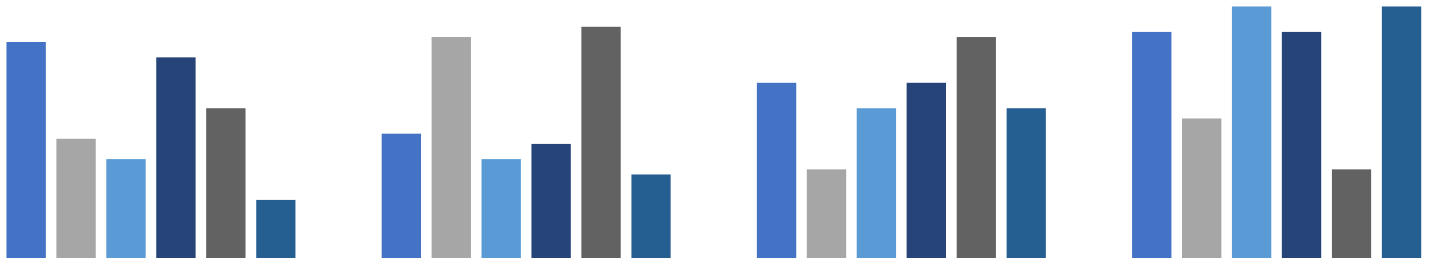
การจัด portfolio ของหุ้นใน SET 50 ด้วย k-means และหาน้ำหนักที่เหมาะสมด้วย modern portfolio theory

เกี่ยวกับงานวิจัย

งานวิจัยนี้จัด portfolio ของหุ้นตามทฤษฎีพอรตการลงทุนสมัยใหม่ (Modern Portfolio Theory) เป็นแนวคิดการกระจายความเสี่ยงโดยถือครองหุ้นเป็นกลุ่ม (Portfolio)แทนที่จะทุ่มในหุ้นเพียงตัวเดียว และลงทุนในหุ้นที่ไม่ค่อยสัมพันธกันมาจัดพอรต ทำให้เมื่อเกิดเหตุการณอะไรขึ้น สินทรัพยหนึ่งราคาอาจจะตก แต่อีกสินทรัพยหนึ่งราคาอาจจะขึ้น ทำให้ผลตอบแทนพอรตโดยรวมไม่ผันผวนมากนัก โดยการเลือกหุ้นมาจัดพอรตในงานวิจัยนี้จะใช้ k-means clustering ในการแบ่งหุ้นเป็นกลุ่ม 7 กลุ่ม หุ้นที่อยู่ในกลุ่มเดียวกันจะมีความคล้ายกันหรือมีความสัมพันธการมาก หุ้นที่อยู่ต่างกลุ่มกันจะมีความคล้ายหรือความสัมพันธกันน้อย จากนั้นจะเลือกหุ้น 1 ตัวในแต่ละกลุ่ม โดยใช้ Sharpe ratio เป็นเกณฑ์ เนื่องจาก ค่า shape ratio เป็นการมอง “ผลตอบแทน” ต่อ 1 หน่วยความเสี่ยงที่เท่ากัน การที่หุ้นตัวหนึ่งมีค่า Sharpe Ratio ที่สูงกว่า เมื่อเทียบกับอีกตัวหนึ่ง แสดงว่า หุ้นตัวนั้นสามารถสร้างผลตอบแทนได้สูงกว่า ณ 1 หน่วย ความเสี่ยงที่เท่ากัน ดังนั้นเราจะเลือกหุ้นที่มีค่า sharpe ratio มากที่สุดเป็นตัวแทนของหุ้นในแต่ละกลุ่ม และเนื่องจากหลักการเลือกหุ้นมาจัดพอรตที่ต้องเลือกหุ้นที่มีความสัมพันธภายในกลุ่มกันเองต่ำ ดังนั้นเราจึงเลือกแค่ 1 หุ้นที่มีค่า sharpe ratio สูงสุดในแต่ละกลุ่มทั้ง 7 กลุ่มมาจัดพอรตหาสัดส่วนน้ำหนักที่เหมาะสมด้วยการสุ่ม 100,000 ครั้ง เพื่อหาสัดส่วนน้ำหนักของพอรตที่มีความเสี่ยงต่ำสุด และหาสัดส่วนน้ำหนักของพอรตที่มีค่า sharpe ratio สูงสุด

SET50



ขั้นตอนการวิจัยหลัก 3 ขั้นตอน

1) การจัดหุ้นเป็นกลุ่ม 7 กลุ่มด้วยวิธี k-means clustering

2) การหาน้ำหนักที่เหมาะสมของ portfolio ตามทฤษฎีพอรตการลงทุนสมัยใหม่ (Modern Portfolio Theory)

3) การเปรียบเทียบอัตราผลตอบแทนสะสม (cumulative return) ของ portfolio ที่มีน้ำหนักที่เหมาะสมตามทฤษฎีพอรตการลงทุนสมัยใหม่ กับที่มีน้ำหนักการลงทุนเท่ากัน (Equally weight portfolio) เทียบกับอัตราผลตอบแทนสะสม (cumulative return) ของ set index

1) การจัดหุ้นเป็นกลุ่ม 7 กลุ่มด้วยวิธี k-means clustering

K-means clustering

- เป็นเทคนิคทาง machine learning แบบ unsupervised learning คือ ไม่มีตัวแปรเป้าหมายในการแบ่งกลุ่มข้อมูล
- จัดกลุ่มตามจำนวน k ที่เรากำหนด โดยใช้จุดศูนยกลางของกลุ่ม (centroid) ที่สุ่มมา กับ ระยะห่างระหว่างข้อมูลในการจัดข้อมูลที่ใกล้กับจุดศูนยกลางนั้นเข้าไปเป็นสมาชิกในกลุ่ม
- โดยข้อมูลที่อยู่ในกลุ่มเดียวกันจะมีความคล้ายกันมาก และข้อมูลที่อยู่คนละกลุ่มจะมีความคล้ายกันน้อย

ขั้นตอนที่ 1) เลือกหุ้นจาก SET50 ทั้งหมด 41 ตัว

ADVANC.BK	AOT.BK	BANPU.BK
BBL.BK	BDMS.BK	BEM.BK
BH.BK	BLA.BK	BTS.BK
CBG.BK	CPALL.BK	CPF.BK
CPN.BK	DTAC.BK	EA.BK
EGCO.BK	GLOBAL.BK	GPSC.BK
HMPRO.BK	INTUCH.BK	IRPC.BK
IVL.BK	JMART.BK	JMT.BK
KBANK.BK	KCE.BK	KTB.BK
KTC.BK	LH.BK	MINT.BK
MTC.BK	PTT.BK	PTTEP.BK
PTTGC.BK	SAWAD.BK	SCC.BK
TISCO.BK	TOP.BK	TRUE.BK
TTB.BK	TU.BK	

SET50

ขั้นตอนที่ 2) หาผลตอบแทนของหุ้น 41 ตัว และ ผลตอบแทนของตลาด (SET) แล้วนำไปเก็บไว้ในตารางเดียวกัน

ดึงข้อมูลราคาปิดของหุ้น 41 ตัว ในปี 2021 จำนวน 1 ปี เพื่อใช้ในการหาอัตราผลตอบแทนของหุ้นแต่ละตัว

```
> head(stocks_return)
# A tibble: 6 x 3
# Groups:   symbol [1]
  symbol    date       stock_ret
  <chr>     <date>         <dbl>
1 ADVANC.BK 2021-01-04   0
2 ADVANC.BK 2021-01-05   0.0142
3 ADVANC.BK 2021-01-06   0.00560
4 ADVANC.BK 2021-01-07   0.00557
5 ADVANC.BK 2021-01-08   0.0222
6 ADVANC.BK 2021-01-11  -0.0136
```

ดึงข้อมูลราคาปิดของหุ้นทั้งหมดในตลาด ในปี 2021 เพื่อใช้หาอัตราผลตอบแทนของตลาด (SET Index)

```
> head(set_index_return)
# A tibble: 6 x 2
  date          set_ret
  <date>          <dbl>
1 2021-01-04  0
2 2021-01-05  0.0262
3 2021-01-06 -0.00948
4 2021-01-07  0.0144
5 2021-01-08  0.0150
6 2021-01-11  0.0000326
```

Joining ข้อมูลระหว่าง stocks_return และ set_index_return ให้อยู่ในตารางเดียวกัน

```
> head(return_data)
# A tibble: 6 x 4
# Groups:   symbol [1]
  symbol    date       stock_ret    set_ret
  <chr>     <date>         <dbl>      <dbl>
1 ADVANC.BK 2021-01-04   0          0
2 ADVANC.BK 2021-01-05   0.0142     0.0262
3 ADVANC.BK 2021-01-06   0.00560   -0.00948
4 ADVANC.BK 2021-01-07   0.00557    0.0144
5 ADVANC.BK 2021-01-08   0.0222     0.0150
6 ADVANC.BK 2021-01-11  -0.0136     0.0000326
```

ขั้นตอนที่ 3) นำข้อมูลผลตอบแทนของหุ้น 41 ตัว และ ผลตอบแทนของตลาด (SET) มาหาค่า Beta

นำข้อมูลอัตราผลตอบแทนของหุ้น(return) กับ อัตราผลตอบแทนของตลาด(market return) มาเข้าโมเดล CAPM

return = risk free + beta×(market return – risk free)

เพื่อหาค่า Beta ของหุ้นทั้ง 41 ตัว ซึ่งค่า Beta เป็นตัวดัชนีวัดความเสี่ยงที่เป็นระบบ ที่บอกถึง “ความรุนแรงและทิศทาง” การเปลี่ยนแปลงของอัตราผลตอบแทนของหลักทรัพย์ใด ๆ เปรียบเทียบกับอัตราการเปลี่ยนแปลงผลตอบแทนของดัชนีตลาดโดยรวมอย่าง SET Index โดยในงานวิจัยไม่มีการคิดผลของ risk free เนื่องจากไม่สามารถหาข้อมูลมาใช้ในการคำนวณได้

```
> head(return_data)
# A tibble: 6 × 4
# Groups:   symbol [1]
  symbol    date       stock_ret    set_ret
  <chr>     <date>         <dbl>      <dbl>
1 ADVANC.BK 2021-01-04   0          0
2 ADVANC.BK 2021-01-05   0.0142     0.0262
3 ADVANC.BK 2021-01-06   0.00560   -0.00948
4 ADVANC.BK 2021-01-07   0.00557    0.0144
5 ADVANC.BK 2021-01-08   0.0222     0.0150
6 ADVANC.BK 2021-01-11  -0.0136     0.0000326
```

