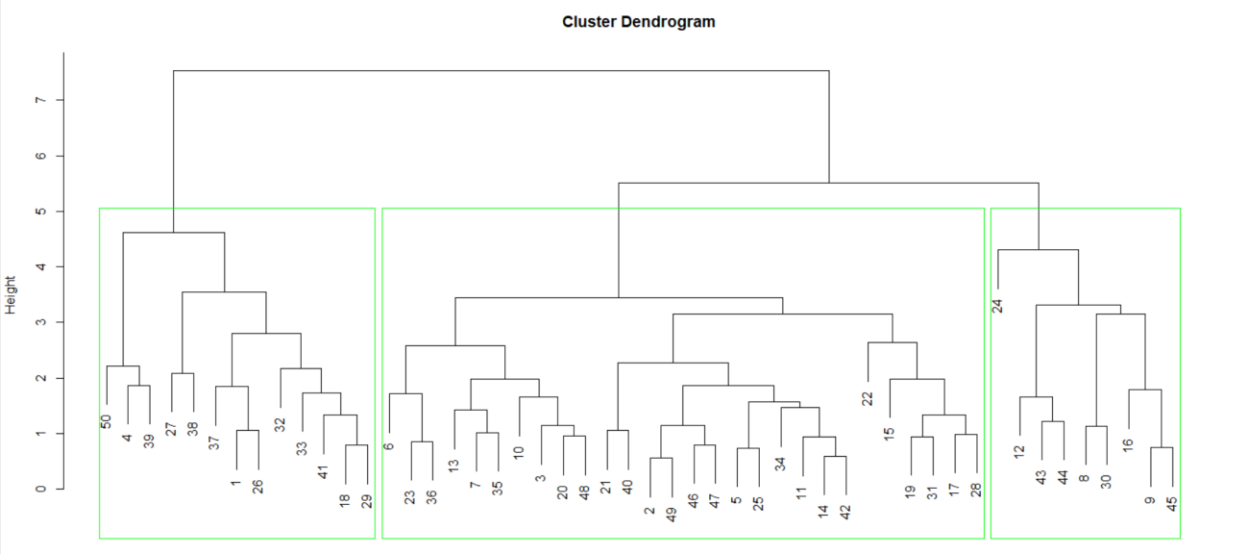

Figure 40 Dendrogram ที่แยกกลุ่มเรียบร้อย

สรุปทั้งหมดของการทำ Hierarchical Clustering ด้วยวิธีการ Euclidean Distance ผ่านข้อมูลที่ยังไม่ผ่านกระบวนการ PCA จำนวณ 50 ข้อมูลพบว่า ภายใต้การตัดสินใจหลังจากทำ Dendrogram จึงตัดสินใจจัดกลุ่มมาทั้งหมด 3 กลุ่มโดยมีสมาชิกแต่ละกลุ่มคือกลุ่มที่ 1 มีสมาชิกทั้งหมด 13 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 28 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 9 ตัว

ข้อมูลที่ผ่านกระบวนการ PCA แล้ว

ขั้นแรกคำนวณระยะทางด้วยวิธี Euclidean Distance และใช้ข้อมูลของตัวแปร test_pca ที่เป็นข้อมูลทำ PCA แล้วพร้อมคำสั่ง dist และเก็บไว้ที่ Distance_mat_pca จากนั้นคุณใช้คำสั่ง hclust เพื่อจัดกลุ่มโดยเลือก Distance_mat_pca และเก็บไว้ในตัวแปร Heirar_cl_pca

```
distance_mat_pca <- dist(test_pca,method = 'euclidean')
distance_mat_pca
Hierar_cl_pca <- hclust(distance_mat_pca)
Hierar_cl_pca
```

Figure 41 กำหนดวิธีหาระยะห่างและจัดกลุ่ม

เมื่อคุณประมวลผลพบจะพบว่าใช้วิธีหาระยะห่างด้วย Euclidean distance และมีจำนวนข้อมูล 50 ข้อมูล

```
Call:
hclust(d = distance_mat_pca)

Cluster method   : complete
Distance         : euclidean
Number of objects: 50
```

Figure 42 ข้อมูลทั้งหมดของการจัดกลุ่ม

จากนั้นทำการ plot ด้วย Dendrogram

```
plot(Hierar_cl_pca)
```

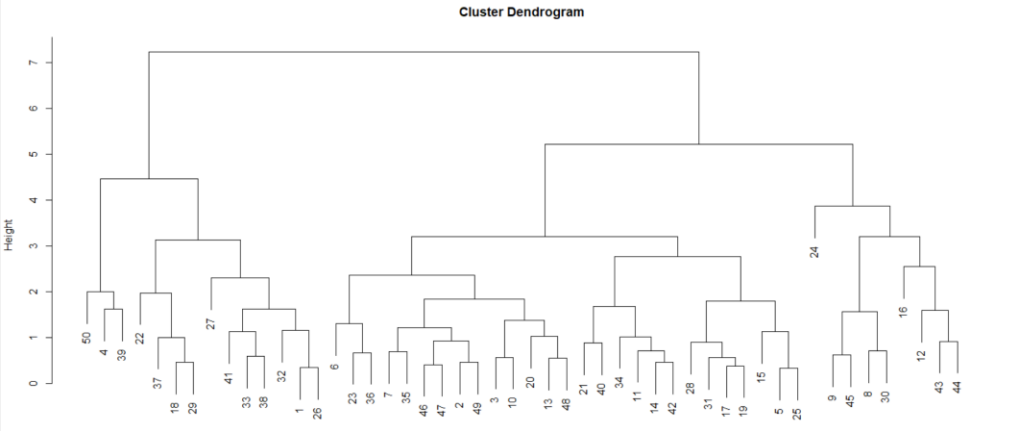

Figure 43 plot dendrogram

จะได้ตาราง Dendrogram ดังนี้ทั้งนี้เราสามารถตัดสินเลือกกลุ่มได้โดยการตีเส้น เราจึงตัดสินใจเลือกกลุ่มทั้งหมด 3 กลุ่ม

ในการเลือกกลุ่ม 3 กลุ่มจะตัดความสูงของระยะห่างอยู่ที่ 5

```
plot(Hierar_cl_pca)
abline(h=5,col = "green")
```

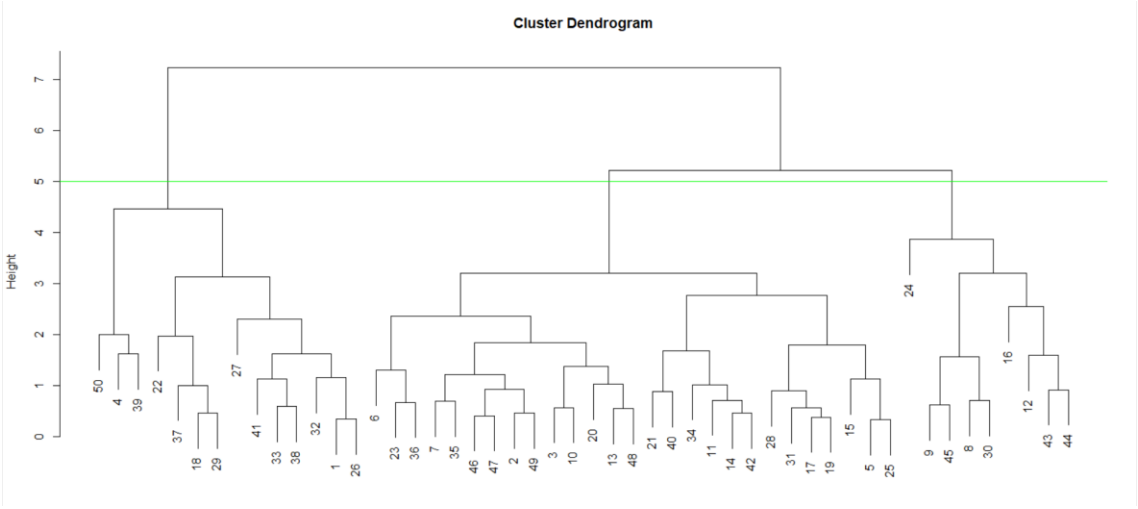

Figure 44 เส้นการตัดสินใจเลือกกลุ่ม

จากนั้นเราจึงทำการแบ่งกลุ่มเท่ากับ 3 กลุ่มโดยที่ใช้ fit = 3 เพื่อแสดงผลของแต่ละกลุ่มจากนั้น plot ด้วยคำสั่ง plot และตามด้วย rect.hclust เพื่อดูสมาชิกของแต่ละกลุ่ม

```
fit<- cutree(Hierar_cl_pca,k=3)
fit
plot(Hierar_cl_pca)
table(fit)
rect.hclust(Hierar_cl_pca,k=3,border = "green")
```

```
fit
 1  2  3
14 27  9
```

