



Sprint 05

Tags	Data Visualization
Status	Done

หา data มาลองฝึกทำได้ที่ : <https://data.world/>

▼ Data Visualization in Google Sheets

<https://docs.google.com/spreadsheets/d/1AmPnRPqDztbBjOS3lx3SVFukeieS1MzUpuizXbv0kMg/edit#gid=0>






▼ SparkLine

- เป็นฟังก์ชันที่สามารถสร้าง chart เล็กๆ ภายใน cell ที่เลือกได้
- สามารถใส่ code สีได้
- เปลี่ยน chart type ได้

```
=SPARKLINE(B3:E3,{"color","red";"linewidth",3})
```

	Y2020				
	Q1	Q2	Q3	Q4	
Toyota	100	200	250	300	
Honda	200	150	120	160	
Mazda	80	250	400	490	
Ford	50	60	55	40	
Nissan	160	150	200	120	

```
=SPARKLINE(B11:C11,{"charttype","bar";"color1","#cf365e";"color2","#717ec9"})
```

	Net Promoter Score		
	0-8	9-10	
Toyota	20	80	
Honda	25	75	
Mazda	12	88	
Ford	35	65	
Nissan	26	74	

▼ Heatmap

- เป็นการ highlight สีใน range ที่เลือก
- อยู่ใน Conditional Formatting
- ควรใช้สีแค่เฉดเดียว แล้วไล่สีอ่อนไปเข้มจะเข้าใจง่ายกว่า

	Y2020			
	Q1	Q2	Q3	Q4
Toyota	100	200	250	300
Honda	200	150	120	160
Mazda	80	250	400	490
Ford	50	60	55	40
Nissan	160	150	200	120


▼ ScoreCard

- เอาไว้โชว์ KPI : Key Performance Indicator เช่น กำไร การเติบโต active_user
- Textbox Show ตัวเลขที่เราต้องการ
- อยู่ในฟังก์ชันการสร้าง chart
- แสดงตัวเลขผลต่างเทียบกับ Base Line (ข้อมูลเก่า) ได้ด้วย
- เทียบแบบ Aggregate ก็ได้

Month	Units Sold	Total Revenue		Base Line
January	5,000	\$ 2,600.00	December	\$ 2,000.00
February	4,000	\$ 2,000.00	<div> Total Revenue \$ 2,600.00 ↑ \$ 600.00 </div>	
March	4,400	\$ 2,200.00		
April	3,000	\$ 1,500.00		
May	4,500	\$ 2,250.00		

Month	Units Sold	Total Revenue	Last Year		Base Line	
January	5,000	\$ 2,600.00	\$ 2,500.00	December	\$ 2,000.00	
February	4,000	\$ 2,000.00	\$ 2,000.00			
March	4,400	\$ 2,200.00	\$ 1,800.00			
April	3,000	\$ 1,500.00	\$ 2,400.00			
May	4,500	\$ 2,250.00	\$ 2,050.00			
	AVG_Revenue	\$ 2,110.00	\$ 2,150.00			

\$ 2,110.00
↓ \$ (40.00)