ได้ Beta ของหุ้นทั้ง 41 ตัวมา

	symbol	Beta		symbol	Beta
1	ADVANC.BK	0.5921	21	IRPC.BK	0.9597
2	AOT.BK	1.3492	22	IVL.BK	1.4243
3	BANPU.BK	0.7524	23	JMART.BK	1.5056
4	BBL.BK	1.2572	24	JMT.BK	1.1019
5	BDMS.BK	0.6008	25	KBANK.BK	1.5906
6	BEM.BK	0.9207	26	KCE.BK	0.4890
7	BH.BK	0.7972	27	KTB.BK	1.3486
8	BLA.BK	0.6026	28	KTC.BK	1.7452
9	BTS.BK	1.0127	29	LH.BK	0.7527
10	CBG.BK	0.6249	30	MINT.BK	1.3638
21	IRPC.BK	0.9597	31	MTC.BK	1.1625
22	IVL.BK	1.4243	32	PTT.BK	0.9774
23	JMART.BK	1.5056	33	PTTEP.BK	0.8924
24	JMT.BK	1.1019	34	PTTGC.BK	1.1407
25	KBANK.BK	1.5906	35	SAWAD.BK	1.3138
26	KCE.BK	0.4890	36	SCC.BK	0.6335
27	KTB.BK	1.3486	37	TISCO.BK	0.5248
28	KTC.BK	1.7452	38	TOP.BK	1.2654
29	LH.BK	0.7527	39	TRUE.BK	0.9972
30	MINT.BK	1.3638	40	TTB.BK	1.3824
			41	TU.BK	0.4467

ขั้นตอนที่ 4) หาผลตอบแทนเฉลี่ยของหุ้น 41 ตัว และ นำไปรวมกับ ค่า Beta กับ อัตราส่วนทางการเงิน

สร้าง ผลตอบแทนเฉลี่ย (Stock mean return) ของหุ้นแต่ละตัว แล้วนำไป

- join กับค่า Beta ที่ได้มาจากสมการ CAPM
- join กับ Financial ratio 5 ตัว ได้แก่
  - อัตราส่วนหนี้สินต่อทุน(Debt to Equity Ratio)
  - อัตราส่วนหมุนเวียนทรัพยสิน(Assets Turnover Ratio)
  - อัตราส่วนผลตอบแทนต่อสินทรัพยรวม(Return on Assets)
  - ราคาหุ้นต่อมูลค่าหุ้นทางบัญชี(Price to Book Value)
  - อัตราส่วนเงินปันผลตอบแทน (Dividend Yield)

```r
# Join mean of stock return data and financial ratio of each stock
dataset <-
  # Calculating mean of stock return
  stocks_return %>%
  group_by(symbol) %>%
  summarize(mean_return = mean(stock_ret)) %>%
  # Inner join with beta_capm
  inner_join(beta_capm,
             by = "symbol") %>%
  # Inner join with financial ratio
  inner_join(read.csv(paste0(path,"data/financial_ratio.csv")),
             by = "symbol")
```

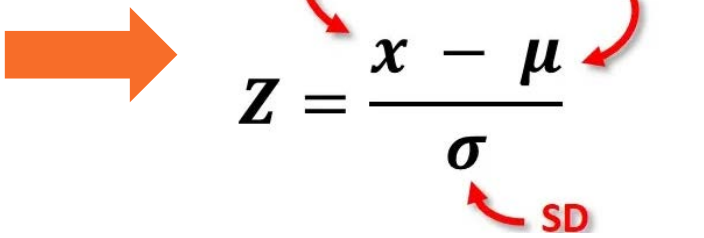
	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	ADVANC.BK	0.00135	0.592	3.36	0.52	34.2	9.15	3.01
2	AOT.BK	0.0000731	1.35	0.73	0.04	-12.8	7.76	0
3	BANPU.BK	0.000388	0.752	3.31	0.42	13.9	1.04	2.12
4	BBL.BK	0.000265	1.26	7.79	0.04	5.63	0.48	2.06
5	BDMS.BK	0.000643	0.601	0.49	0.57	9.24	4.53	2.39
6	BEM.BK	0.000188	0.921	2.05	0.1	2.67	3.41	1.18

ขั้นตอนที่ 5) Normalizing dataset by Z – score normalization

ข้อมูลที่จะใช้ใน k-means ต้องเป็นข้อมูลที่อยู่ในสเกลเดียวกัน ถ้าไม่ยังนั้นจะทำให้ตัวแปรแต่ละตัวมีน้ำหนักในการคำนวณระยะห่างไม่เท่ากัน

Dataset

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	ADVANC.BK	0.00135	0.592	3.36	0.52	34.2	9.15	3.01
2	AOT.BK	0.0000731	1.35	0.73	0.04	-12.8	7.76	0
3	BANPU.BK	0.000388	0.752	3.31	0.42	13.9	1.04	2.12
4	BBL.BK	0.000265	1.26	7.79	0.04	5.63	0.48	2.06
5	BDMS.BK	0.000643	0.601	0.49	0.57	9.24	4.53	2.39
6	BEM.BK	0.000188	0.921	2.05	0.1	2.67	3.41	1.18



ได้ข้อมูลที่ถูกทำให้เป็นค่ามาตรฐานโดยค่าเฉลี่ยของข้อมูลที่ถูกทำให้เป็นค่ามาตราฐานจะมีค่าเฉลี่ยเท่ากับ 0 และส่วนเบี่ยงเบียนมาตรฐานเท่ากับ 1

```
> head(scaled_df)
           mean_return       Beta debt_to_equity asset_turnover        roe        pbv   dvd_yield
ADVANC.BK   0.3663054 -1.1652551      0.2357267     0.21790727  2.1549650  1.3195809  0.49217449
AOT.BK     -0.8216977  0.9862064     -0.8372327    -1.09755745 -2.4890419  0.9541635 -1.44713758
BANPU.BK   -0.5287558 -0.7097284      0.2153282    -0.05614788  0.1487461 -0.8124592 -0.08124336
BBL.BK     -0.6431010  0.7247687      2.0430309    -1.09755745 -0.6683465 -0.9596777 -0.11990075
BDMS.BK    -0.2910966 -1.1405322     -0.9351454     0.35493484 -0.3121020  0.1050279  0.09271486
BEM.BK     -0.7142528 -0.2314680     -0.2987132    -0.93312436 -0.9604472 -0.1894092 -0.68687570
```

ขั้นตอนที่ 6) หาค่า k หรือจำนวนกลุ่มที่เหมาะสม โดยใช้วิธี Elbow Method

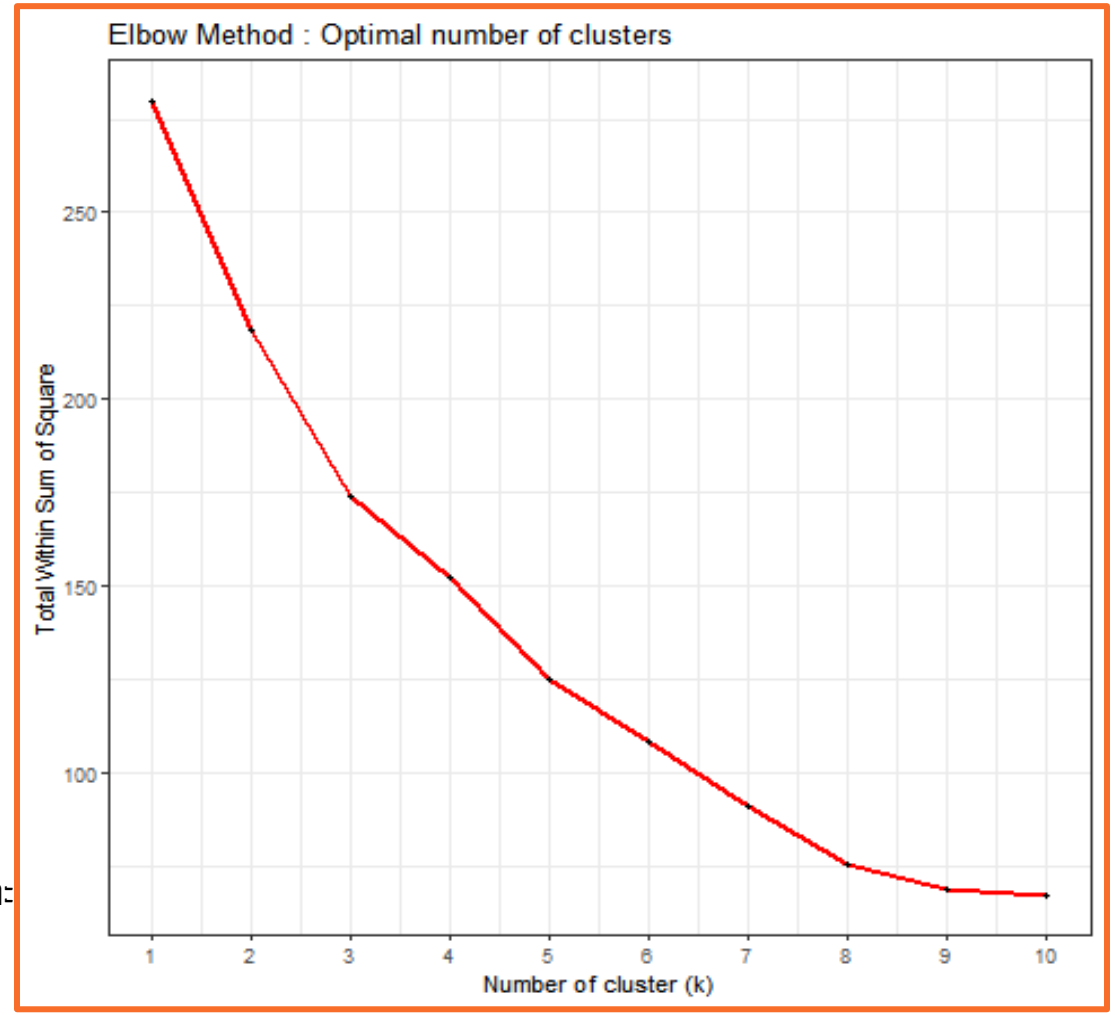
```r
# Elbow method for find best k
elbow_df <- data.frame(
  k = 1:10,
  tot_withiness =
    map_dbl(1:10, function(k){
      model <- kmeans(x=scaled_df, center = k)
      model$tot.withinss
    }))
elbow_df

png(file=paste0(path, "graph/elbow_plot.png"))
ggplot(data = elbow_df) +
  geom_line(mapping = aes(x = k, y = tot_withiness), color = "red", size=1) +
  geom_point(mapping = aes(x = k, y = tot_withiness), size = 1) +
  scale_x_continuous(breaks = 1:10) +
  theme_bw() +
  labs(x = "Number of cluster (k)",
       y = "Total Within Sum of Square",
       title = "Elbow Method : Optimal number of clusters")
dev.off()
```

create elbow_df

สร้าง Dataframe ที่มีข้อมูล Total Within Sum of Square ของ k ตั้งแต่ 1 ถึง 10

Plot elbow method

นำ elbow_df มา plot line graph ด้วย ggplot2 และ export graph ดูจากกราฟนี้จะได้ k = 3