Figure 45 จำนวนสมาชิกในแต่ละกลุ่ม

หากเราแบ่งกลุ่มเท่ากับ 3 พบว่ากลุ่มที่ 1 มีสมาชิกทั้งหมด 14 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 27 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 9 ตัว

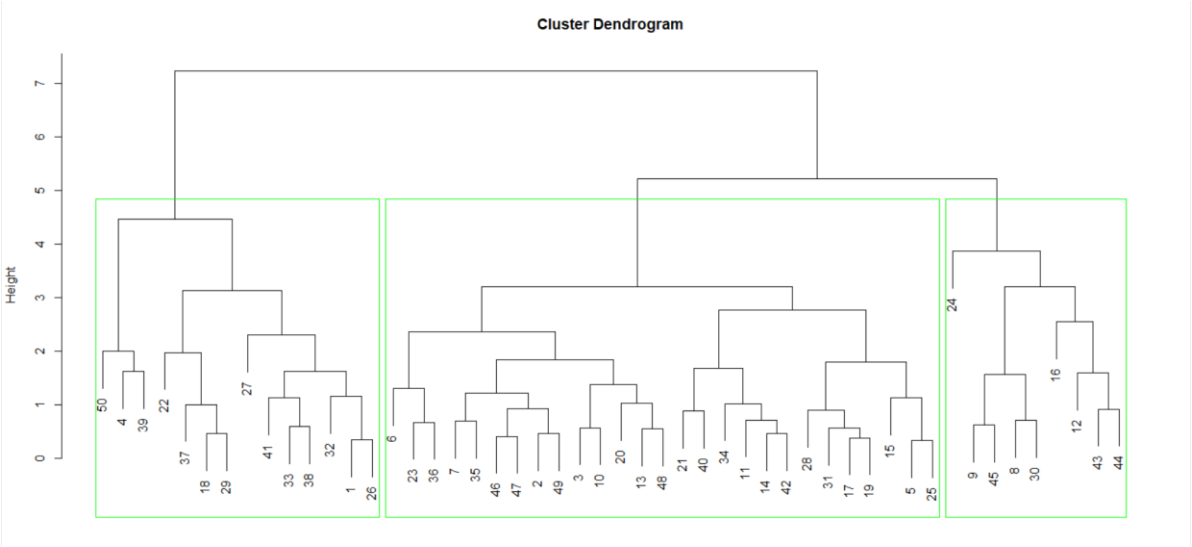

Figure 46 Dendrogram ที่แยกกลุ่มเรียบร้อย

สรุปทั้งหมดของการทำ Hierarchical Clustering ด้วยวิธีการ Euclidean Distance ผ่านข้อมูลที่ผ่านกระบวนการ PCA แล้วจำนวณ 50 ข้อมูลพบว่า ภายใต้การตัดสินใจหลังจากทำ Dendrogram จึงตัดสินใจจัดกลุ่มมาทั้งหมด 3 กลุ่มโดยมีสมาชิกแต่ละกลุ่มคือกลุ่มที่ 1 มีสมาชิกทั้งหมด 14 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 27 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 9 ตัว

การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA กับข้อมูลที่ผ่านกระบวนการ PCA แล้วจากการทำ Clustering ด้วย Hierarchical Clustering ด้วยวิธีการ Euclidean Distance

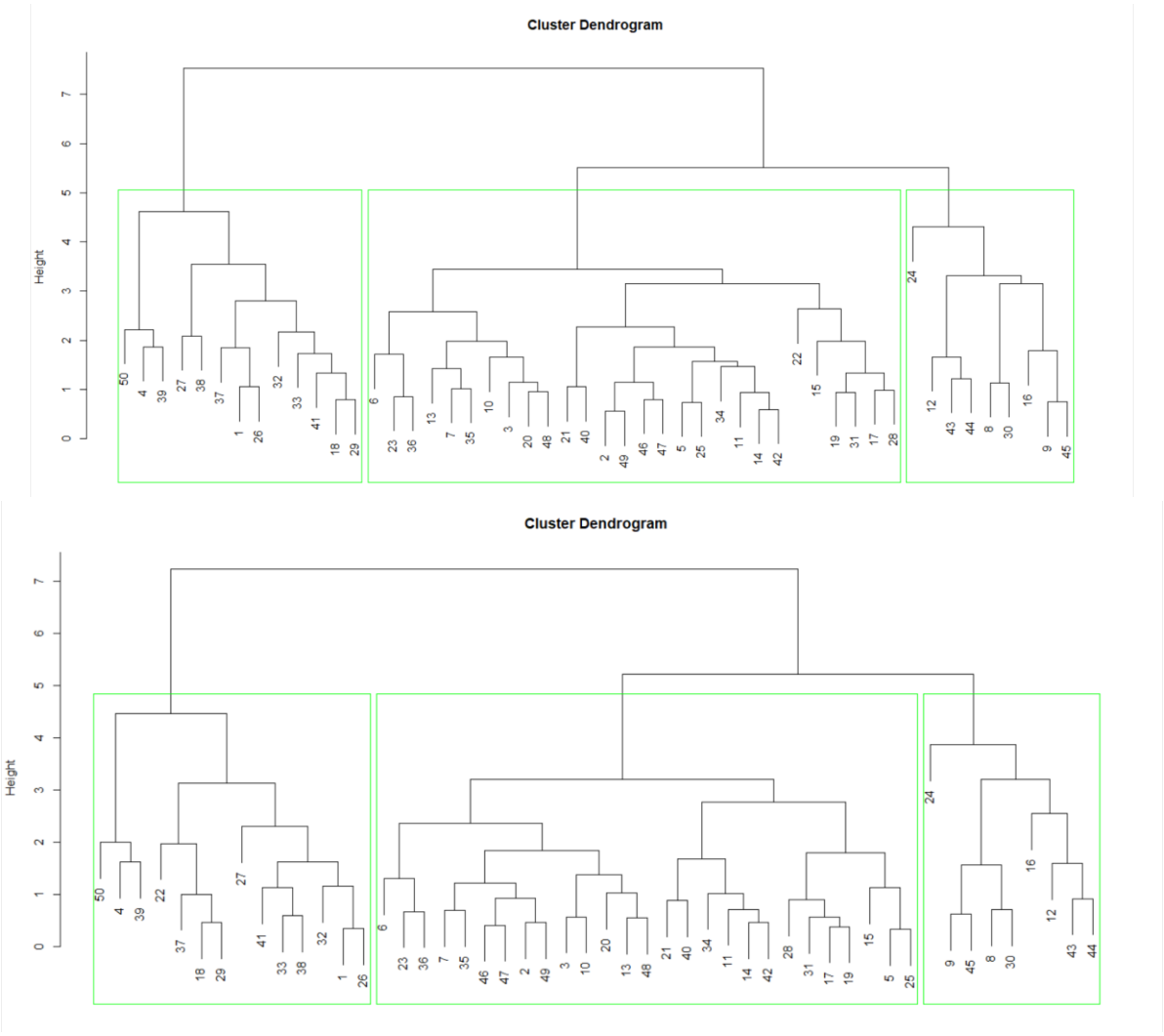

Figure 47 การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA (ด้านบน)กับข้อมูลที่ผ่าน PCA แล้ว(ด้านล่าง)

จากผลการเปรียบเทียบด้วย Visualization โดยกำหนดที่กลุ่มทั้ง 3 กลุ่มพบว่าข้อมูลทั้งกลุ่มนั้นแตกต่างกันในการจัดกลุ่มทั้งนี้ถ้าหากวัดที่ค่าระยะห่างของการตัด (h) จะทำให้สามารถพูดได้ว่ากรณีข้อมูลที่ไม่ผ่านกระบวนการ PCA กับข้อมูลที่ผ่านกระบวนการ PCA แล้วไม่แตกต่างกันที่ตัด h เท่ากับ 5