Chart editor
×

Setup

Customize

123

Scorecard chart

Data range

C4:C5,C4,C2,F2,C2:C6,D2:D6

Combine ranges

Horizontally

Key Value

123

C2:C6

Average

Baseline Value

123

D2:D6

Average

☒ Aggregate

Count
 Max
 Median
 Min
 Sum

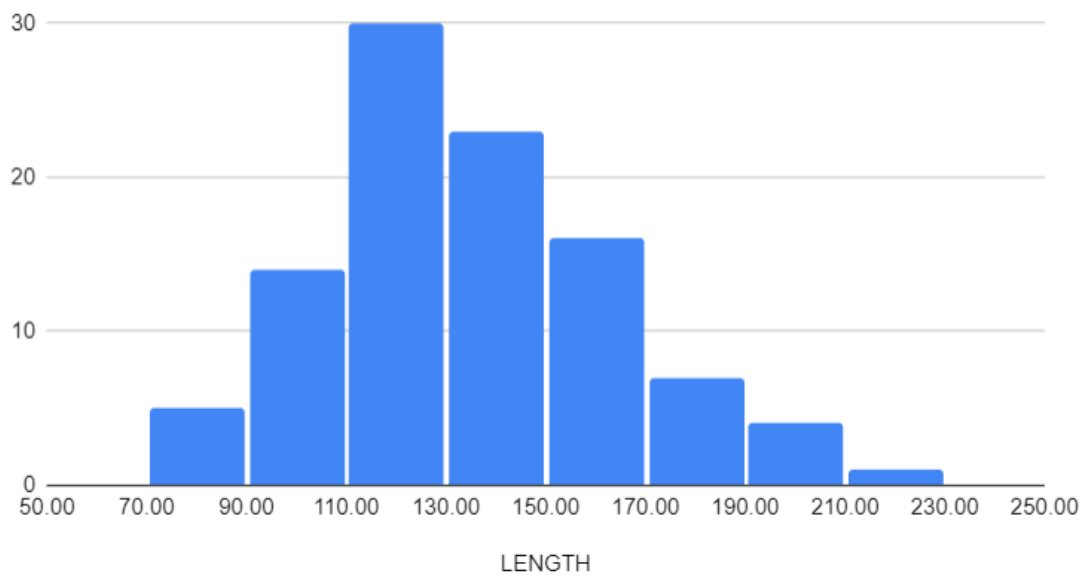
☐ Switch rows / columns

☐ Use row 4 as header

▼ Histogram

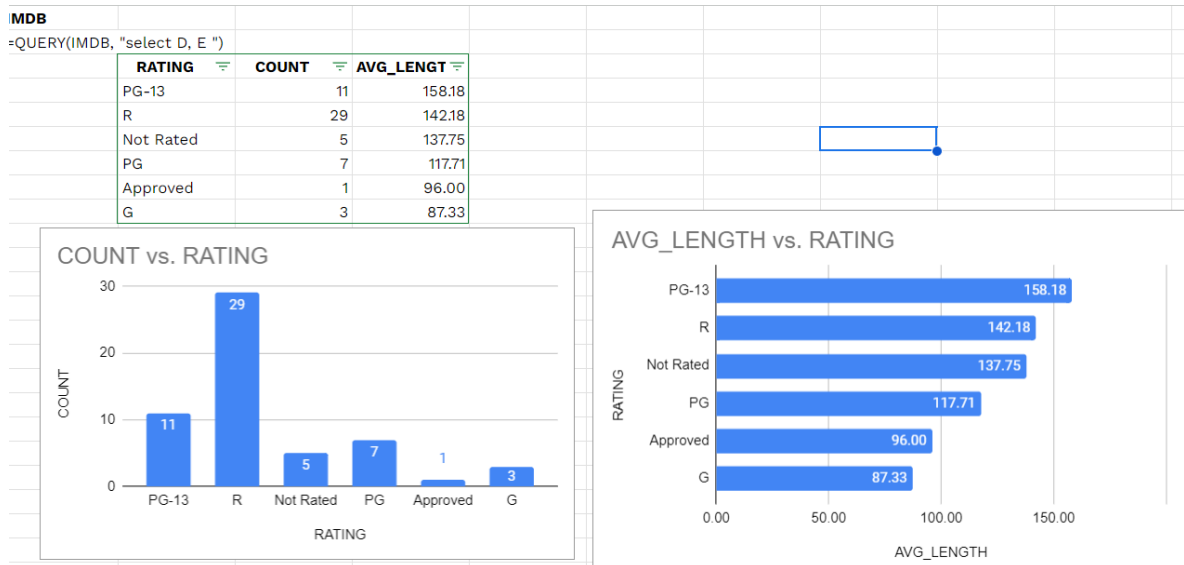
- ใช้อ้างอิงข้อมูลที่อยู่ในแต่ละช่วง

Histogram of LENGTH



▼ Bar

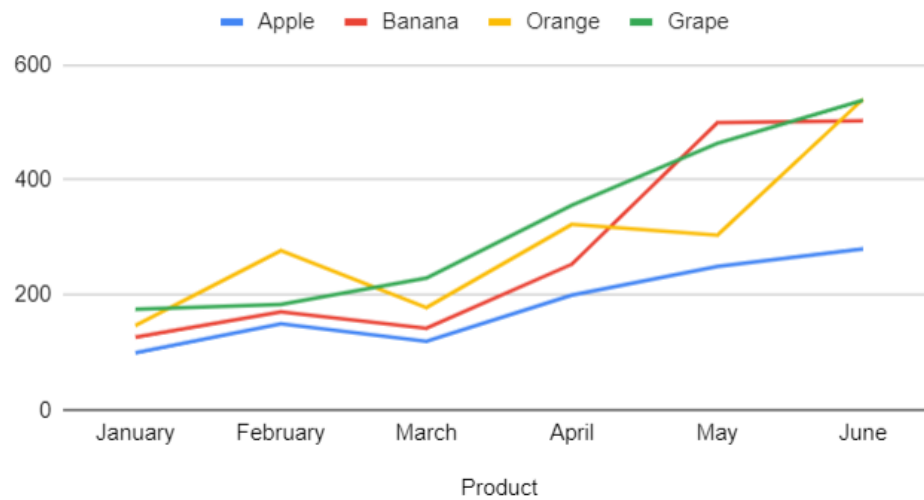
- ใช้หาข้อมูล จากการ Aggregate ต่างๆ จากการแบ่งเป็น หมวดหมู่



▼ Line

- นิยมใช้ line chart กับ time series (ข้อมูลที่เกี่ยวข้องกับ "Date time")