ขั้นตอนที่ 6) หาค่า k หรือจำนวนกลุ่มที่เหมาะสม โดยใช้วิธี Silhouette Method

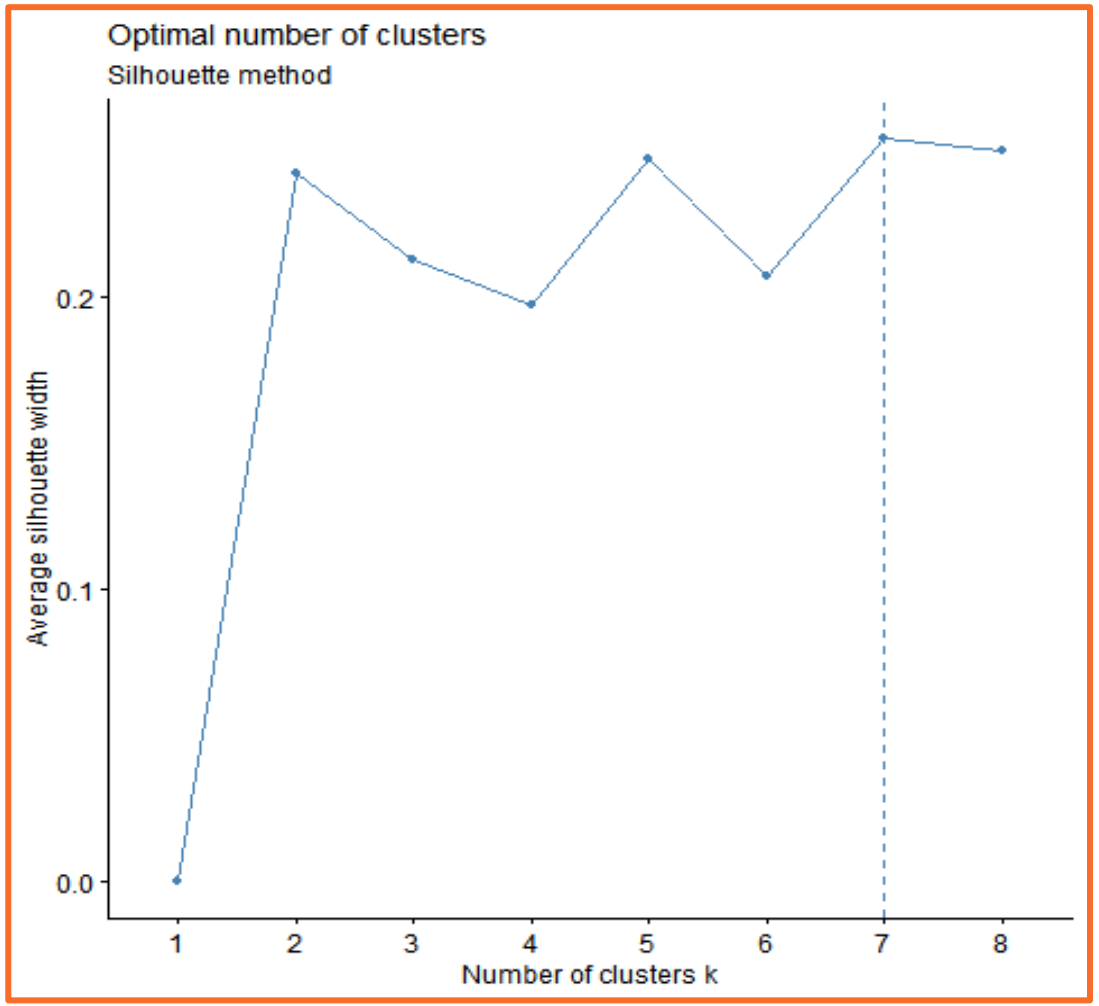
```r
# Silhouette method for find best k
png(file=paste0(path, "graph/silhouette_plot.png"))
fviz_nbclust(scaled_df,
             FUN = kmeans,
             k.max = 8,
             method = "silhouette") +
  labs(subtitle = "Silhouette method")
dev.off()
```

Plot Silhouette method

Plot silhouette method ด้วย library factoextra และ export graph

What is optimal k?

ถ้าสร้างกราฟ silhouette method ด้วย library นี้ตัวฟังกชันจะเลือก k มาให้ที่ k = 7 ซึ่งเป็นการหาค่า k ที่ชัดเจนมากกว่าวิธี Elbow method ดังนั้นจึงเลือกค่า k=7



ขั้นตอนที่ 7) modeling k-means

```r
# k-means modeling
set.seed(123) # set seed
km <- kmeans(scaled_df, centers=7)
km
```

K-Means Clustering Model

- set seed เพื่อให้กลุ่มที่สุ่มออกมาได้ ค่าที่สุ่มชุดเดิมทุกครั้ง และไม่ให้เลขของกลุ่มเปลี่ยนไปเปลี่ยนมา

- ใช้ฟังกชัน kmeans ของ library stat เพื่อจัดกลุ่มข้อมูลทั้ง 7

จะได้ Result หลักๆ ออกมาเป็น

Cluster means : ค่าเฉลี่ยที่ group by ด้วยกลุ่ม

Clustering vector : แสดงว่าแต่ละหุ้นอยู่กลุ่มไหน

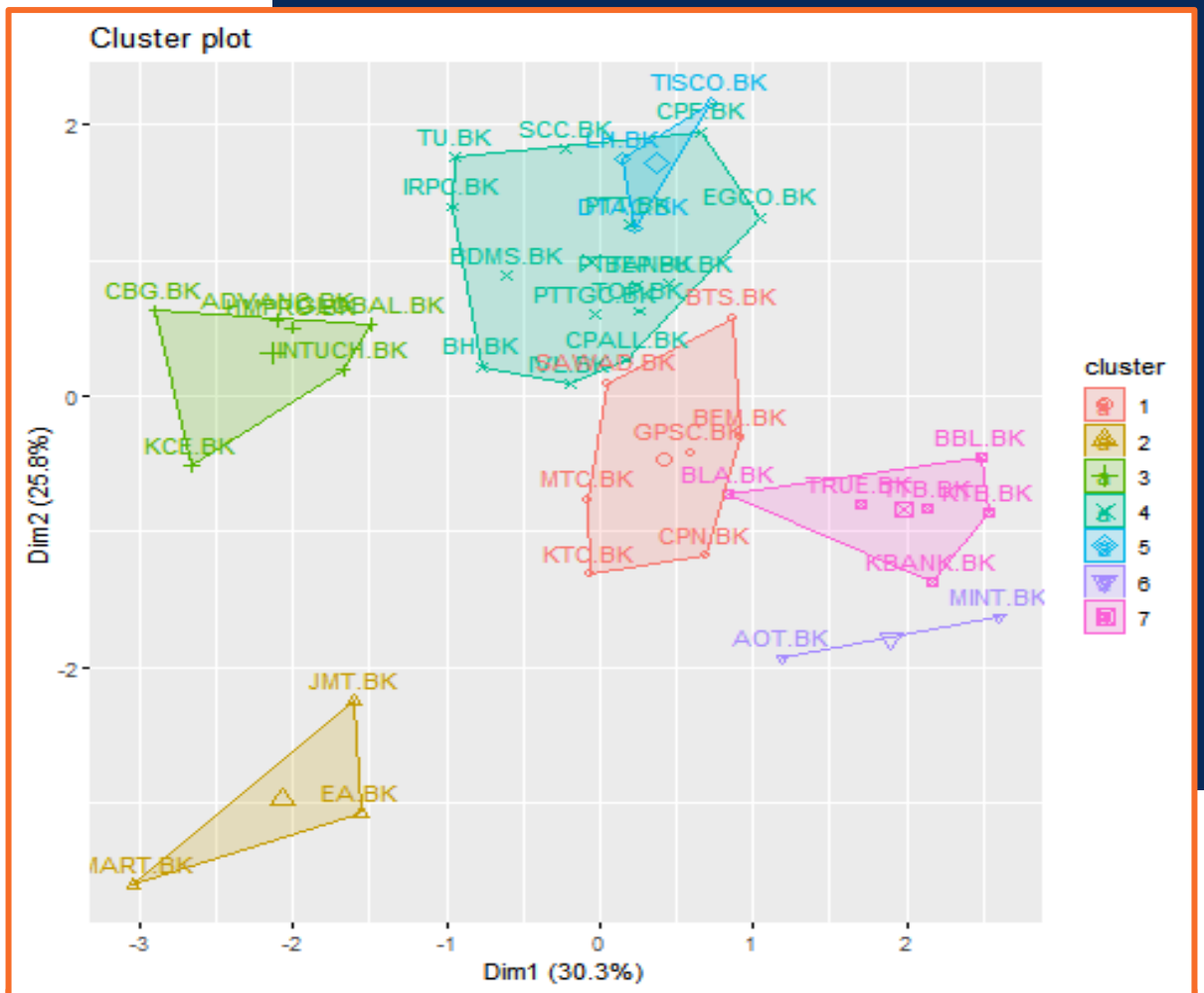
```
> km
K-means clustering with 7 clusters of sizes 7, 3, 6, 14, 3, 2, 6

Cluster means:
  mean_return       Beta debt_to_equity asset_turnover         roe        pbv  dvd_yield
1  -0.5600045  0.7216428     -0.2952163     -0.6708144  0.09024136 -0.1019043 -0.3012223
2   2.3587214  1.1218508     -0.7148419     -0.5951230  0.58656824  2.2291813 -1.0090206
3   0.4078446 -0.9627830     -0.6169293      0.7842879  1.27603123  1.1272329 -0.1574843
4  -0.4987577 -0.3654954     -0.3765188      0.7660176 -0.06377411 -0.5411189  0.1994829
5   0.1253681 -0.9499480      0.7076112     -0.5220416  0.28262559 -0.2481214  2.5861161
6  -0.6625458  1.0069509     -0.1151270     -0.8646106 -2.81913551  0.2995667 -1.4471376
7   0.3880658  0.5520872      1.8818830     -0.9422595 -0.72739164 -0.8361193 -0.2627183

Clustering vector:
ADVANC.BK    AOT.BK  BANPU.BK    BBL.BK   BDMS.BK    BEM.BK     BH.BK    BLA.BK    BTS.BK    CBG.BK
        3         6         4         7         4         1         4         7         1         3
 CPALL.BK    CPF.BK    CPN.BK   DTAC.BK     EA.BK   EGCO.BK GLOBAL.BK   GPSC.BK  HMPRO.BK INTUCH.BK
        4         4         1         5         2         4         3         1         3         3
  IRPC.BK    IVL.BK  JMART.BK   JMT.BK  KBANK.BK    KCE.BK    KTB.BK    KTC.BK     LH.BK  MINT.BK
        4         4         2         2         7         3         7         1         5         6
   MTC.BK    PTT.BK  PTTEP.BK  PTTGC.BK  SAWAD.BK    SCC.BK  TISCO.BK    TOP.BK   TRUE.BK    TTB.BK
        1         4         4         4         1         4         5         4         7         7
    TU.BK
        4

Within cluster sum of squares by cluster:
[1] 12.062580  4.780206 18.876348 36.469370  5.032415  2.277839 10.392982
 (between_SS / total_SS =  67.9 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
```

Cluster plot

• ใช้ฟังกชัน fviz_cluster() จาก library factoextra โดยไส้ในมาจากการใช้ ggplot2 ในการสร้างกราฟ

• เนื่องจากข้อมูลที่ใส่เข้าไปในโมเดล k-means มีหลายมิติ ในฟังกชันจะใช้การลดมิติด้วย k-means เพื่อลดให้เหลือ 2 มิติแล้วค่อย plot ออกมาเป็น graph ดังรูป