4.4.2 Hierarchical Clustering ด้วยวิธีการ Euclidean Distance

ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA

ขั้นแรกคำนวณระยะทางด้วยวิธี manhattan distance และใช้ข้อมูลของตัวแปร test ที่เป็นข้อมูลยังไม่ทำ PCA พร้อมคำสั่ง dist และเก็บไว้ที่ Distance_mat_man จากนั้นคุณใช้คำสั่ง hclust เพื่อจัดกลุ่มโดยเลือก Distance_mat_man และเก็บไว้ในตัวแปร Heirar_cl_man

```
distance_mat_man <- dist(test,method = 'manhattan')
distance_mat_man
Hierar_cl_man <- hclust(distance_mat_man)
Hierar_cl_man
```

Figure 48 กำหนดวิธีหาระยะห่างและจัดกลุ่ม

เมื่อคุณประมวลผลพบจะพบว่าใช้วิธีหาระยะห่างด้วย manhattan distance และมีจำนวนข้อมูล 50 ข้อมูล

```
Call:
hclust(d = distance_mat_man)

Cluster method   : complete
Distance         : manhattan
Number of objects: 50
```

Figure 49 ข้อมูลทั้งหมดของการจัดกลุ่ม

จากนั้นทำการ plot ด้วย Dendrogram

```
plot(Hierar_cl_man)
```

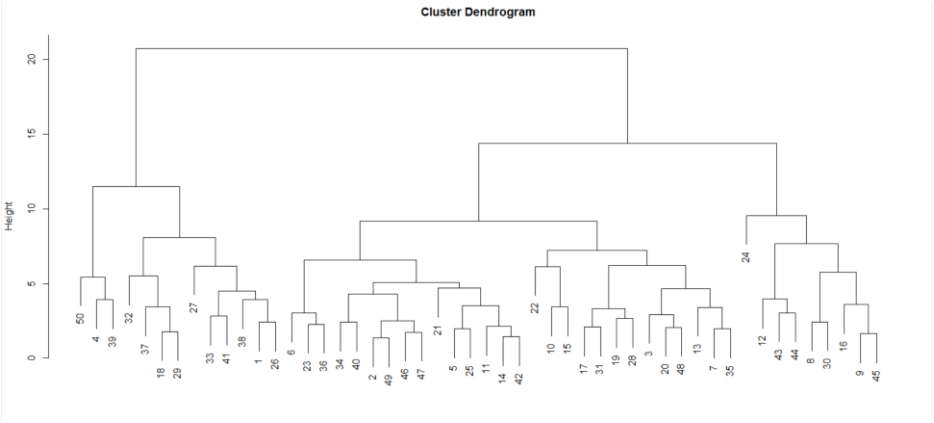

Figure 50 plot dendrogram

จะได้ตาราง Dendrogram ดังนี้ทั้งนี้เราสามารถตัดสินเลือกกลุ่มได้โดยการตีเส้น เราจึงตัดสินใจเลือกกลุ่มทั้งหมด 3 กลุ่ม

ในการเลือกกลุ่ม 3 กลุ่มจะตัดความสูงของระยะห่างอยู่ที่ 13

```
plot(Hierar_cl_man)
abline(h=13,col = "green")
```

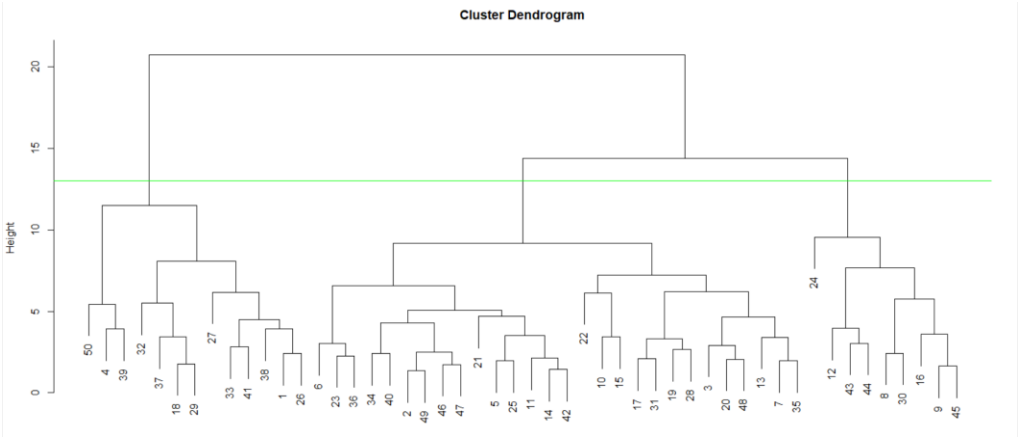

Figure 51 เส้นการตัดสินใจเลือกกลุ่ม

จากนั้นเราจึงทำการแบ่งกลุ่มเท่ากับ 3 กลุ่มโดยที่ใช้ fit = 3 เพื่อแสดงผลของแต่ละกลุ่มจากนั้น plot ด้วยคำสั่ง plot และตามด้วย rect.hclust เพื่อดูสมาชิกของแต่ละกลุ่ม

```
fit<- cutree(Hierar_cl_man,k=3)
fit
plot(Hierar_cl_man)
table(fit)
rect.hclust(Hierar_cl_man,k=3,border = "green")
```

```
fit
 1  2  3
13 28  9
```

Figure 52 จำนวนสมาชิกในแต่ละกลุ่ม

หากเราแบ่งกลุ่มเท่ากับ 3 พบว่ากลุ่มที่ 1 มีสมาชิกทั้งหมด 13 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 28 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 9 ตัว

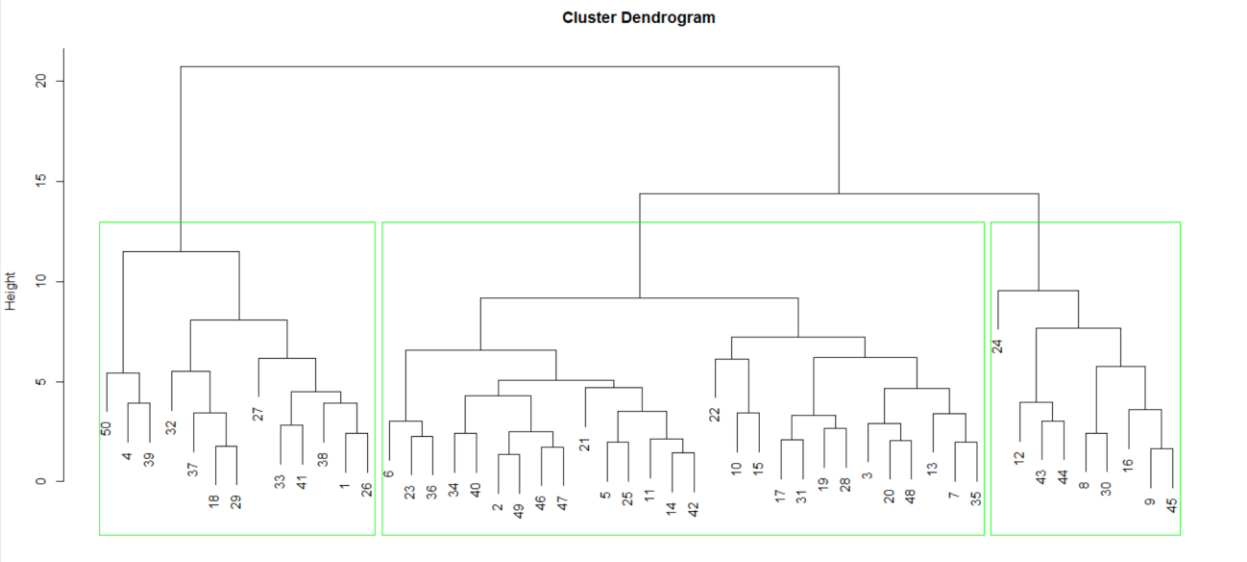

Figure 53 Dendrogram ที่แยกกลุ่มเรียบร้อย

สรุปทั้งหมดของการทำ Hierarchical Clustering ด้วยวิธีการ Manhattan Distance ผ่านข้อมูลที่ยังไม่ผ่านกระบวนการ PCA จำนวณ 50 ข้อมูลพบว่า ภายใต้การตัดสินใจหลังจากทำ Dendrogram จึงตัดสินใจจัดกลุ่มมาทั้งหมด 3 กลุ่มโดยมีสมาชิกแต่ละกลุ่มคือกลุ่มที่ 1 มีสมาชิกทั้งหมด 13 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 28 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 9 ตัว