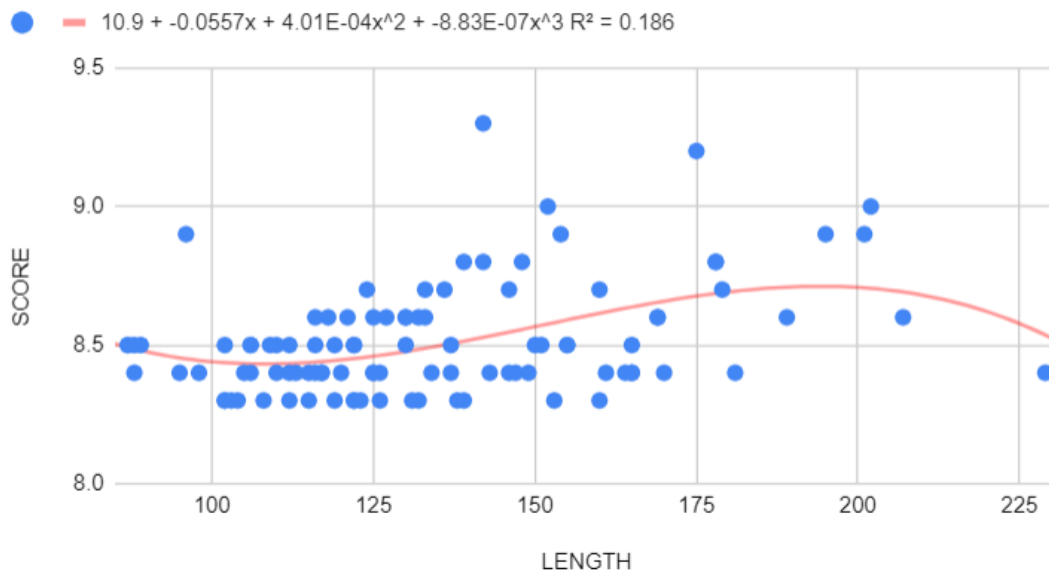
Apple, Banana, Orange and Grape



▼ Scatter

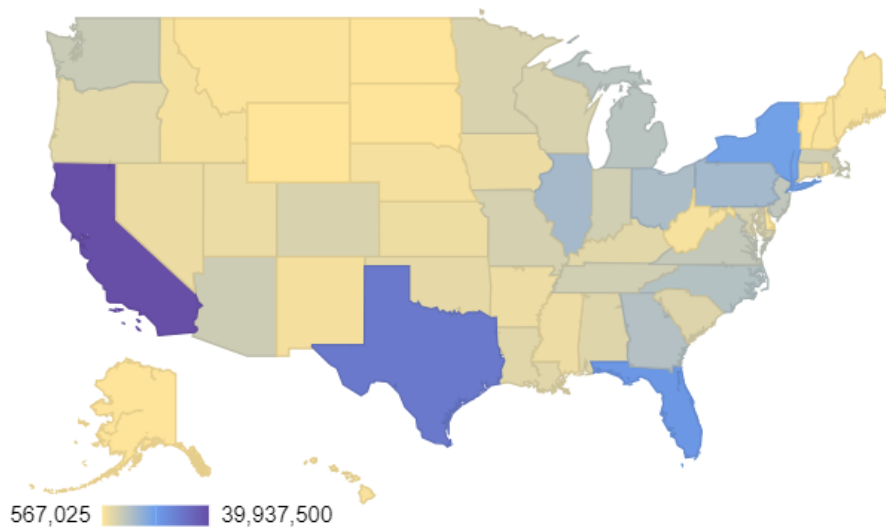
- ใช้ความสัมพันธ์ระหว่าง 2 column
- แต่ละ column ต้องเป็นตัวเลขทั้งคู่
- สร้าง Trend line
- Polynomial ไม่ควรเกิน 4