• export graph to .png




```
# plot graph
png(file=paste0(path, "graph/cluster_scatter_plot.png"))
fviz_cluster(km, data = scaled_df)
dev.off()
```

ขั้นตอนที่ 8) เก็บ km.cluster ที่ระบุจำนวนกลุ่ม

```r
# Create cluster group dataframe and mutate symbol by row names
km_df <- data.frame(km$cluster) %>%
  rownames_to_column("symbol")

# Join with dataset
cluster <- dataset %>%
  inner_join(km_df, by = "symbol")
# Write cluster to csv
cluster %>% write.csv(paste0(path, "data/all_cluster.csv"),
                      row.names = FALSE)
```

- เก็บเลขที่จัดกลุ่มของหุ้น (km.cluster) และที่มีข้อมูล ratio ต่างๆของแต่ละหุ้น ไว้ใน dataframe ที่เป็น cluster
- write cluster to .csv

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	ADVANC.BK	0.00135	0.592	3.36	0.52	34.2	9.15	3.01	3
2	AOT.BK	0.0000731	1.35	0.73	0.04	-12.8	7.76	0	6
3	BANPU.BK	0.000388	0.752	3.31	0.42	13.9	1.04	2.12	4
4	BBL.BK	0.000265	1.26	7.79	0.04	5.63	0.48	2.06	7
5	BDMS.BK	0.000643	0.601	0.49	0.57	9.24	4.53	2.39	4
6	BEM.BK	0.000188	0.921	2.05	0.1	2.67	3.41	1.18	1
7	BH.BK	0.000884	0.797	0.19	0.56	6.79	6.75	2.27	4
8	BLA.BK	0.00250	0.603	6.24	0.14	6.8	1.34	0.62	7
9	BTS.BK	0.000236	1.01	2.26	0.22	7.95	2.06	3.31	1
10	CBG.BK	0.000399	0.625	0.89	0.97	28.6	12.4	2.01	3

… with 31 more rows

Result

k-means clustering

จัดกลุ่มได้ข้อมูลออกมาทั้งหมด 7 cluster ได้แก่

Cluster 1 : มีหุ้นทั้งหมด 7 ตัวคือ

BEM, BTS, CPN, GPSC,

KTC, MTC, SAWAD

Cluster 2 : มีหุ้นทั้งหมด 3 ตัวคือ

EA, JMART, JMT

Cluster 3 : มีหุ้นทั้งหมด 6 ตัวคือ

ADVANC, CBG, GLOBAL, HMPRO,

INTUCH, KCE

Result

k-means clustering

จัดกลุ่มได้ข้อมูลออกมาทั้งหมด 7 cluster ได้แก่

Cluster 4 : มีหุ้นทั้งหมด 14 ตัวคือ

BANPU, BDMS, BH, CPALL, CPF, EGCO,

IRPC, IVL, PTT, PTTEP, PTTGC, SCC, TOP, TU

Cluster 5 : มีหุ้นทั้งหมด 3 ตัวคือ

DTAC, LH, TISCO

Result

k-means clustering

จัดกลุ่มได้ข้อมูลออกมาทั้งหมด 7 cluster ได้แก่

Cluster 6 : มีหุ้นทั้งหมด 2 ตัวคือ

AOT, MINT

Cluster 7 : มีหุ้นทั้งหมด 6 ตัวคือ

BBL, BLA, KBANK, KTB, TRUE, TTB

ลักษณะของข้อมูล Cluster 1

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	BEM.BK	0.000188	0.921	2.05	0.1	2.67	3.41	1.18	1
2	BTS.BK	0.000236	1.01	2.26	0.22	7.95	2.06	3.31	1
3	CPN.BK	0.000886	1.41	2.48	0.13	10.1	3.54	1.24	1
4	GPSC.BK	0.000704	1.23	1.41	0.3	6.93	2.37	1.69	1
5	KTC.BK	0.000233	1.75	2.32	0.24	23.7	5.98	1.49	1
6	MTC.BK	0.000215	1.16	2.96	0.18	21.7	5.24	0.63	1
7	SAWAD.BK	0.0000168	1.31	0.93	0.2	20.2	3.6	2.91	1

ลักษณะของข้อมูล Cluster 2

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	EA.BK	0.00276	1.58	1.55	0.15	20.2	11.6	0.31	2
2	JMART.BK	0.00469	1.51	1.07	0.35	23.5	15.1	0.82	2
3	JMT.BK	0.00302	1.10	0.47	0.17	11.3	11.1	0.91	2

ลักษณะของข้อมูล Cluster 3

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	ADVANC.BK	0.00135	0.592	3.36	0.52	34.2	9.15	3.01	3
2	CBG.BK	0.000399	0.625	0.89	0.97	28.6	12.4	2.01	3
3	GLOBAL.BK	0.00108	0.643	0.99	0.91	18.3	4.89	0.89	3
4	HMPRO.BK	0.000379	0.989	1.56	1.11	24.5	9.06	2.07	3
5	INTUCH.BK	0.00180	0.642	0.18	0.07	27.3	6.77	3.12	3
6	KCE.BK	0.00336	0.489	0.64	0.78	19.1	8.2	0.91	3

ลักษณะของข้อมูล Cluster 5

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	DTAC.BK	0.00193	0.726	7.15	0.4	15.1	5.42	6.54	5
2	LH.BK	0.000679	0.753	1.49	0.27	13.9	2.19	5.68	5
3	TISCO.BK	0.000662	0.525	4.91	0.08	16.8	1.95	6.56	5

ลักษณะของข้อมูล Cluster 4

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	BANPU.BK	0.000388	0.752	3.31	0.42	13.9	1.04	2.12	4
2	BDMS.BK	0.000643	0.601	0.49	0.57	9.24	4.53	2.39	4
3	BH.BK	0.000884	0.797	0.19	0.56	6.79	6.75	2.27	4
4	CPALL.BK	0.000212	0.913	6.14	0.81	12.9	5.52	1.53	4
5	CPF.BK	-0.000230	0.676	2.58	0.65	6.4	1.02	3.94	4
6	EGCO.BK	-0.000156	0.750	1.13	0.17	3.81	0.82	3.7	4
7	IRPC.BK	0.000373	0.960	1.17	1.41	17.8	0.92	1.56	4
8	IVL.BK	0.000654	1.42	2.34	0.95	18.4	1.56	1.62	4
9	PTT.BK	-0.000165	0.977	1.59	0.81	11.5	1.11	2.63	4
10	PTTEP.BK	0.00103	0.892	0.89	0.32	10.1	1.14	3.6	4
11	PTTGC.BK	0.000255	1.14	1.34	0.79	15.0	0.85	1.69	4
12	SCC.BK	0.000336	0.634	1.12	0.67	13.7	1.29	3.63	4
13	TOP.BK	-0.000158	1.27	1.98	1.04	10.6	0.87	1.41	4
14	TU.BK	0.00181	0.447	1.76	0.91	14.4	1.59	3.69	4

ลักษณะของข้อมูล Cluster 6

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	AOT.BK	0.0000731	1.35	0.73	0.04	-12.8	7.76	0	6
2	MINT.BK	0.000415	1.36	4.27	0.21	-19.5	2.78	0	6

ลักษณะของข้อมูล Cluster 7

	symbol	mean_return	Beta	debt_to_equity	asset_turnover	roe	pbv	dvd_yield	km.cluster
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	BBL.BK	0.000265	1.26	7.79	0.04	5.63	0.48	2.06	7
2	BLA.BK	0.00250	0.603	6.24	0.14	6.8	1.34	0.62	7
3	KBANK.BK	0.00125	1.59	7.48	0.07	8.3	0.72	1.76	7
4	KTB.BK	0.000901	1.35	8.84	0.04	6.14	0.52	2.08	7
5	TRUE.BK	0.00165	0.997	6.68	0.24	-1.72	1.96	1.46	7
6	TTB.BK	0.00167	1.38	7.34	0.05	5.04	0.68	3.05	7

2) การหาน้ำหนักที่เหมาะสมของ portfolio ตามทฤษฎีพอรตการลงทุนสมัยใหม่ (Modern Portfolio Theory)

- ดึงข้อมูลราคาและอัตราผลตอบแทนของหุ้น ใน set50 ทั้ง 41 ตัวมาคำนวณ Sharpe Ratio ของแต่ละหุ้น

- นำไฟลที่ชื่อ all_cluster ที่มีข้อมูลของ ratio และชื่อหุ้นทั้งหมดรวมอยู่

- นำไฟลมา join กันด้วย symbol

- นำข้อมูลดังกล่าวมา loop ตั้งแต่ 1 : k โดย k คือจำนวน cluster

- ในแต่ละครั้งที่วน loop จะหาค่าที่ Sharpe ratio ที่มากที่สุดของแต่ละ cluster แล้วนำข้อมูลที่มีแค่ symbol และ Sharpe Ratio มาต่อท้ายตาราง

- ผลที่ได้จะเป็น portfolio ใหม่ที่มีหุ้นจากแต่ละ cluster ที่มี Sharpe ratio มากที่สุด

```r
# Function for get Maximum Sharpe Ratios of each stock in each cluster (Risk free rate = 0%)
symbol_optimize_func <- function() {

  # Get Sharpe Ratio
  stocks_sharpe_ratio <- tq_get(symbols_set50,
                                from = date_from,
                                to = date_to) %>%
    na.omit() %>%
    group_by(symbol) %>%
    tq_transmute(select = adjusted,
                 mutate_fun = periodReturn,
                 period = "daily",
                 col_rename = "stock_ret") %>%
    tq_performance(Ra = stock_ret,
                   performance_fun = SharpeRatio.annualized) %>%
    rename("sharpe_ratio" = "AnnualizedSharpeRatio(Rf=0%)") %>%
    left_join(read_csv(paste0(path, "data/all_cluster.csv")),
              by = "symbol")

  # Set number of cluster
  k = max(read_csv(paste0(path,"data/all_cluster.csv"))$km.cluster) %>%
    as.numeric()

  # create empty dataframes
  symbol_max_sr_each_cluster <- data.frame(symbol = character(),
                                           sharpe_ratio = integer())
  # loop for get maximum Sharpe ratio of each cluster and append to empty tibble
  for (i in 1:k) {
    x <- stocks_sharpe_ratio %>%
      filter(km.cluster==i)
    x <- stocks_sharpe_ratio %>%
      filter(sharpe_ratio==max(x$sharpe_ratio)) %>%
      select(symbol, sharpe_ratio)
    symbol_max_sr_each_cluster <- bind_rows(symbol_max_sr_each_cluster, x)
  }

  # return symbol_max_sr_each_cluster
  return(symbol_max_sr_each_cluster)
}
```

```r
# Get symbol of portfolio for optimization
portfolio_optimization <- symbol_optimize_func() %>% as.data.frame()
portfolio_optimization %>% write_csv(paste0(path, "data/portfolio_optimization.csv"))
symbol_for_optimization <- portfolio_optimization[["symbol"]]
```

	symbol	sharpe_ratio
1	GPSC.BK	0.91114429
2	JMT.BK	1.35010739
3	GLOBAL.BK	0.62946091
4	PTTEP.BK	0.52478992
5	TISCO.BK	0.93799038
6	AOT.BK	0.41185030
7	KTB.BK	0.08869616

	symbol
1	GPSC.BK
2	JMT.BK
3	GLOBAL.BK
4	PTTEP.BK
5	TISCO.BK
6	AOT.BK
7	KTB.BK

Export to csv
เพื่อสำหรับใช้ symbol
ในส่วนต่อไป

Create function for Optimization portfolio

```r
# create function for optimization portfolio
optimize_function <- function(symbol,
                              num_loop) {⟺}

optimal_portfolio <-
  optimize_function(symbol = symbol_for_optimization,
                    num_loop = 100000)
```