ข้อมูลที่ผ่านกระบวนการ PCA แล้ว

ขั้นแรกคำนวณระยะทางด้วยวิธี Manhattan Distance และใช้ข้อมูลของตัวแปร test_pca ที่เป็นข้อมูลทำ PCA แล้วพร้อมคำสั่ง dist และเก็บไว้ที่ Distance_mat_pca_man จากนั้นคุณใช้คำสั่ง hclust เพื่อจัดกลุ่มโดยเลือก Distance_mat_pca_man และเก็บไว้ในตัวแปร Heirar_cl_pca_man

```
distance_mat_pca_man <- dist(test_pca,method = 'manhattan')
distance_mat_pca_man
Hierar_cl_pca_man <- hclust(distance_mat_pca_man)
Hierar_cl_pca_man
```

Figure 54 กำหนดวิธีหาระยะห่างและจัดกลุ่ม

เมื่อคุณประมวลผลพบจะพบว่าใช้วิธีหาระยะห่างด้วย Manhattan distance และมีจำนวนข้อมูล 50 ข้อมูล

```
Call:
hclust(d = distance_mat_pca_man)

Cluster method   : complete
Distance         : manhattan
Number of objects: 50
```

Figure 55 ข้อมูลทั้งหมดของการจัดกลุ่ม

จากนั้นทำการ plot ด้วย Dendrogram

```
plot(Hierar_cl_pca_man)
```

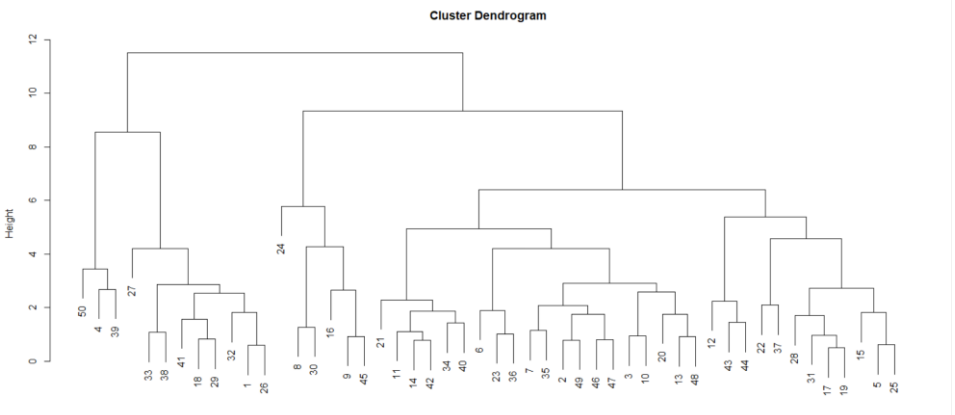

Figure 56 plot dendrogram

จะได้ตาราง Dendrogram ดังนี้ทั้งนี้เราสามารถตัดสินเลือกกลุ่มได้โดยการตีเส้น เราจึงตัดสินใจเลือกกลุ่มทั้งหมด 3 กลุ่ม

ในการเลือกกลุ่ม 3 กลุ่มจะตัดความสูงของระยะห่างอยู่ที่ 9

```
plot(Hierar_cl_pca_man)
abline(h=9,col = "green")
```

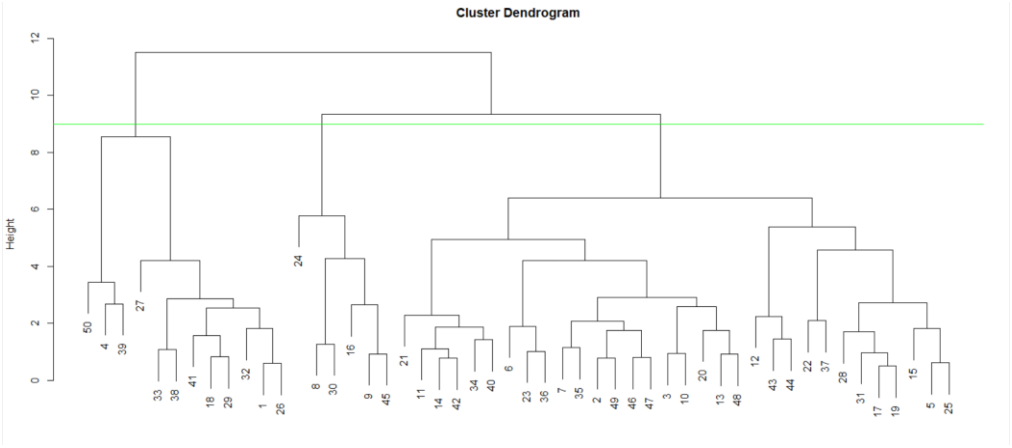

Figure 57 เส้นการตัดสินใจเลือกกลุ่ม

จากนั้นเราจึงทำการแบ่งกลุ่มเท่ากับ 3 กลุ่มโดยที่ใช้ fit = 3 เพื่อแสดงผลของแต่ละกลุ่มจากนั้น plot ด้วยคำสั่ง plot และตามด้วย rect.hclust เพื่อดูสมาชิกของแต่ละกลุ่ม

```
fit<- cutree(Hierar_cl_pca_man,k=3)
fit
plot(Hierar_cl_pca_man)
table(fit)
rect.hclust(Hierar_cl_pca_man,k=3,border = "green")
```

```
fit
 1  2  3
12 32  6
```

Figure 58 จำนวนสมาชิกในแต่ละกลุ่ม

หากเราแบ่งกลุ่มเท่ากับ 3 พบว่ากลุ่มที่ 1 มีสมาชิกทั้งหมด 12 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 32 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 6 ตัว

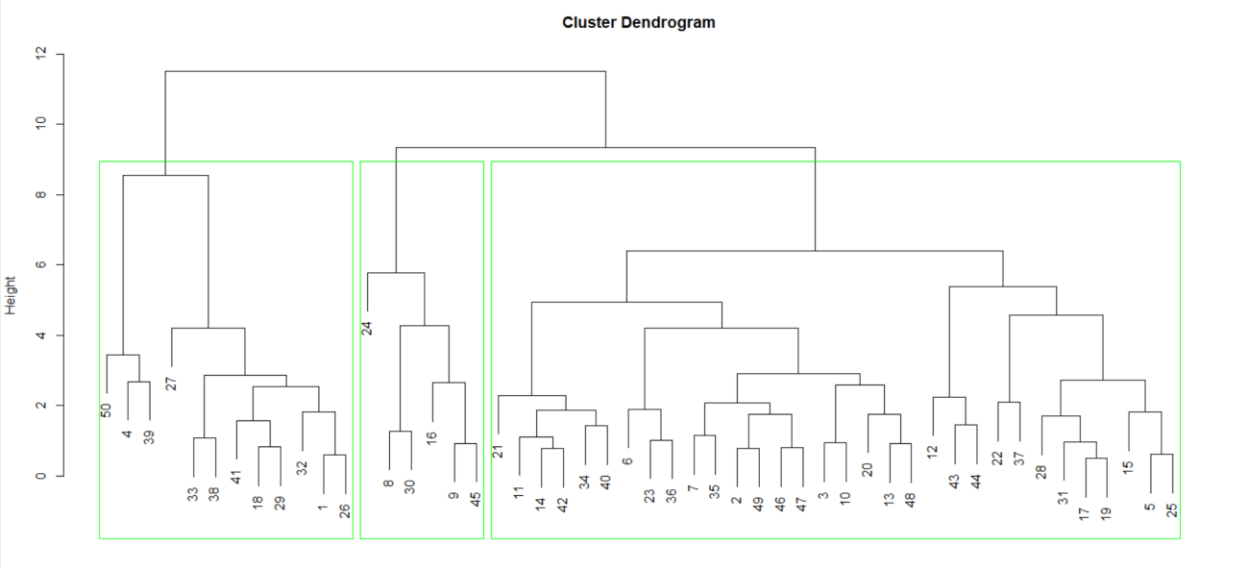

Figure 59 Dendrogram ที่แยกกลุ่มเรียบร้อย

สรุปทั้งหมดของการทำ Hierarchical Clustering ด้วยวิธีการ Manhattan Distance ผ่านข้อมูลที่ผ่านกระบวนการ PCA แล้วจำนวณ 50 ข้อมูลพบว่า ภายใต้การตัดสินใจหลังจากทำ Dendrogram จึงตัดสินใจจัดกลุ่มมาทั้งหมด 3 กลุ่มโดยมีสมาชิกแต่ละกลุ่มคือกลุ่มที่ 1 มีสมาชิกทั้งหมด 12 ตัวกลุ่มที่ 2 สมาชิกทั้งหมด 32 ตัวและกลุ่มที่ 3 มีสมาชิกทั้งหมด 6 ตัว

การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA กับข้อมูลที่ผ่านกระบวนการ PCA แล้วจากการทำ Clustering ด้วย Hierarchical Clustering ด้วยวิธีการ Manhattan Distance

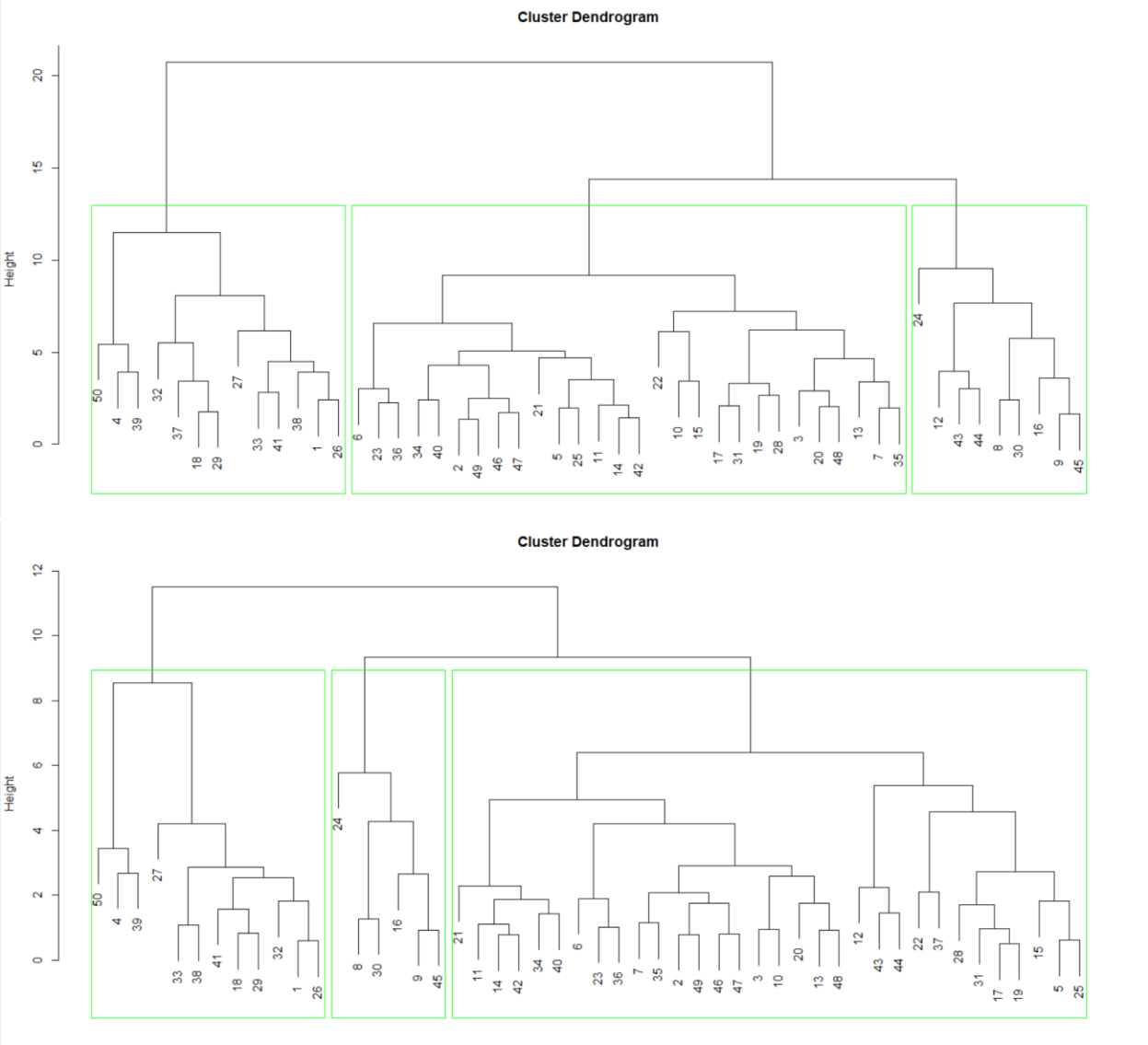

Figure 60 การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA (ด้านบน)กับข้อมูลที่ผ่าน PCA แล้ว(ด้านล่าง)

จากผลการเปรียบเทียบด้วย Visualization โดยกำหนดที่กลุ่มทั้ง 3 กลุ่มพบว่าข้อมูลทั้งกลุ่มนั้นแตกต่างกันในการจัดกลุ่มทั้งนี้ถ้าหากวัดที่ค่าระยะห่างของการตัด (h) จะทำให้สามารถพูดได้ว่าข้อมูลที่ผ่านกระบวนการ PCA แล้วทำได้ดีกว่าโดยที่ตัด h เท่ากับ 9 ซึ่งมีระยะทางที่น้อยกว่า

4.5 DBScan ระหว่างผ่าน PCA และยังไม่ผ่าน PCA

4.5.1 DBScan ด้วยวิธีการ Euclidean Distance

ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA

ขั้นตอนแรกของ DBScan คือ กำหนดระยะห่างเป็น Euclidean distance โดยใช้คำสั่ง proxy และใช้ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA และเก็บไว้ใน dist_matrix. จากนั้นทำการหา DBscan ด้วยการกำหนด parameter 3 ตัวได้แก่ dist_matrix, eps =1,และ minPts =2

```
dist_matrix <- proxy::dist(standardized_data,method = "Euclidean")

Db_cl <- dbscan::dbscan(dist_matrix,eps = 1, minPts = 2)
Db_cl
```

Figure 61 กำหนดระยะห่างและตั้งพารามิเตอร์สำหรับ DBScan

เมื่อทำการประมวลผลจะพบว่าด้วยวิธีการ Euclidean distance จะได้กลุ่มทั้งหมด 12 กลุ่มประกอบไปด้วย 8,59,16,2,2,4,2,3,5,2,2 และ 2 ตามลำดับและ มี Noise point ทั้งหมด 60 ตัว

```
DBSCAN clustering for 167 objects.
Parameters: eps = 1, minPts = 2
Using Euclidean distances and borderpoints = TRUE
The clustering contains 12 cluster(s) and 60 noise points.

 0  1  2  3  4  5  6  7  8  9 10 11 12
60  8 59 16  2  2  4  2  3  5  2  2  2
```

Figure 62 ข้อมูลของการจัดกลุ่ม

จากนั้นทำการ plot dbscan ด้วยกลุ่มทั้งหมด

```
plot(standardized_data, col = Db_cl$cluster)
```

Figure 63 การ plot ข้อมูล

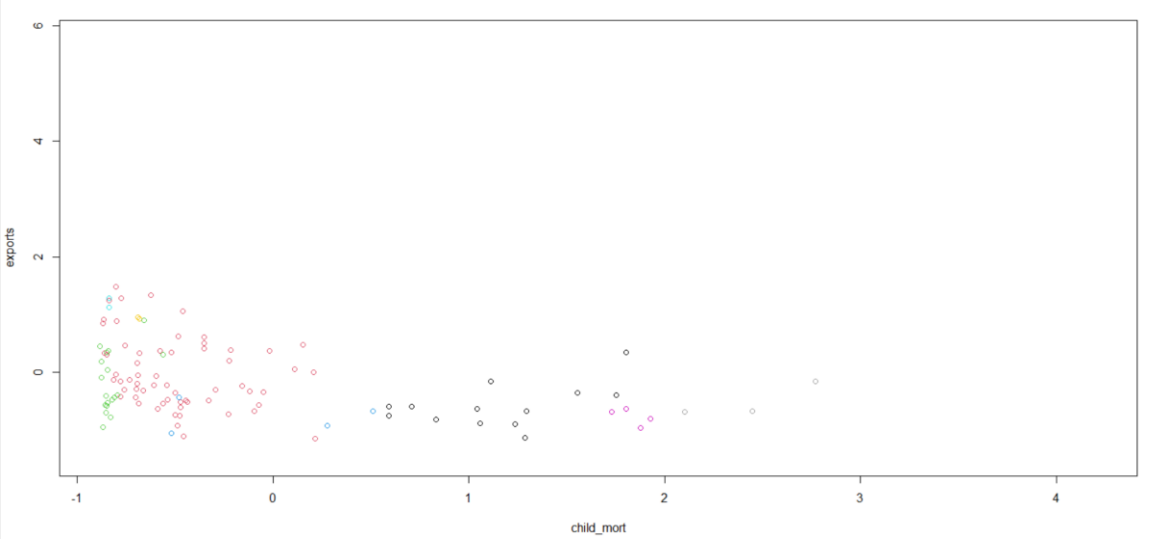

Figure 64 กราฟแสดงผลจาก DBScan

สรุปทั้งหมดของการทำ DBScan ด้วยวิธีการ Euclidean Distance ผ่านข้อมูลที่ยังไม่ผ่านกระบวนการ PCA พบว่าถ้ากำหนด eps = 1 และ minPts = 2 จะได้ทั้งหมด 12 กลุ่มประกอบไปด้วย 8,59,16,2,2,4,2,3,5, 2,2 และ 2 ตามลำดับและ มี Noise point ทั้งหมด 60 ตัว