SCORE vs. LENGTH



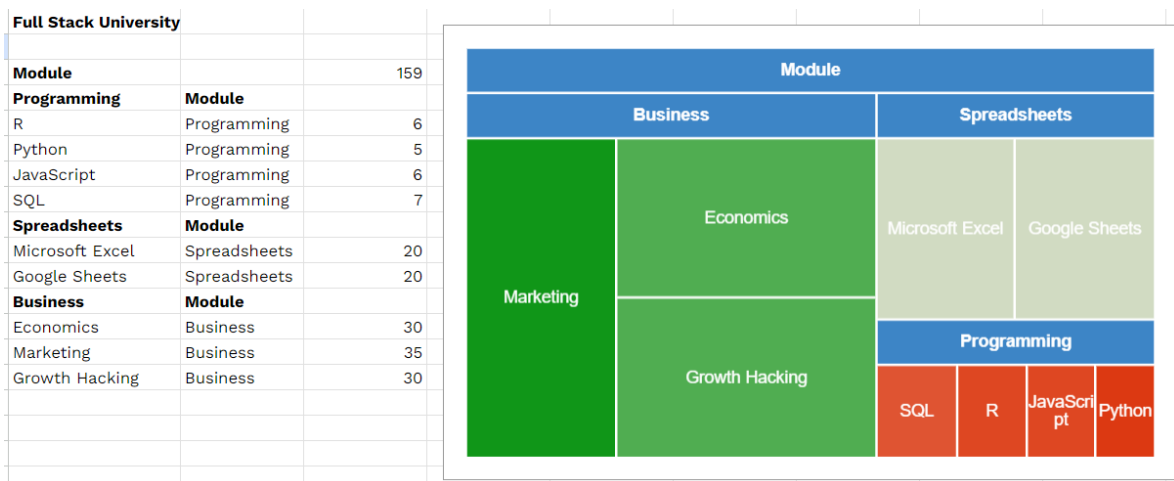
▼ Geomap

- show เป็น map



▼ Treemap

- ต้องวางข้อมูลตาม Default Template
- ไม้ดู contribution ได้เบื้องต้น



▼ Data Visualization in R

ขั้นแรกในการทำ Statistic คือ การ Plot กราฟ

data type : factor เอาไว้ใช้กำหนดชื่อแกนให้กับข้อมูลก่อนจะนำไป plot กราฟ

▼ code

```
library(tidyverse)
## get working directory
getwd()

## library tidyverse
library(tidyverse)
```

```

## basic plot base R
hist(mtcars$mpg)

## Analyzing horse power
## Histogram - one Quantitative Variable
hist(mtcars$hp)
mean(mtcars$hp)
median(mtcars$hp)

str(mtcars$mpg)
mtcars$am <- factor(mtcars$am,
                    levels = c(0, 1),
                    labels = c("Auto", "Manual"))

## Bar Plot - one Qualitative Variable
table(mtcars$am)
barplot(mtcars$am)

## Box Plot
boxplot(mtcars$hp)
fivenum(mtcars$hp)

min(mtcars$hp)
quantile(mtcars$hp, probs = .25)
median(mtcars$hp)
quantile(mtcars$hp, probs = .75)
max(mtcars$hp)

## Whisker Calculation
Q3 <- quantile(mtcars$hp, probs = .75)
Q1 <- quantile(mtcars$hp, probs = .25)
IQR_hp <- Q3 - Q1

Q3 + 1.5 * IQR_hp
Q1 - 1.5 * IQR_hp

boxplot.stats(mtcars$hp, coef = 1.5)

## filter out outliers
mtcars_no_outlier <- mtcars %>%
  filter(hp < 335)

boxplot(mtcars_no_outlier$hp)

## Boxplot 2 Variables
## Quantitative x Quantitative
data(mtcars)

```

```
mtcars$am <- factor(mtcars$am,
                    levels = c(0, 1),
                    labels = c("Auto", "Manual"))
boxplot(mpg ~ am, data = mtcars,
        col = c("gold", "salmon"))

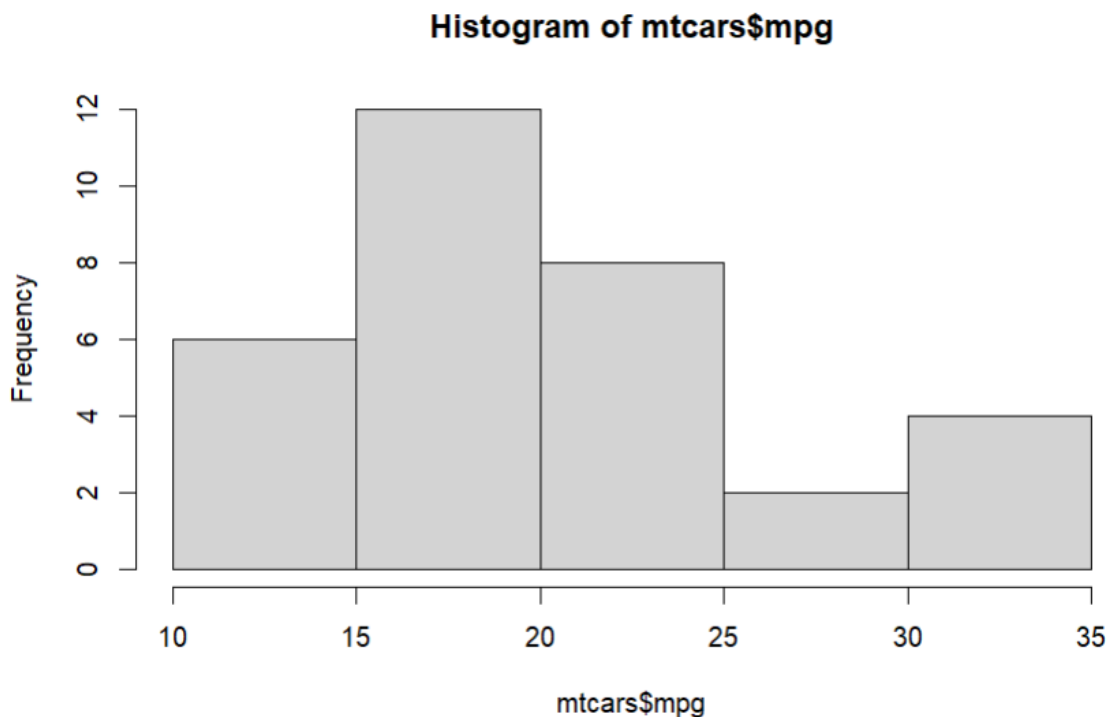
## Scatter Plot
## 2 x Quantitative
plot(mtcars$hp, mtcars$mpg, pch = 16,
     col = "blue",
     main = "My first scatter plot",
     xlab = "Horse Power",
     ylab = "Miles Per Gallon")

cor(mtcars$hp, mtcars$mpg)
lm(mpg ~ hp, data = mtcars)
```