Create function for optimization portfolio

ข้อมูลที่จะใช้ในการ optimization
ทั้งหมดจะใช้ข้อมูลทั้งหมด 5 ปี
- ตั้งแต่ วันที่ 1 เดือน 1 ปี 2016
- จนถึง วันที่ 31 เดือน 12 ปี 2021

```
# Set date
date_from = "2016-01-01"
date_to = "2021-12-31"
```

สร้างตัวแปรที่เก็บ return ของหุ้นใน portfolio

```
# Storing return of stock
stock_daily_returns <- symbol %>%
  tq_get(from = date_from,
         to = date_to) %>%
  group_by(symbol) %>%
  tq_transmute(select = adjusted,
               mutate_fun = periodReturn,
               period = "daily",
               col_rename = "stock_ret")
```

นำข้อมูล return มาคำนวณ Geometric mean return
สำหรับหุ้นแต่ละตัวใน portfolio โดยมีที่มาของสูตรดังรูป

$$GAR = \sqrt[n]{(1+r1) * (1+r2) * \ldots . (1+rn)} - 1$$

```
# Calculating geometric mean return of each stocks
geometric_mean_ret <- stock_daily_returns %>%
  tq_performance(Ra = stock_ret,
                 performance_fun = Return.annualized,
                 scale = 252)
```

Mean Return นี้จะถูก normalized ด้วย 252 เนื่องจากใช้ข้อมูลรายวัน

	symbol	AnnualizedReturn
1	GPSC.BK	0.33299233
2	JMT.BK	0.57832803
3	GLOBAL.BK	0.22122867
4	PTTEP.BK	0.19331310
5	TISCO.BK	0.23720778
6	AOT.BK	0.11750486
7	KTB.BK	0.02216795

Create function for optimization portfolio

- นำตัวแปรที่เก็บ return ของหุ้นใน portfolio มาคำนวณ Covariance Matrix ของหุ้นแต่ละตัว
- โดยจะถูก normalize ด้วย 252 เหมือนกัน ก่อนที่จะไปหา SD

```
# Calculating covariance matrix of each stocks
cov_matrix <- stock_daily_returns %>%
  spread(symbol, value = stock_ret) %>%
  select(!(date)) %>%
  cov() * 252
```

	AOT.BK	GLOBAL.BK	GPSC.BK	JMT.BK	KTB.BK	PTTEP.BK	TISCO.BK
AOT.BK	0.08140157	0.02869625	0.03789574	0.02714438	0.03424002	0.04398976	0.02660555
GLOBAL.BK	0.02869625	0.12352222	0.03435445	0.04435113	0.02518193	0.03115597	0.02172752
GPSC.BK	0.03789574	0.03435445	0.13356546	0.04035892	0.02735997	0.04093419	0.02213127
JMT.BK	0.02714438	0.04435113	0.04035892	0.18348966	0.02431935	0.02756682	0.01880395
KTB.BK	0.03424002	0.02518193	0.02735997	0.02431935	0.06246569	0.04471312	0.02706909
PTTEP.BK	0.04398976	0.03115597	0.04093419	0.02756682	0.04471312	0.13569120	0.02884749
TISCO.BK	0.02660555	0.02172752	0.02213127	0.01880395	0.02706909	0.02884749	0.06395301

สูตรที่คำนวณ Covariance Matrix

$$\begin{bmatrix} Var(x_1) & \cdots\cdots & Cov(x_n,x_1) \\ \vdots & \cdot & \vdots \\ \vdots & \cdot & \vdots \\ Cov(x_n,x_1) & \cdots\cdots & Var(x_n) \end{bmatrix}$$

Population Covariance Formula

$$Cov(x,y) = \frac{\Sigma(x_i-\bar{x})(y_i-\bar{y})}{N}$$

Sample Covariance

$$Cov(x,y) = \frac{\Sigma(x_i-\bar{x})(y_i-y)}{N-1}$$

Create function for optimization portfolio

```r
# Define number of loop
num_port <- num_loop

# Creating a matrix to store the weights
all_wts <- matrix(nrow = num_port,
                  ncol = length(symbol))

# Creating an empty vector to store
# Portfolio returns
port_returns <- vector("numeric", length = num_port)

# Creating an empty vector to store
# Portfolio Standard deviations
port_risk <- vector("numeric", length = num_port)

# Creating an empty vector to store
# Portfolio Sharpe Ratios
sharpe_ratio <- vector("numeric", length = num_port)
```

- กำหนดจำนวนที่จะ loop
- สร้าง matrix ที่จะเก็บ weight ทั้งหมดเอาไว้
- สร้าง vector ที่มีข้อมูลประเภท numeric
  - สำหรับเก็บ return ของ portfolio
  - สำหรับเก็บ risk ของ portfolio
  - สำหรับเก็บ Sharpe ratio ของ portfolio

Create function for optimization portfolio

```r
# Loop storing weight, return, standard deviation and Sharpe ratio
for (i in seq_along(port_returns)) {

  # random weight and adjusted sum weight = 1
  wts <- runif(length(symbol))
  wts <- wts/sum(wts)
  # Storing weight in the matrix
  all_wts[i,] <- wts

  # Calculating portfolio annualized return
  port_ret <- sum(wts * geometric_mean_ret$AnnualizedReturn)
  # Storing portfolio annualized returns
  port_returns[i] <- port_ret

  # Calculating portfolio risk (standard deviation)
  port_sd <- sqrt(t(wts) %*% (cov_matrix  %*% wts))
  # Storing portfolio risks (standard deviation)
  port_risk[i] <- port_sd

  # Calculating portfolio Sharpe ratio
  port_sr <- port_ret / port_sd
  # Storing portfolio Sharpe ratio (Assuming 0% Risk free rate)
  sharpe_ratio[i] <- port_sr
}
```

Loop สำหรับเก็บค่า weight, return, sd และ Sharpe Ratio

ในการ Loop แต่ละครั้งสิ่งที่เกิดขึ้น คือ

- wts คือ weight ที่ถูกสุ่มขึ้นมาตามจำนวนหุ้น และปรับให้ผลรวมน้ำหนัก = 1
- เก็บค่า wts ไว้ใน all_wts โดยจะใส่ไว้ในแต่ละแถวของ matrix

- port_ret คือ คำนวณ return ของ portfolio จาก Geometric mean return ที่คำนวณมาของแต่ละหุ้น แล้วคูณด้วยน้ำหนักของแต่ละหุ้นที่สุ่มมา
- จากนั้นก็ฝากค่า port_ret ไว้ใน vector port_returns

$$\text{Expected value} = R = \sum_{i=1}^{n} (R_i, P_i)$$

Where,

R_i = the return associated with each outcome

P_i = the probability of occurrence of each outcome

R = the expected value.

Create function for optimization portfolio

```r
# Loop storing weight, return, standard deviation and Sharpe ratio
for (i in seq_along(port_returns)) {

  # random weight and adjusted sum weight = 1
  wts <- runif(length(symbol))
  wts <- wts/sum(wts)
  # Storing weight in the matrix
  all_wts[i,] <- wts

  # Calculating portfolio annualized return
  port_ret <- sum(wts * geometric_mean_ret$AnnualizedReturn)
  # Storing portfolio annualized returns
  port_returns[i] <- port_ret

  # Calculating portfolio risk (standard deviation)
  port_sd <- sqrt(t(wts) %*% (cov_matrix  %*% wts))
  # Storing portfolio risks (standard deviation)
  port_risk[i] <- port_sd

  # Calculating portfolio Sharpe ratio
  port_sr <- port_ret / port_sd
  # Storing portfolio Sharpe ratio (Assuming 0% Risk free rate)
  sharpe_ratio[i] <- port_sr
}
```

Loop สำหรับเก็บค่า weight, return, sd และ Sharpe Ratio

ในการ Loop แต่ละครั้งสิ่งที่เกิดขึ้น คือ

- คำนวณ standard deviation ของ portfolio ด้วย cov_matrix และน้ำหนักที่ถูกสุ่มมาในแต่ละรอบ
- จากนั้นก็ฝากค่า port_sd ไว้ใน vector port_risk

$$V = \sigma_p^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j \mathrm{Cov}(r_i, r_j)$$

$$\sigma_p^2 = W^T S(W)$$

$$\sigma_p = \sqrt{W^T S(W)} = \left[\begin{bmatrix} w_1 & \cdots & w_j \end{bmatrix} \begin{bmatrix} \sigma_{11}^2 & \cdots & \mathrm{cov}_{1j} \\ \vdots & \ddots & \vdots \\ \mathrm{cov}_{j1} & \cdots & \sigma_{jj}^2 \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_j \end{bmatrix}\right]^{\frac{1}{2}}$$

Create function for optimization portfolio

```r
# Loop storing weight, return, standard deviation and Sharpe ratio
for (i in seq_along(port_returns)) {

  # random weight and adjusted sum weight = 1
  wts <- runif(length(symbol))
  wts <- wts/sum(wts)
  # Storing weight in the matrix
  all_wts[i,] <- wts

  # Calculating portfolio annualized return
  port_ret <- sum(wts * geometric_mean_ret$AnnualizedReturn)
  # Storing portfolio annualized returns
  port_returns[i] <- port_ret

  # Calculating portfolio risk (standard deviation)
  port_sd <- sqrt(t(wts) %*% (cov_matrix  %*% wts))
  # Storing portfolio risks (standard deviation)
  port_risk[i] <- port_sd

  # Calculating portfolio Sharpe ratio
  port_sr <- port_ret / port_sd
  # Storing portfolio Sharpe ratio (Assuming 0% Risk free rate)
  sharpe_ratio[i] <- port_sr
}
```

Loop สำหรับเก็บค่า weight, return, sd และ Sharpe Ratio

ในการ Loop แต่ละครั้งสิ่งที่เกิดขึ้น คือ

- คำนวณ Sharpe Ratio ของ portfolio ด้วย portfolio return และ
และ portfolio standard deviation โดยให้ Risk Free Rate = 0%
- จากนั้นก็ฝากค่า port_sr ไว้ใน vector sharpe_ratio

$$S_p = \frac{E(r_p) - R_{rf}}{\sigma_p}$$

Create function for optimization portfolio

```r
# Create portfolio values to store returns, risks and Sharpe ratios
portfolio_values <- tibble(Return = port_returns,
                           Risk = port_risk,
                           SharpeRatio = sharpe_ratio)

# Rename columns all weights by symbol
all_wts <- tk_tbl(all_wts)
colnames(all_wts) <- symbol

# Combing all the values together
portfolio_values <- tk_tbl(cbind(all_wts, portfolio_values))
```