ข้อมูลที่ผ่านกระบวนการ PCA แล้ว

ขั้นตอนแรกของ DBScan คือ กำหนดระยะห่างเป็น Euclidean distance โดยใช้คำสั่ง proxy และใช้ข้อมูลที่ผ่านกระบวนการ PCA แล้วและเก็บไว้ใน dist_matrix_pca จากนั้นทำการหา DBscan ด้วยการกำหนด parameter 3 ตัวได้แก่ dist_matrix_pca, eps =1,และ minPts =2

```
dist_matrix_pca <- proxy::dist(pca_components,method = "Euclidean")

Db_cl_pca <- dbscan::dbscan(dist_matrix_pca,eps = 1, minPts = 2)
Db_cl_pca
```

Figure 65 กำหนดระยะห่างและตั้งพารามิเตอร์สำหรับ DBScan

เมื่อทำการประมวลผลจะพบว่าด้วยวิธีการ Euclidean distance จะได้กลุ่มทั้งหมด 7 กลุ่มประกอบไปด้วย 133,2,2,2,2,2และ 3 ตามลำดับและ มี Noise point ทั้งหมด 21 ตัว

```
DBSCAN clustering for 167 objects.
Parameters: eps = 1, minPts = 2
Using Euclidean distances and borderpoints = TRUE
The clustering contains 7 cluster(s) and 21 noise points.

  0   1   2   3   4   5   6   7
 21 133   2   2   2   2   2   3
```

Figure 66 ข้อมูลของการจัดกลุ่ม

จากนั้นทำการ plot dbscan ด้วยกลุ่มทั้งหมด

```
plot(standardized_data, col = Db_cl_pca$cluster)
```

Figure 67 การ plot ข้อมูล

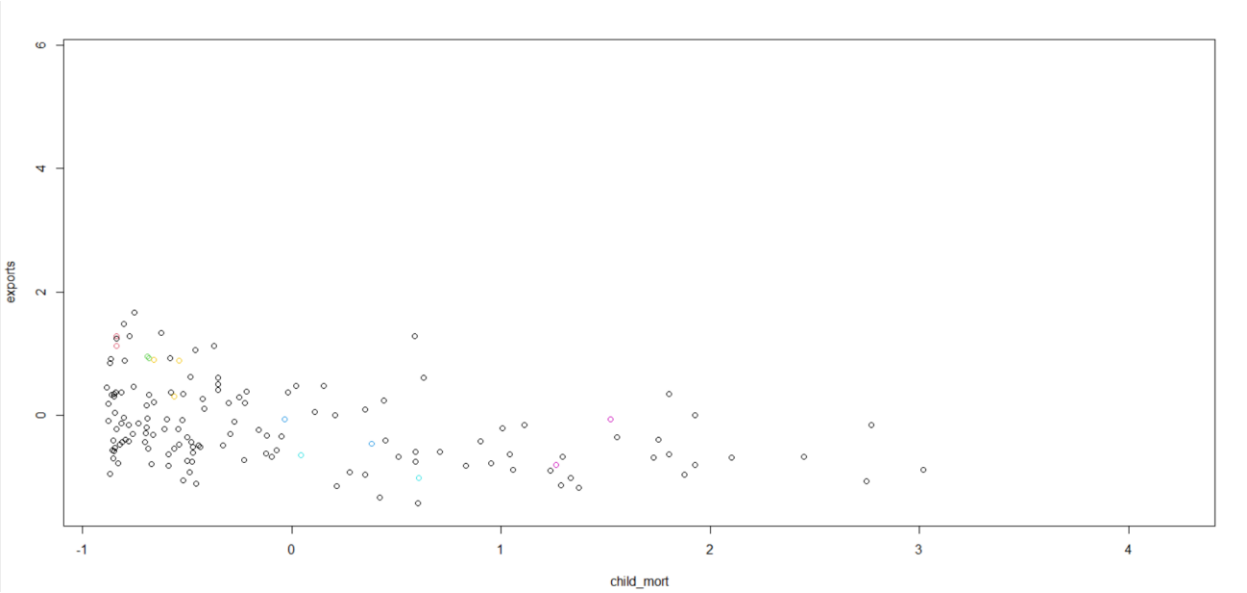

*Figure 68 กราฟแสดงผลจาก **DBScan***

สรุปทั้งหมดของการทำ DBScan ด้วยวิธีการ Euclidean Distance ผ่านข้อมูลที่ผ่านกระบวนการ PCA แล้วพบว่าถ้ากำหนด eps = 1 และ minPts = 2 จะได้ทั้งหมด 12 กลุ่มประกอบไปด้วย 7 กลุ่มประกอบไปด้วย 133,2,2,2,2,2และ 3 ตามลำดับและ มี Noise point ทั้งหมด 21 ตัว

การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA กับข้อมูลที่ผ่านกระบวนการ PCA แล้วจากการทำ Clustering ด้วย DBScan ด้วยวิธีการ Euclidean Distance