▼ hist

เป็นฟังก์ชัน basic plot ที่อยู่ใน base R

```
hist(mtcars$mpg)
```



▼ str

ไว้ใช้ดูโครงสร้าง และ Data Type ของชุดข้อมูล


```
str(mtcars)
```

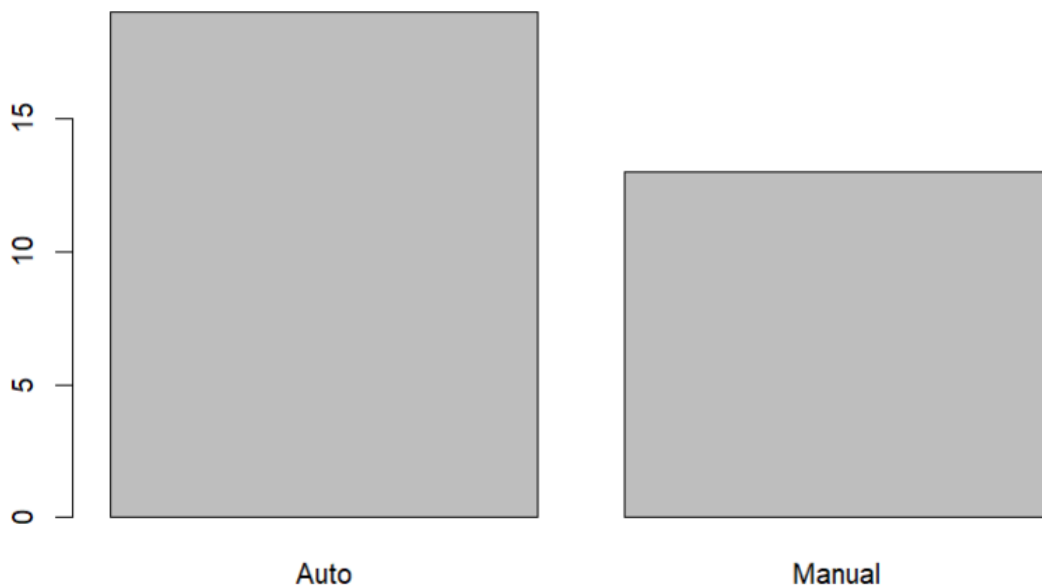
```
> str(mtcars)
'data.frame':   32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num   6  6  4  6  8  6  8  4  4  6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110  93 110 175 105 245  62  95 123 ...
 $ drat: num   3.9  3.9  3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num   2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num   0  0  1  1  0  1  0  1  1  1 ...
 $ am  : num   1  1  1  0  0  0  0  0  0  0 ...
 $ gear: num   4  4  4  3  3  3  3  4  4  4 ...
 $ carb: num   4  4  1  1  2  1  4  2  2  4 ...
```

▼ barplot

- เป็นการ plot กราฟในรูปของ Bar chart
- ใช้ร่วมกับ ฟังก์ชัน table : เป็นฟังก์ชันที่นับความถี่ของข้อมูล
- table คล้ายฟังก์ชัน count แต่ count จะดีกว่า

```
mtcars$am <- factor(mtcars$am,
                    levels = c(0, 1),
                    labels = c("Auto", "Manual"))
```