- นำข้อมูลที่เก็บมาได้จากการ loop ที่สุ่ม weight โดยจะมีข้อมูล weight, return, risk และ Sharpe ratio
- นำข้อมูลที่ได้มารวมกันและออกมาเป็น dataframe

	GPSC.BK	JMT.BK	GLOBAL.BK	PTTEP.BK	TISCO.BK	AOT.BK	KTB.BK	Return	Risk	SharpeRatio
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	0.150	0.226	0.111	0.165	0.0251	0.262	0.0610	0.275	0.227	1.21
2	0.148	0.119	0.0437	0.165	0.123	0.203	0.198	0.217	0.208	1.04
3	0.160	0.142	0.166	0.141	0.145	0.147	0.0994	0.253	0.210	1.20
4	0.284	0.251	0.253	0.0321	0.0514	0.00312	0.125	0.318	0.219	1.45
5	0.217	0.0446	0.109	0.126	0.104	0.233	0.167	0.202	0.212	0.951
6	0.208	0.126	0.207	0.178	0.153	0.0914	0.0363	0.270	0.217	1.24
7	0.251	0.0900	0.140	0.352	0.0924	0.0628	0.0111	0.264	0.241	1.10
8	0.188	0.0582	0.0579	0.0971	0.213	0.226	0.160	0.208	0.209	0.998
9	0.178	0.275	0.268	0.102	0.0546	0.107	0.0157	0.323	0.229	1.41
10	0.198	0.200	0.162	0.0488	0.166	0.120	0.104	0.283	0.208	1.36

Create function for optimization portfolio

เลือกข้อมูลมาทั้งหมด 2 ชุด

1. ชุดข้อมูลที่มีค่า variance หรือความเสี่ยงต่ำที่สุด
2. ชุดข้อมูลที่มีค่า Sharpe Ratio มากที่สุด

```
# Create min_var to store minimum variance
min_var <- portfolio_values[which.min(portfolio_values$Risk),]
# Create max_sr to store maximum Sharpe ratio
max_sr <- portfolio_values[which.max(portfolio_values$SharpeRatio),]
```

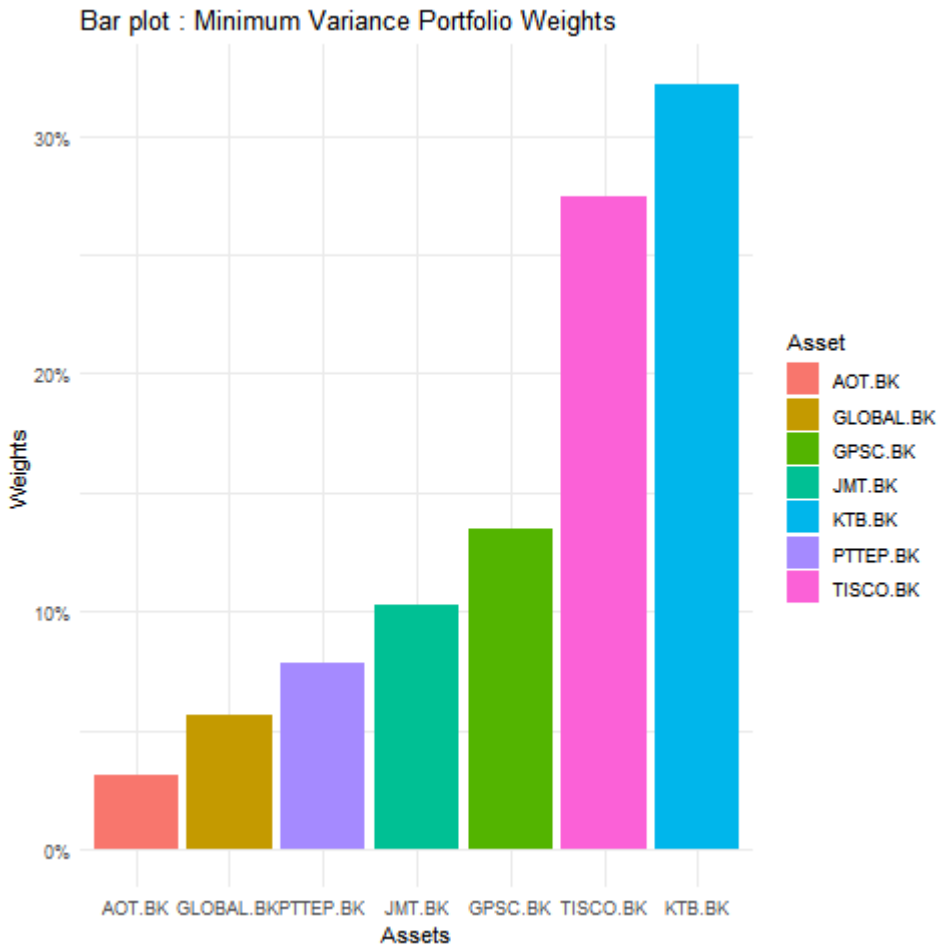
```
> min_var
# A tibble: 1 × 10
  GPSC.BK JMT.BK GLOBAL.BK PTTEP.BK TISCO.BK AOT.BK KTB.BK Return  Risk SharpeRatio
    <dbl>  <dbl>     <dbl>    <dbl>    <dbl>  <dbl>  <dbl>  <dbl> <dbl>       <dbl>
1   0.135  0.103    0.0565   0.0782    0.274 0.0313  0.322  0.208 0.194        1.07
```

```
> max_sr
# A tibble: 1 × 10
  GPSC.BK JMT.BK GLOBAL.BK PTTEP.BK TISCO.BK AOT.BK KTB.BK Return  Risk SharpeRatio
    <dbl>  <dbl>     <dbl>    <dbl>    <dbl>  <dbl>  <dbl>  <dbl> <dbl>       <dbl>
1   0.393  0.431   0.00742   0.0324   0.0587 0.0274 0.0504  0.406 0.236        1.72
```

Create function for optimization portfolio

สร้าง bar plot ที่แสดงค่า weight หรือน้ำหนักของหุ้น
แต่ละตัว โดยจะแสดงน้ำหนักที่ให้ค่าความเสี่ยงน้อยที่สุด

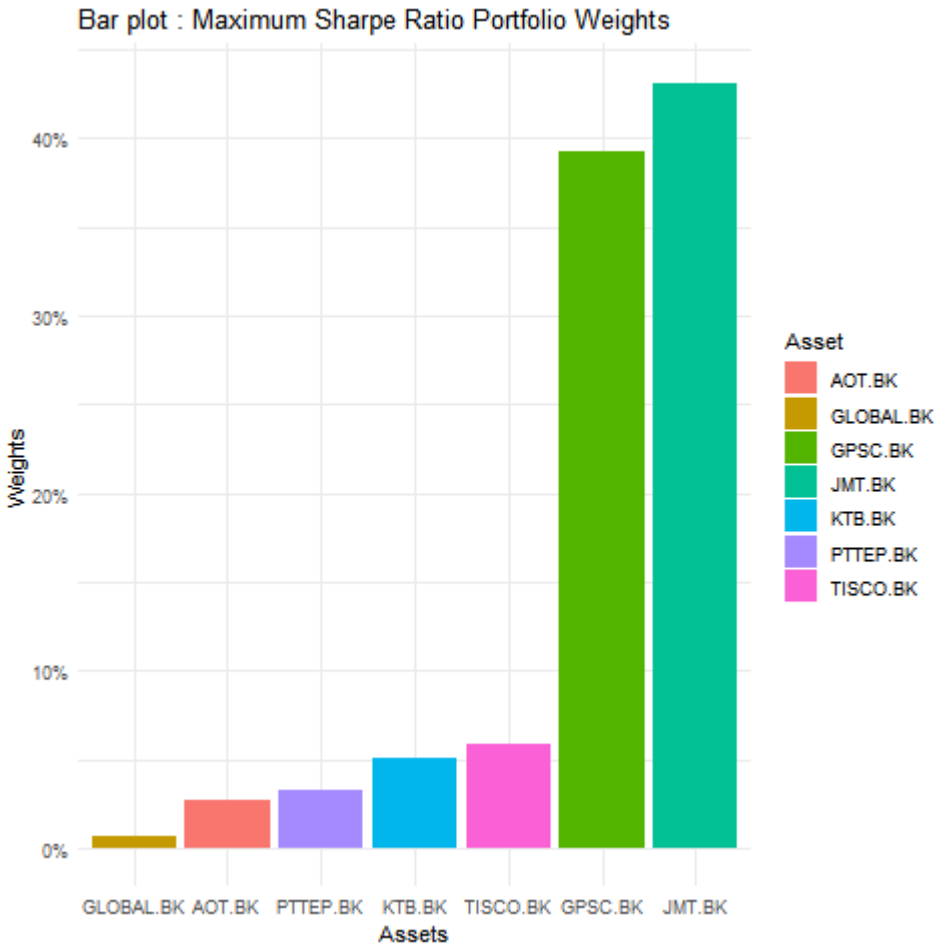
```
# Storing bar plot weight of minimum variance
barplot_wts_min_var <- min_var %>%
  gather(-(Return:SharpeRatio),
         key = Asset,
         value = Weights) %>%
  mutate(Asset = as.factor(Asset)) %>%
  ggplot(aes(x = fct_reorder(Asset,Weights),
             y = Weights, fill = Asset)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(x = "Assets",
       y = "Weights",
       title = "Bar plot : Minimum Variance Portfolio Weights") +
  scale_y_continuous(labels = scales::percent)
```



Create function for optimization portfolio

สร้าง bar plot ที่แสดงค่า weight หรือน้ำหนักของหุ้น
แต่ละตัว โดยจะแสดงน้ำหนักที่ให้ค่า Sharpe ratio มากที่สุด

```
# Storing bar plot weight of maximum Sharpe ratio
barplot_wts_max_sr <- max_sr %>%
  gather(-(Return:SharpeRatio),
         key = Asset,
         value = Weights) %>%
  mutate(Asset = as.factor(Asset)) %>%
  ggplot(aes(x = fct_reorder(Asset,Weights),
             y = Weights, fill = Asset)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(x = "Assets",
       y = "Weights",
       title = "Bar plot : Maximum Sharpe Ratio Portfolio Weights") +
  scale_y_continuous(labels = scales::percent)
```