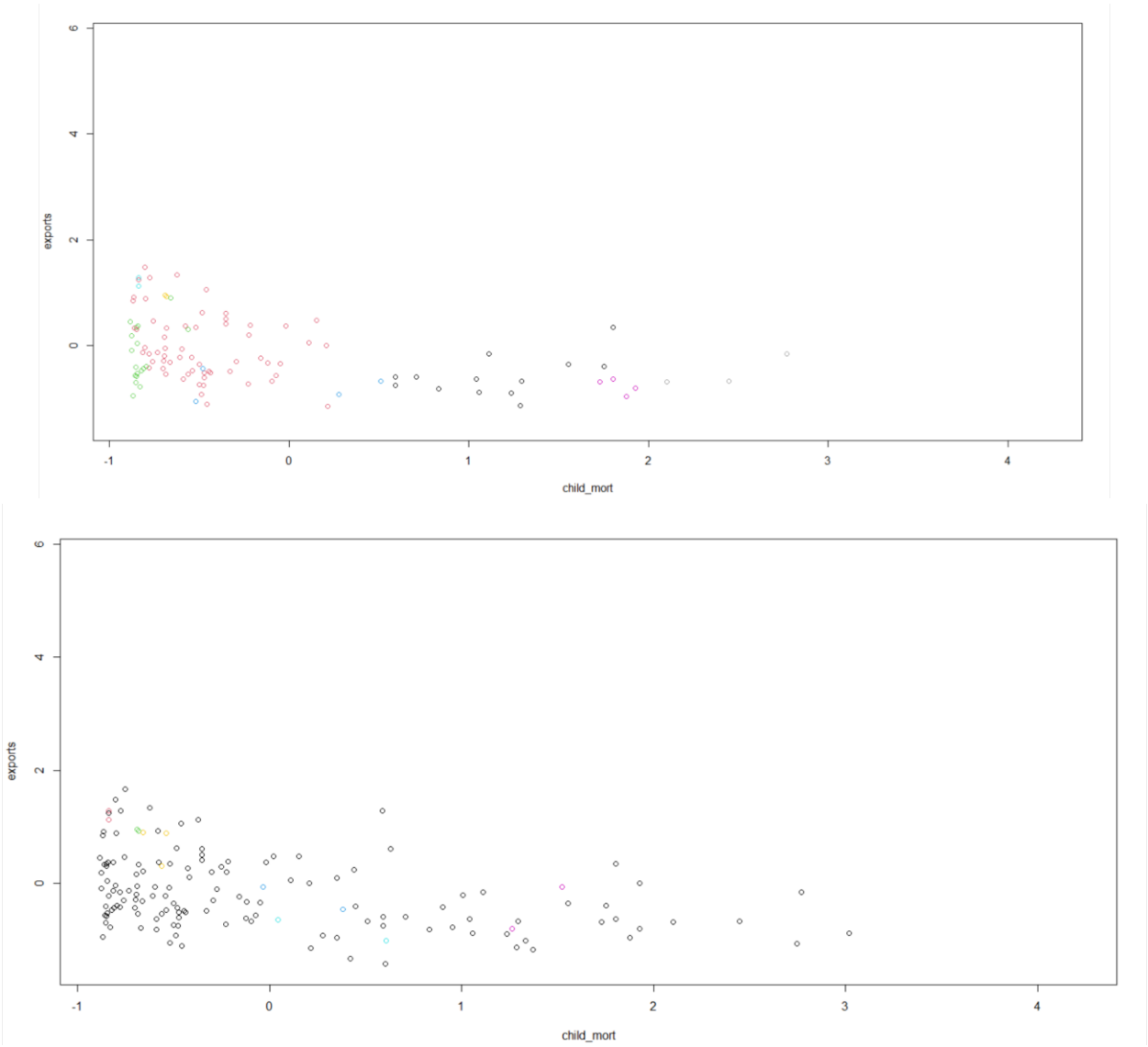

Figure 69 การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA (ด้านบน)กับข้อมูลที่ผ่าน PCA แล้ว(ด้านล่าง)

จากผลการเปรียบเทียบด้วย Visualization โดยกำหนดที่ eps = 1 และ Minpts = 2 พบว่าข้อมูลทั้งกลุ่มนั้นแตกต่างกันในการจัดกลุ่มทั้งนี้ถ้าหากวัดการจับกลุ่มข้อมูลที่ผ่านกระบวนการ PCA แล้วทำได้ดีกว่าด้วยจำนวนกลุ่มนี้น้อยกว่า แล้วถ้าเปรียบเทียบค่า noise point ข้อมูลที่ผ่านกระบวนการ PCA แล้วทำได้ดีกว่าด้วยจำนวน noise point ที่น้อยกว่าจำนวน 21 ตัว

4.5.1 DBScan ด้วยวิธีการ Manhattan Distance

ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA

ขั้นตอนแรกของ DBScan คือ กำหนดระยะห่างเป็น Manhattan distance โดยใช้คำสั่ง proxy และใช้ข้อมูลที่ยังไม่ผ่านกระบวนการ PCA และเก็บไว้ใน dist_matrix_man จากนั้นทำการหา DBscan ด้วยการกำหนด parameter 3 ตัวได้แก่ dist_matrix_man, eps =1,และ minPts =2

```
dist_matrix_man <- proxy::dist(standardized_data,method = "Manhattan")

Db_cl_man <- dbscan::dbscan(dist_matrix_man,eps = 1, minPts = 2)
Db_cl_man
```

Figure 70 กำหนดระยะห่างและตั้งพารามิเตอร์สำหรับ DBScan

เมื่อทำการประมวลผลจะพบว่าด้วยวิธีการ Manhattan distance จะได้กลุ่มทั้งหมด 3 กลุ่มประกอบไปด้วย 2,2 และ 2ตามลำดับและ มี Noise point ทั้งหมด 161 ตัว

```
DBSCAN clustering for 167 objects.
Parameters: eps = 1, minPts = 2
Using Manhattan distances and borderpoints = TRUE
The clustering contains 3 cluster(s) and 161 noise points.

  0   1   2   3
161   2   2   2
```

Figure 71 ข้อมูลของการจัดกลุ่ม

จากนั้นทำการ plot dbscan ด้วยกลุ่มทั้งหมด

```
plot(standardized_data, col = Db_cl_man$cluster)
```

Figure 72 การ plot ข้อมูล

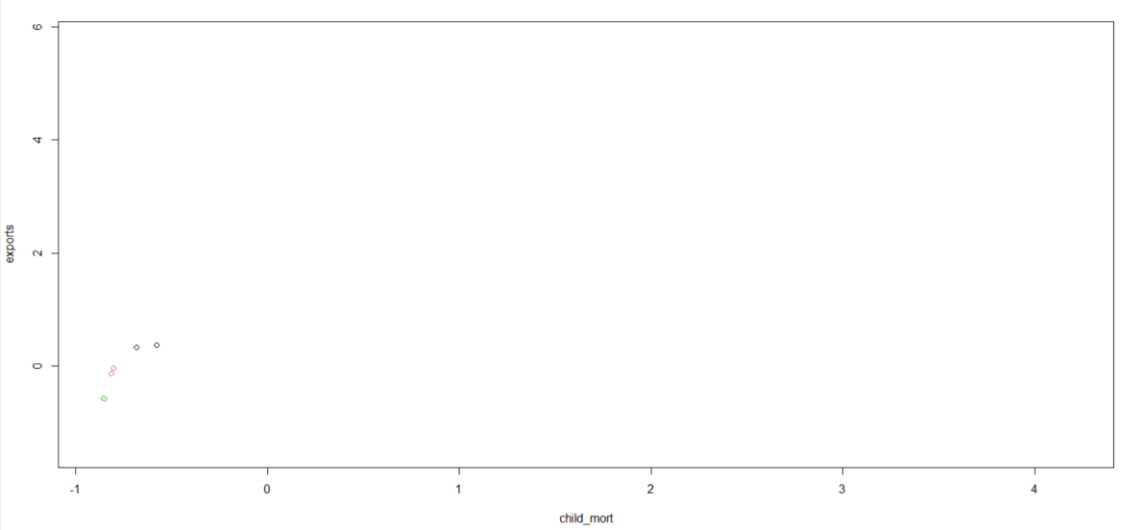

Figure 73 กราฟแสดงผลจาก DBScan

สรุปทั้งหมดของการทำ DBScan ด้วยวิธีการ Manhattan Distance ผ่านข้อมูลที่ยังไม่ผ่านกระบวนการ PCA พบว่าถ้ากำหนด eps = 1 และ minPts = 2 จะได้ทั้งหมด 3 กลุ่มประกอบไปด้วย 2,2 และ 2 ตามลำดับและมี Noise point ทั้งหมด 161 ตัว

ข้อมูลที่ผ่านกระบวนการ PCA แล้ว

ขั้นตอนแรกของ DBScan คือ กำหนดระยะห่างเป็น Manhattan distance โดยใช้คำสั่ง proxy และใช้ข้อมูลที่ผ่านกระบวนการ PCA แล้วและเก็บไว้ใน dist_matrix_pca_man จากนั้นทำการหา DBscan ด้วยการกำหนด parameter 3 ตัวได้แก่ dist_matrix_pca_man, eps =1,และ minPts =2

```
dist_matrix_pca_man <- proxy::dist(pca_components,method = "Manhattan")

Db_cl_pca_man <- dbscan::dbscan(dist_matrix_pca_man,eps = 1, minPts = 2)
Db_cl_pca_man
```

Figure 74 กำหนดระยะห่างและตั้งพารามิเตอร์สำหรับ DBScan

เมื่อทำการประมวลผลจะพบว่าด้วยวิธีการ Euclidean distance จะได้กลุ่มทั้งหมด 16 กลุ่มประกอบไปด้วย 18,44,3,5,3,2,2,4,2,2,2,2,2,6,2 และ 2 ตามลำดับและ มี Noise point ทั้งหมด 66 ตัว

```
DBSCAN clustering for 167 objects.
Parameters: eps = 1, minPts = 2
Using Manhattan distances and borderpoints = TRUE
The clustering contains 16 cluster(s) and 66 noise points.

 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
66 18 44  3  5  3  2  2  4  2  2  2  2  2  6  2  2
```

Figure 75 ข้อมูลของการจัดกลุ่ม

จากนั้นทำการ plot dbscan ด้วยกลุ่มทั้งหมด

```
plot(pca_components, col = Db_cl_pca_man$cluster)
```