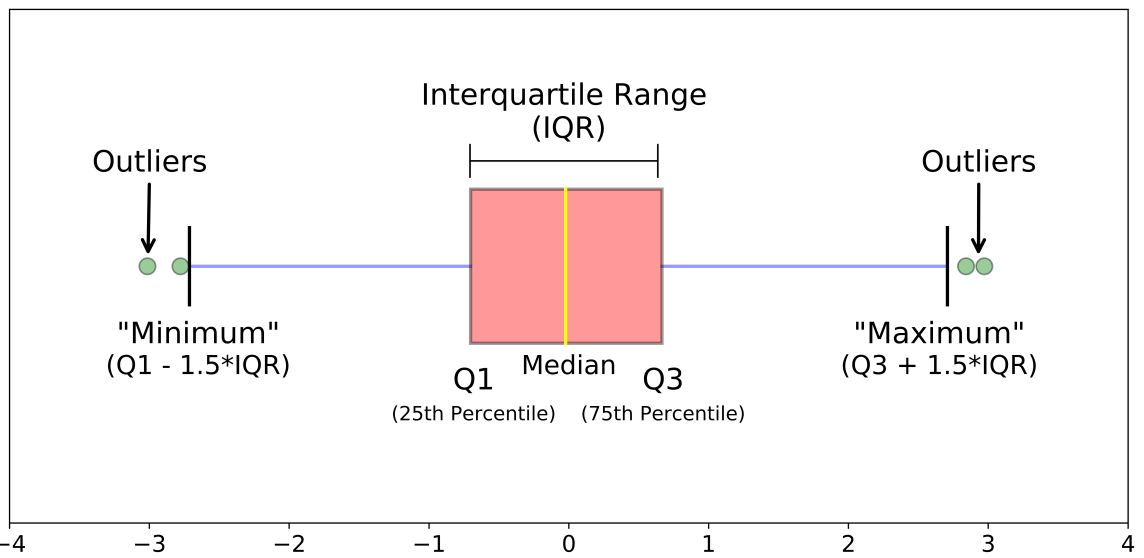
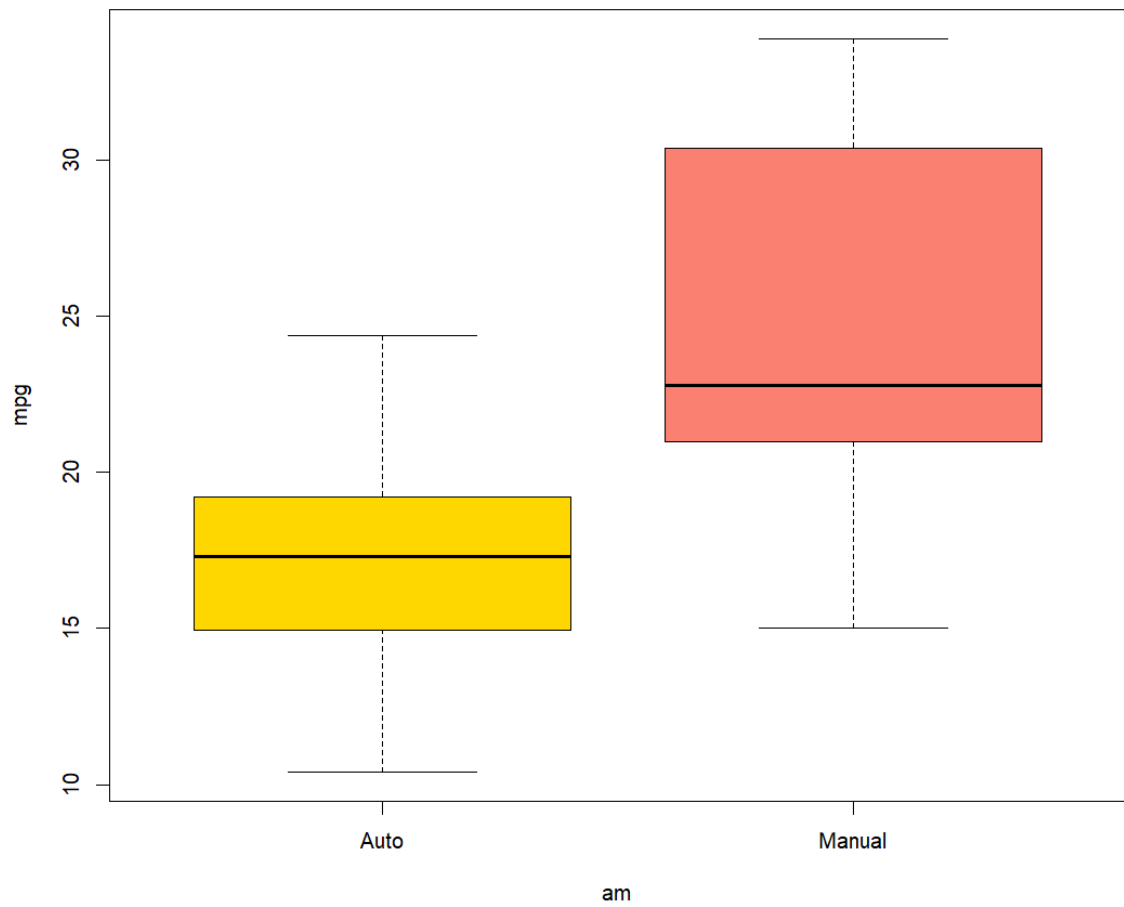
```
## Bar Plot - one Qualitative Variable
barplot(table(mtcars$am))
```