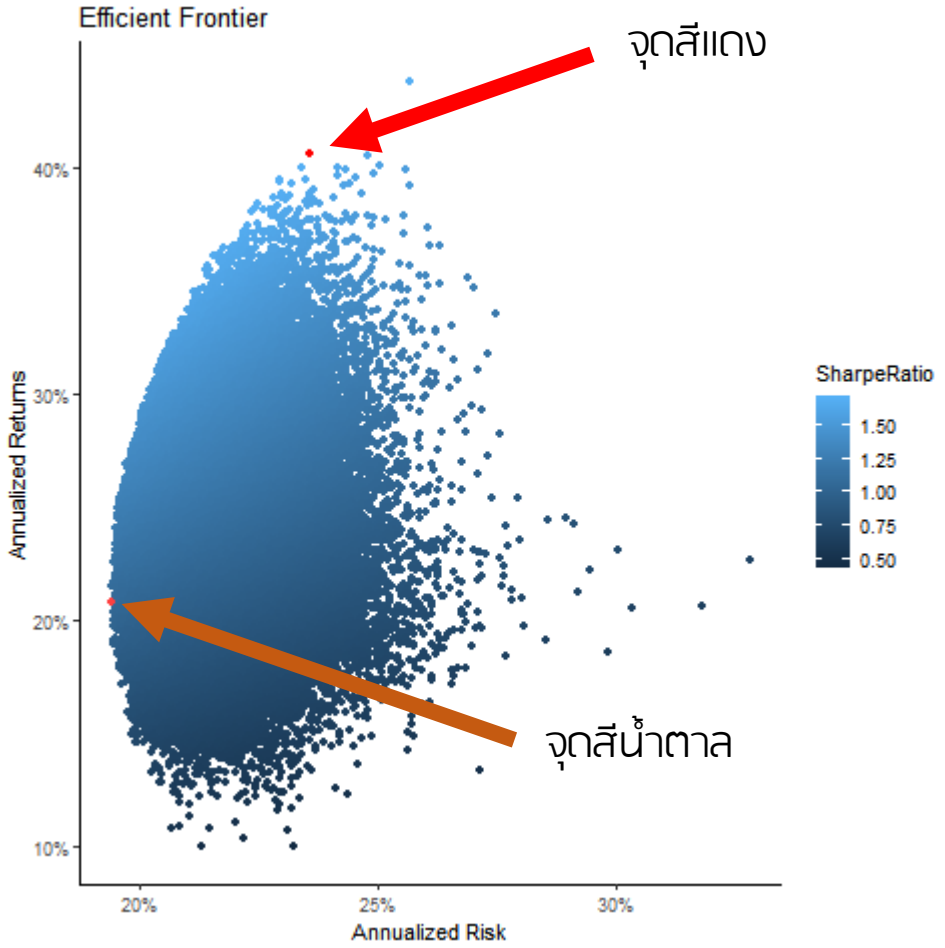


Create function for optimization portfolio

Plot graph Efficient frontier

- เป็นกราฟจุดที่แสดงถึงความสัมพันธระหว่าง return และ risk โดยในกราฟจะกำหนดให้อยู่ในรูปของ %
- เนื่องจากใช้ Loop เพื่อสุ่มน้ำหนัก 100000 ครั้ง ดังนั้นจะมี 100000 จุด
- จุดสีแดงบ่งบอกถึง Maximum Sharpe Ratio
- จุดสีน้ำตาลบอกถึง Minimum Variance

```
# Storing graph Efficient frontier
eff_frontier <- portfolio_values %>%
  ggplot(aes(x = Risk, y = Return, color = SharpeRatio)) +
  geom_point() +
  theme_classic() +
  scale_y_continuous(labels = scales::percent) +
  scale_x_continuous(labels = scales::percent) +
  labs(x = "Annualized Risk",
       y = "Annualized Returns",
       title = "Efficient Frontier") +
  geom_point(aes(x = Risk,
                 y = Return), data = min_var, color = "brown1") +
  geom_point(aes(x = Risk,
                 y = Return), data = max_sr, color = "red")
```



Create function for optimization portfolio

Return ค่าต่าง
เพื่อเก็บผลต่างๆ

```r
# Storing and return result
result <- list(
  portfolio_values = portfolio_values,
  min_var = min_var,
  max_sr = max_sr,
  barplot_wts_min_var = barplot_wts_min_var,
  barplot_wts_max_sr = barplot_wts_max_sr,
  eff_frontier = eff_frontier
)
return(result)
```

- Export ไฟลต่างๆที่สำคัญ to .csv
- Export graph ต่างๆที่สำคัญ to .png

```r
## Save table and graph by loop
# export portfolio values of each portfolio to .csv
port_val <- optimal_portfolio$portfolio_values
port_val %>%
  write_csv(paste0(path, "data/optimize_portfolio_values.csv"))

# export min varaince of each portfolio to .csv
min_var <- optimal_portfolio$min_var
min_var %>%
  write_csv(paste0(path, "data/optimize_portfolio_min_var.csv"))

# export max sharpe ratio of each portfolio to .csv
max_sr <- optimal_portfolio$max_sr
max_sr %>%
  write_csv(paste0(path, "data/optimize_portfolio_max_sr.csv"))
```

```r
# plot and export bar plot of min variance weight to .png
bar_min_var <- optimal_portfolio$barplot_wts_min_var
png(file = paste0(path, "graph/barplot_wts_min_var.png"))
plot(bar_min_var)
dev.off()

# plot and export bar plot of max sharpe ratio weight to .png
bar_max_sr <- optimal_portfolio$barplot_wts_max_sr
png(file = paste0(path, "graph/barplot_wts_max_sr.png"))
plot(bar_max_sr)
dev.off()

# plot and export graph efficient frontier to .png
eff_frontier <- optimal_portfolio$eff_frontier
png(file = paste0(path, "graph/efficient_frontier.png"))
plot(eff_frontier)
dev.off()
```

3) การเปรียบเทียบอัตราผลตอบแทนสะสม (cumulative return) ของ Portfolio ที่มีน้ำหนักที่เหมาะสมตามทฤษฎีพอรตการลงทุนสมัยใหม่ กับที่มีน้ำหนักการลงทุนเท่ากัน(Equally weight portfolio) เทียบกับ อัตราผลตอบแทนสะสม(cumulative return)ของ SET Index

ข้อมูลที่จะนำมาดู cumulative return
จะใช้ข้อมูลทั้งหมด 5 ปี

```
# Set date
date_from = "2016-01-01"
date_to = "2021-12-31"
```

READ CSV

SET DATE

ดึงข้อมูล Maximum Sharpe Ratio

```
# Read optimize_portfolio_max_sr.csv
max_sr <- read_csv(paste0(path, "data/optimize_portfolio_max_sr.csv"))
max_sr
```

ดึงข้อมูล Minimum Variance

```
# Read optimize_portfolio_min_var.csv
min_var <- read_csv(paste0(path, "data/optimize_portfolio_min_var.csv"))
min_var
```

ดึงข้อมูล symbol ของ portfolio เดียวกับที่ใช้ optimization

```
# Get symbol portfolio
portfolio_symbol <- read_csv(paste0(path, "data/portfolio_optimization.csv"))[["symbol"]]
```

Create function for Calculate portfolio growth

```
port_growth_func <- function(port_symbol,
                             port_wts = NULL) {
```

Create function for portfolio growth graph

ดึงข้อมูลราคาและคำนวณ return ของแต่ละหุ้น
จากนั้นให้คำนวณ port growth ด้วยการใช้
Cumulative return โดยให้ wealth.index = TRUE

```
# Get portfolio growth
portfolio_growth <- port_symbol %>%
  tq_get(from = date_from,
         to = date_to) %>%
  na.omit() %>%
  group_by(symbol) %>%
  tq_transmute(select = adjusted,
               mutate_fun= periodReturn,
               period = "daily",
               col_rename = "stock_return") %>%
  tq_portfolio(assets_col = symbol,
               returns_col = stock_return,
               weights = port_wts,
               col_rename= "port_growth",
               wealth.index= TRUE)
```

ดึงข้อมูลราคาและคำนวณ return ของ SET index
จากนั้นให้คำนวณ set growth ด้วยการใช้
Cumulative return โดยให้ wealth.index = TRUE

```
# Get benchmark growth
benchmark_growth <- tq_get("^SET.BK",
                           from = date_from,
                           to = date_to) %>%
  na.omit() %>%
  tq_transmute(select = adjusted,
               mutate_fun= periodReturn,
               period = "daily",
               col_rename= "set_return") %>%
  mutate(symbol="SET.BK") %>%
  tq_portfolio(assets_col= symbol,
               returns_col= set_return,
               col_rename= "set_growth",
               wealth.index= TRUE)
```

Create function for portfolio growth graph

Plot graph portfolio growth

ด้วย portfolio growth และ benchmark growth

```
# Plot comparison graph between portfolio and SET index
port_graph <- ggplot() +
  geom_line(
    data=portfolio_growth,
    mapping=aes(x=date, y=port_growth),
    color="blue") +
  geom_line(
    data=benchmark_growth,
    mapping=aes(x=date, y=set_growth),
    color="red") +
  theme_bw()

return(port_graph)
```

```r
# Create function for get optimal weight
get_opti_wts_func <- function(data) {
  opti_wts_port <- data %>%
    select(-Return, -Risk, -SharpeRatio)
  opti_wts_port <- data.frame(r1=names(opti_wts_port),
                              t(opti_wts_port),
                              row.names = NULL)
  colnames(opti_wts_port) <- c("symbol", "weights")
  return(opti_wts_port)
}
```

สร้างฟังกชันสำหรับดึง weight ที่เป็น optimal มา ภายในฟังกชันจะมี

- การรับข้อมูล และเลือกแค่ column ที่ไม่ใช่ Return, Risk และ Sharpe Ratio
- transpose ข้อมูลโดยเปลี่ยนจาก row เป็น column
- ตั้งชื่อ column ใหม่ ออกมาได้ข้อมูลลักษณะ ดังนี้

```r
# Get optimal weight of maximum Sharpe ratio portfolio
opti_wts_max_sr <- get_opti_wts_func(data = max_sr)
# Get optimal weight of minimum variance portfolio
opti_wts_min_var <- get_opti_wts_func(data = min_var)
```

ลักษณะข้อมูลมีรูปแบบค่อนข้างเฉพาะ จึงต้องดึงด้วยวิธีเฉพาะสำหรับข้อมูลนี้

```
> min_var
# A tibble: 1 × 10
  GPSC.BK JMT.BK GLOBAL.BK PTTEP.BK TISCO.BK AOT.BK KTB.BK Return  Risk SharpeRatio
    <dbl>  <dbl>     <dbl>    <dbl>    <dbl>  <dbl>  <dbl>  <dbl> <dbl>       <dbl>
1   0.135  0.103    0.0565   0.0782    0.274 0.0313  0.322  0.208 0.194        1.07
> max_sr
# A tibble: 1 × 10
  GPSC.BK JMT.BK GLOBAL.BK PTTEP.BK TISCO.BK AOT.BK KTB.BK Return  Risk SharpeRatio
    <dbl>  <dbl>     <dbl>    <dbl>    <dbl>  <dbl>  <dbl>  <dbl> <dbl>       <dbl>
1   0.393  0.431   0.00742   0.0324   0.0587 0.0274 0.0504  0.406 0.236        1.72
```