Figure 76 การ plot ข้อมูล

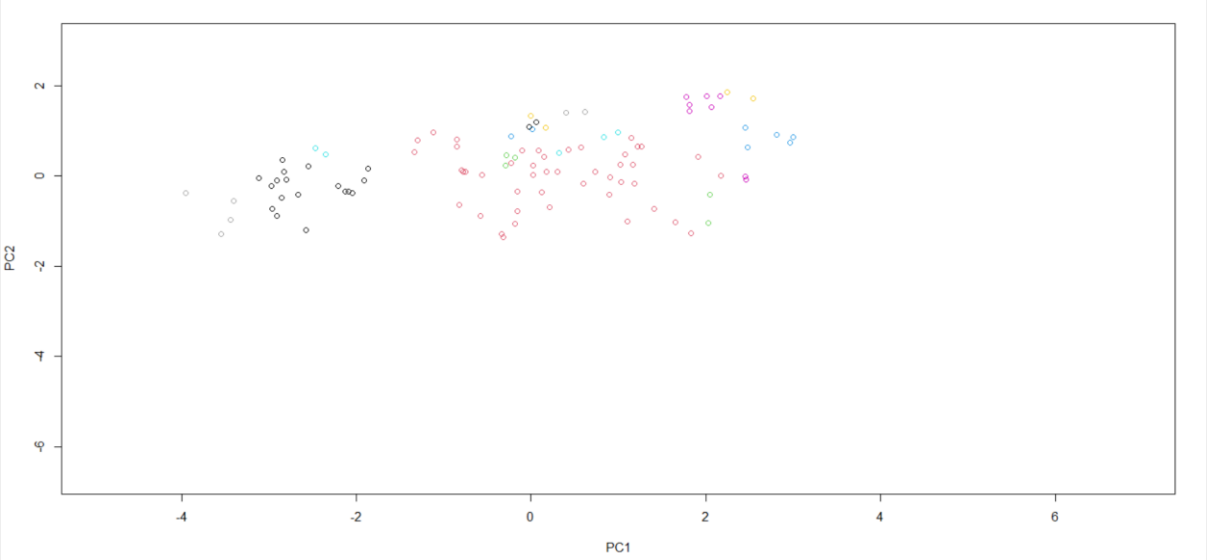

*Figure 77 กราฟแสดงผลจาก **DBScan***

สรุปทั้งหมดของการทำ DBScan ด้วยวิธีการ Manhattan Distance ผ่านข้อมูลที่ผ่านกระบวนการ PCA แล้วพบว่า ถ้ากำหนด eps = 1 และ minPts = 2 จะได้ทั้งหมด 12 กลุ่มประกอบไปด้วย 18,44,3,5,3,2,2,4,2,2,2,2,2,6,2 และ 2 ตามลำดับและ มี Noise point ทั้งหมด 66 ตัว

การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA กับข้อมูลที่ผ่านกระบวนการ PCA แล้วจากการทำ Clustering ด้วย DBScan ด้วยวิธีการ Manhattan Distance

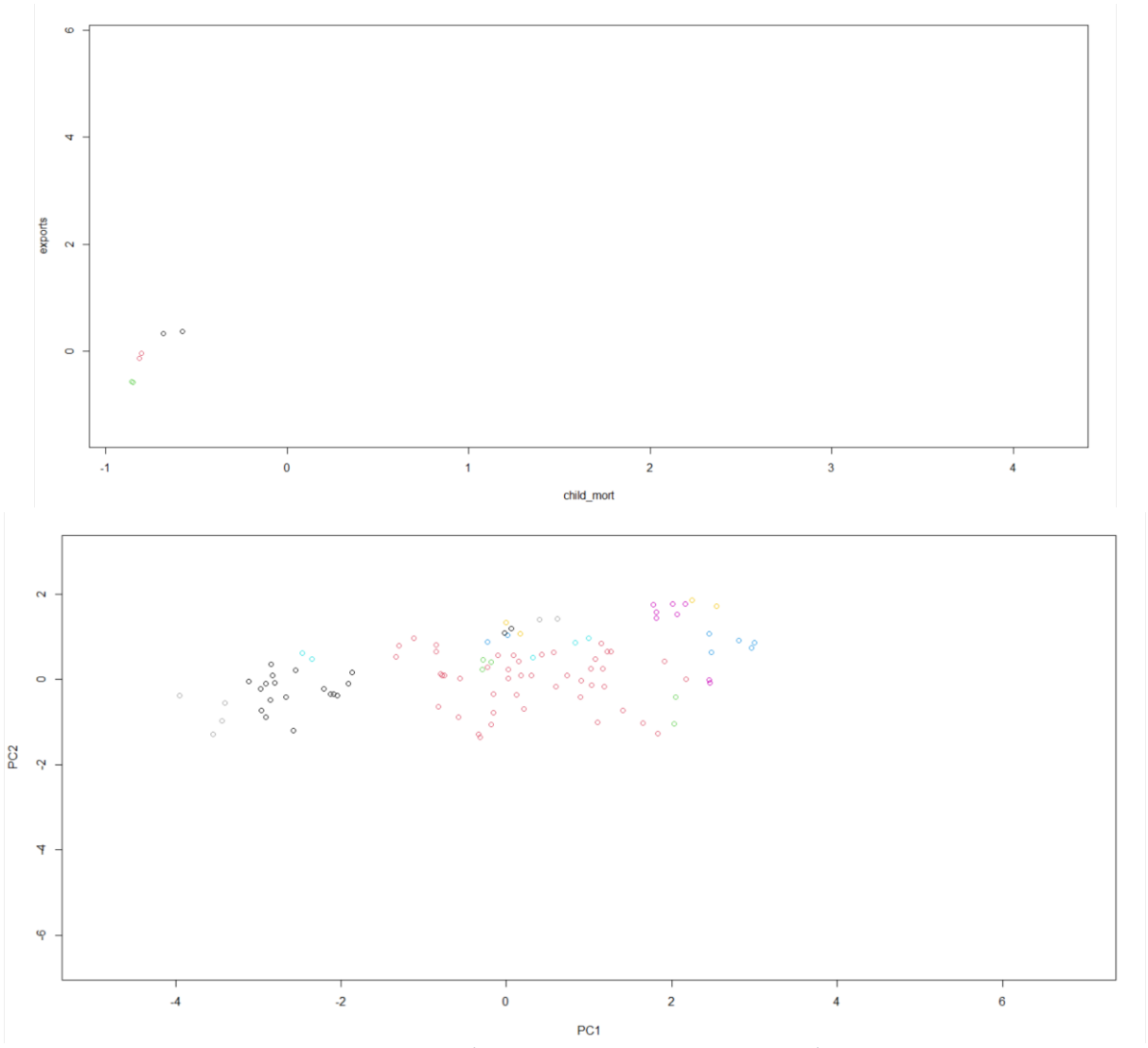

Figure 78 การเปรียบเทียบระหว่างข้อมูลที่ไม่ผ่านกระบวนการ PCA (ด้านบน)กับข้อมูลที่ผ่าน PCA แล้ว(ด้านล่าง)

จากผลการเปรียบเทียบด้วย Visualization โดยกำหนดที่ eps = 1 และ Minpts = 2 พบว่าข้อมูลทั้งกลุ่มนั้นแตกต่างกันในการจัดกลุ่มทั้งนี้ถ้าหากวัดการจับกลุ่มข้อมูลที่ยังไม่ผ่านกระบวนการ PCA ทำได้ดีกว่าด้วยจำนวนกลุ่มนี้น้อยกว่า แต่ถ้าเปรียบเทียบค่า noise point ข้อมูลที่ผ่านกระบวนการ PCA แล้วทำได้ดีกว่าด้วยจำนวน noise point ที่น้อยกว่าจำนวน 66 ตัว

อ้างอิง

geeksforgeeks.org. (2020). **DBScan Clustering in R Programming.** สืบค้นเมื่อ 27 กันยายน 2566 ,จาก https://www.geeksforgeeks.org/dbscan-clustering-in-r-programming/

geeksforgeeks.org. (2020). **K-Means Clustering in R Programming.** สืบค้นเมื่อ 27 กันยายน 2566 , จาก https://www.geeksforgeeks.org/k-means-clustering-in-r-programming/

geeksforgeeks.org. (2021). **K Hierarchical Clustering in R Programming.** สืบค้นเมื่อ 27 กันยายน 2566, จาก https://www.geeksforgeeks.org/hierarchical-clustering-in-r-programming/

Kaggle.(2020). **Unsupervised Learning on Country Data.** สืบค้นเมื่อ 27 กันยายน 2566,จาก https://www.kaggle.com/datasets/rohan0301/unsupervised-learning-on-country-data

Mathematics Learning Support Centre. (2007). **Cluster Analysis**. สืบค้นเมื่อ 27 กันยายน 2566 , จาก https://www.statstutor.ac.uk/resources/uploaded/clusteranalysis.pdf