▼ boxplot

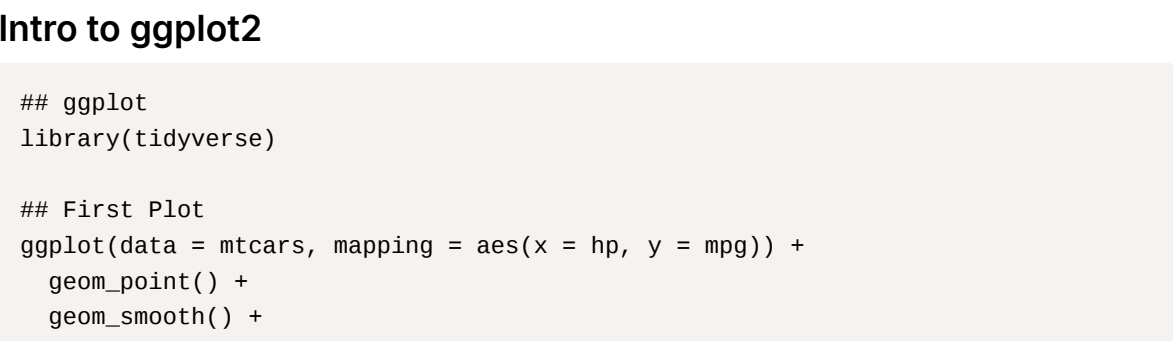
- เป็นกราฟค่าสถิติที่สำคัญๆ โดยจะบอกตำแหน่งของข้อมูล เช่น min, max, median=q2, q1, q3, outlier
- ส่วนแขนเรียกว่า Whisker

```
## Quantitative x Quantitative
data(mtcars)
mtcars$am <- factor(mtcars$am,
                    levels = c(0, 1),
                    labels = c("Auto", "Manual"))
boxplot(mpg ~ am, data = mtcars,
        col = c("gold", "salmon"))
```



▼ plot

- ```
plot(mtcars$hp, mtcars$mpg, pch = 16,
 col = "blue",
 main = "My first scatter plot",
 xlab = "Horse Power",
 ylab = "Miles Per Gallon")
```



```

geom_rug()

ggplot(mtcars, aes(x = hp, y = mpg)) +
 geom_point(size = 3, col = "blue", alpha = 0.3)

ggplot(mtcars, aes(hp)) +
 geom_histogram(bins = 10, fill = "red", alpha = .4)

ggplot(mtcars, aes(hp)) +
 geom_boxplot()

p <- ggplot(mtcars, aes(hp))
p + geom_histogram(bins = 10)
p + geom_density()
p + geom_boxplot()

Box Plot by groups
diamonds %>%
 count(cut)

ggplot(diamonds, aes(cut)) +
 geom_bar(fill = "salmon")

ggplot(diamonds, aes(cut, fill = color)) +
 geom_bar(position = "fill")

ggplot(diamonds, aes(cut, fill = color)) +
 geom_bar(position = "stack")

ggplot(diamonds, aes(cut, fill = color)) +
 geom_bar(position = "dodge")

Scatter Plot
set.seed(42)
small_diamonds <- sample_n(diamonds, 5000)

ggplot(small_diamonds, aes(carat, price)) +
 geom_point(size = 1) +
 geom_smooth(method = "lm") +
 facet_wrap(~color, ncol = 2) +
 theme_minimal() +
 labs(title = "Relationship between carat and price by color",
 x = "Carat",
 y = "Price USD",
 caption = " Source : Diamonds from ggplot2 package")

Final Example
ggplot(small_diamonds, aes(carat, price, col = cut)) +

```

```
geom_point(size = 3, alpha = .2) +
facet_wrap(~color, ncol = 2) +
theme_minimal()
```

- เป็นฟังก์ชันที่สร้าง visualization ใน R
- รวมอยู่ใน library(tidyverse)
- ต้องการ 3 requirement : Geometry, Mapping, Data
- Data : ชุดข้อมูล อาจจะมีหลาย column
- Mapping : การดึง column ไป map ที่ chart ของเรา
- Geometry : การ design chart เช่น ขนาด, สี เป็นต้น
- วิธีการเลือกกราฟ 1.ดูว่าใช้กี่ตัวแปร 2.ดูว่าเป็น quanli = discrete หรือ quanti = continuous

## Data visualization with ggplot2 : CHEAT SHEET

### Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data set**, a **coordinate system**, and **geoms**—visual marks that represent data points.

data  
F = F  
A = A

+

coordinate system  
x = x  
y = y

=

plot

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

data  
F = F  
A = A  
color = F  
size = A

+

coordinate system  
x = x  
y = y

=

plot

Complete the template below to build a graph.

```
ggplot(data = <DATA>) +
 <GEOM FUNCTION>[mapping = aes(<MAPPINGS>),
 stat = <STAT>, position = <POSITION>] +
 <COORDINATE FUNCTION> +
 <FACET FUNCTION> +
 <SCALE FUNCTION> +
 <THEME FUNCTION>
```

**ggplot**(data = mpg, aes(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

**last\_plot()** Returns the last plot.

**ggsave**("plot.png", width = 5, height = 5) Saves last plot as 5" x 5" file named "plot.png" in working directory. Matches file type to file extension.