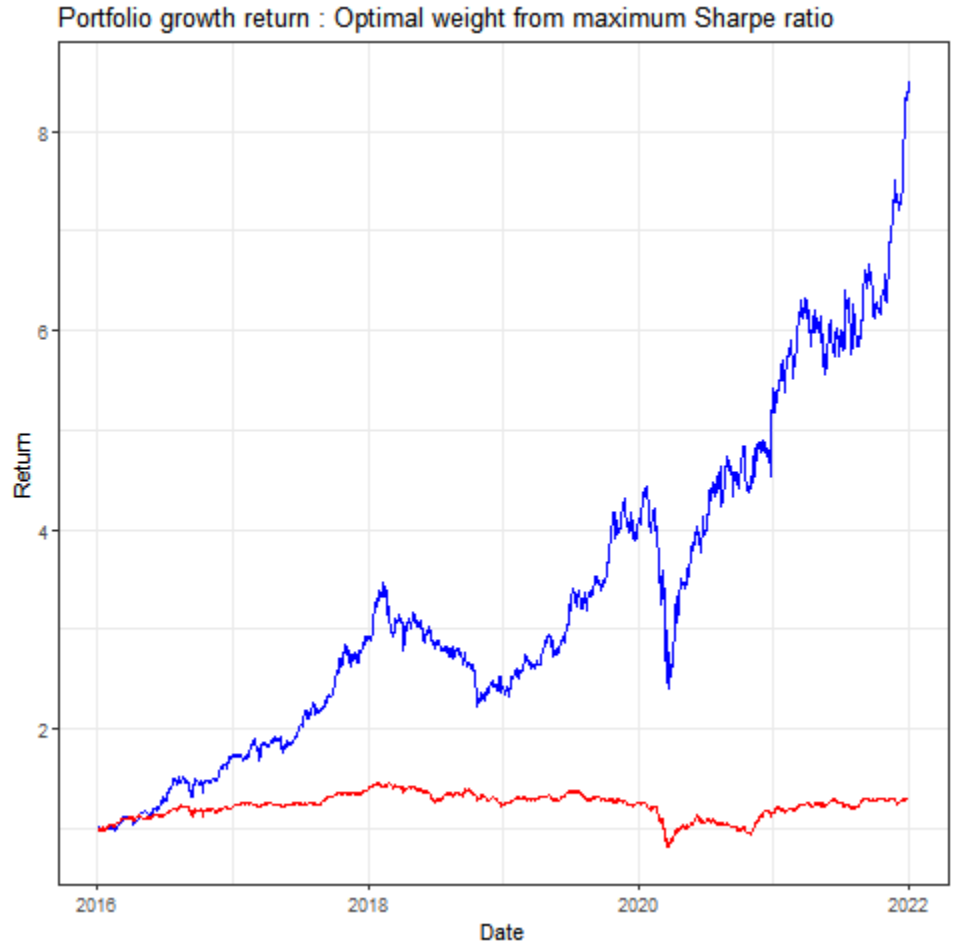
Maximum Sharpe Ratio

	symbol	weights
1	GPSC.BK	0.388200667
2	JMT.BK	0.419201027
3	GLOBAL.BK	0.051276679
4	PTTEP.BK	0.018826896
5	TISCO.BK	0.089448268
6	AOT.BK	0.006687227
7	KTB.BK	0.026359237

Minimum Variance

	symbol	weights
1	GPSC.BK	0.11225883
2	JMT.BK	0.10214355
3	GLOBAL.BK	0.05680652
4	PTTEP.BK	0.09083886
5	TISCO.BK	0.30083989
6	AOT.BK	0.01953357
7	KTB.BK	0.31757877

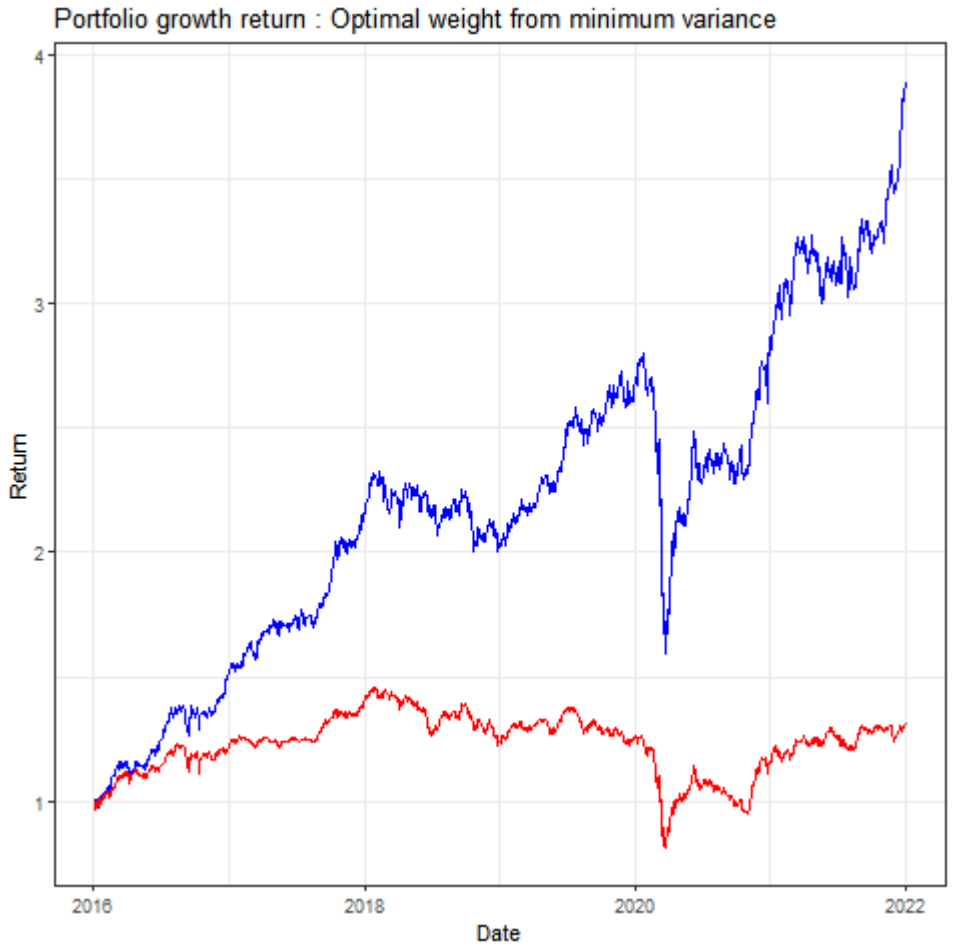
```r
# Use Maximum Sharpe Ratio weights
# Create graph from optimal weight maximum Sharpe ratio portfolio
opti_wts_max_sr_graph <- port_growth_func(port_symbol = portfolio_symbol,
                                          port_wts = opti_wts_max_sr)
png(file=paste0(path,"graph/portgrowth_optimal_weight_max_sr_graph.png"))
plot(opti_wts_max_sr_graph +
       labs(x = "Date",
            y = "Return",
            title = "Portfolio growth return : Optimal weight from maximum Sharpe ratio"))
dev.off()
```



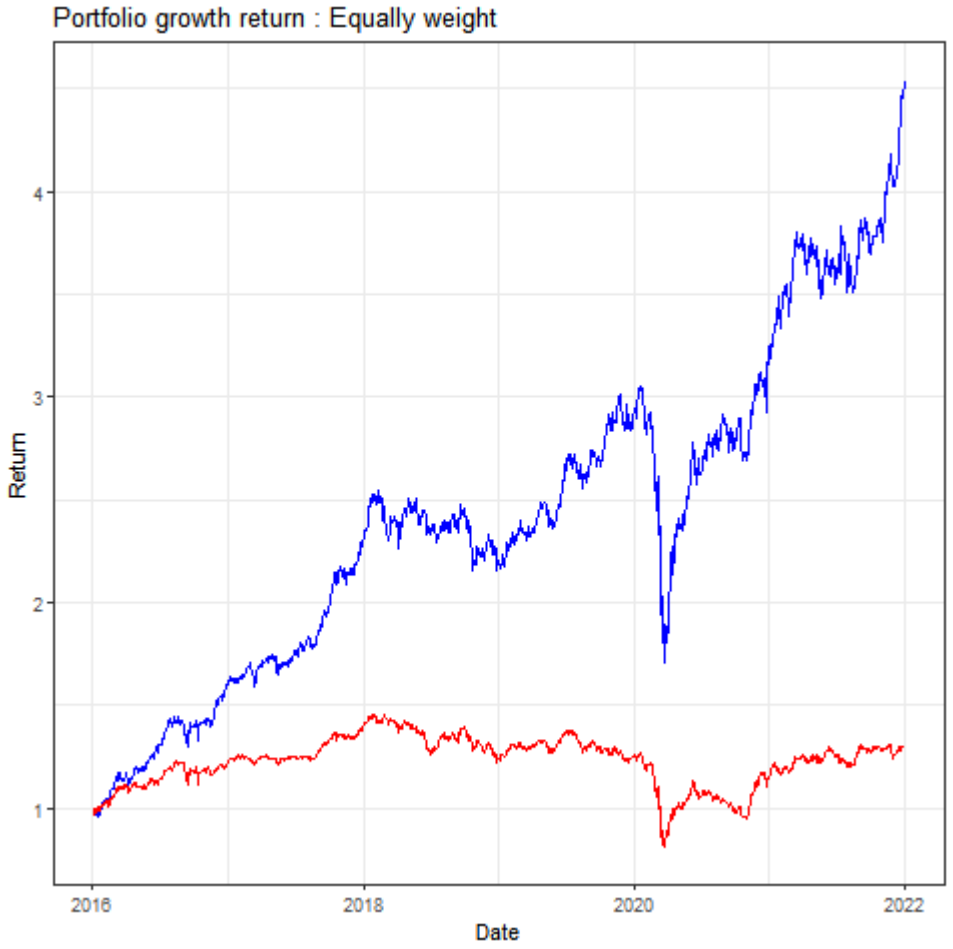
Plot graph cumulative return
ที่เปรียบเทียบระหว่าง Portfolio ที่มี
Optimal weight portfolio
From maximum Sharpe ratio
และ
เทียบกับ ตลาด SET index

Plot graph cumulative return
ที่เปรียบเทียบระหว่าง Portfolio ที่มี
Optimal weight portfolio
From minimum variance
และ
เทียบกับ ตลาด SET index

```r
# Use Minimum variance weights
# Create graph from optimal weight minimum variance portfolio
opti_wts_min_var_graph <- port_growth_func(port_symbol = portfolio_symbol,
                                           port_wts = opti_wts_min_var)
png(file=paste0(path,"graph/portgrowth_optimal_weight_min_var_graph.png"))
plot(opti_wts_min_var_graph +
       labs(x = "Date",
            y = "Return",
            title = "Portfolio growth return : Optimal weight from minimum variance"))
dev.off()
```



```r
# Use Maximum Sharpe Ratio weights
# Create graph from optimal weight maximum Sharpe ratio portfolio
opti_wts_max_sr_graph <- port_growth_func(port_symbol = portfolio_symbol,
                                          port_wts = opti_wts_max_sr)
png(file=paste0(path,"graph/portgrowth_optimal_weight_max_sr_graph.png"))
plot(opti_wts_max_sr_graph +
       labs(x = "Date",
            y = "Return",
            title = "Portfolio growth return : Optimal weight from maximum Sharpe ratio"))
dev.off()
```



Plot graph cumulative return ที่เปรียบเทียบระหว่าง Portfolio ที่มี Equally weight portfolio และ เทียบกับ ตลาด SET index

รายชื่อสมาชิกกลุ่ม

2010511104009 พัชร โสฬสโชคชัย

2010511104025 อัครชัย แสนศิลป์ชัย

2010511104029 ภูบดี กลางถิ่น

2010511104032 เตชณัฐ สุวรรณกันทร

2010511104036 อคิณ ภมรรุ่งโรจน

Thank you

For listening ~