### Aes

**Common aesthetic values.**  
**color and fill** - string ("red", "RRRRBBB")  
**linetype** - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotteddash", 5 = "longdash", 6 = "twodash")  
**lineend** - string ("round", "butt", or "square")  
**linejoin** - string ("round", "mitre", or "bevel")  
**size** - integer (line width in mm) 0 1 2 3 4 5 6 7 8 9 10 11 12  
**shape** - integer/shape name or a single character ("a") 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

### Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

#### GRAPHICAL PRIMITIVES

```
a <- ggplot(economics, aes(date, unemployment))
b <- ggplot(seals, aes(x = long, y = lat))
```

**a + geom\_blank()** and **a + expand\_limits()**  
 Ensure limits include values across all plots.  
**b + geom\_curve**(aes(yend = lat + 1, xend = long + 1), curvature = 1) - x, yend, y, alpha, angle, color, curvature, linetype, size  
**a + geom\_path**(lineend = "butt", linejoin = "round", linemitre = 1) - x, y, alpha, color, group, linetype, size  
**a + geom\_polygon**(aes(alpha = 50)) - x, y, alpha, color, fill, group, subgroup, linetype, size  
**b + geom\_rect**(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)) - xmin, xmax, ymin, ymax, alpha, color, fill, linetype, size  
**a + geom\_ribbon**(aes(ymin = unemployment - 900, ymax = unemployment + 900)) - x, ymax, ymin, alpha, color, fill, group, linetype, size

#### LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size  
**b + geom\_abline**(aes(intercept = 0, slope = 1))  
**b + geom\_hline**(aes(yintercept = lat))  
**b + geom\_vline**(aes(xintercept = long))  
**b + geom\_segment**(aes(yend = lat + 1, xend = long + 1))  
**b + geom\_spoke**(aes(angle = 1:1155, radius = 1))

#### ONE VARIABLE continuous

```
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)
```

**c + geom\_area**(stat = "bin") - x, y, alpha, color, fill, linetype, size  
**c + geom\_density**(kernel = "gaussian") - x, y, alpha, color, fill, group, linetype, size, weight  
**c + geom\_dotplot**() - x, y, alpha, color, fill  
**c + geom\_freqpoly**() - x, y, alpha, color, group, linetype, size  
**c + geom\_histogram**(binwidth = 5) - x, y, alpha, color, fill, linetype, size, weight  
**c2 + geom\_qq**(aes(sample = hwy)) - x, y, alpha, color, fill, linetype, size, weight

#### discrete

```
d <- ggplot(mpg, aes(fill))
```

**d + geom\_bar**() - x, y, alpha, color, fill, linetype, size, weight

#### TWO VARIABLES both continuous

```
e <- ggplot(mpg, aes(cty, hwy))
```

**e + geom\_label**(aes(label = cty), nudge\_x = 1, nudge\_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust  
**e + geom\_point**() - x, y, alpha, color, fill, shape, size, stroke  
**e + geom\_quantile**() - x, y, alpha, color, group, linetype, size, weight  
**e + geom\_rug**(sides = "bl") - x, y, alpha, color, linetype, size  
**e + geom\_smooth**(method = lm) - x, y, alpha, color, fill, group, linetype, size, weight  
**e + geom\_text**(aes(label = cty), nudge\_x = 1, nudge\_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

#### one discrete, one continuous

```
f <- ggplot(mpg, aes(class, hwy))
```

**f + geom\_col**() - x, y, alpha, color, fill, group, linetype, size  
**f + geom\_boxplot**() - x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight  
**f + geom\_dotplot**(binaxis = "y", stackdir = "center") - x, y, alpha, color, fill, group  
**f + geom\_violin**(scale = "area") - x, y, alpha, color, fill, group, linetype, size, weight

#### both discrete

```
g <- ggplot(diamonds, aes(cut, color))
```

**g + geom\_count**() - x, y, alpha, color, fill, shape, size, stroke  
**e + geom\_jitter**(height = 2, width = 2) - x, y, alpha, color, fill, shape, size

#### continuous bivariate distribution

```
h <- ggplot(diamonds, aes(carat, price))
```

**h + geom\_bin2d**(binwidth = c(0.25, 500)) - x, y, alpha, color, fill, linetype, size, weight  
**h + geom\_density\_2d**() - x, y, alpha, color, group, linetype, size  
**h + geom\_hex**() - x, y, alpha, color, fill, size

#### continuous function

```
i <- ggplot(economics, aes(date, unemployment))
```

**i + geom\_area**() - x, y, alpha, color, fill, linetype, size  
**i + geom\_line**() - x, y, alpha, color, group, linetype, size  
**i + geom\_step**(direction = "hv") - x, y, alpha, color, group, linetype, size

#### visualizing error

```
df <- data.frame(grp = c("A", "B"), fit = 4.5, se = 1.2)
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))
```

**j + geom\_crossbar**(fatten = 2) - x, y, ymax, ymin, alpha, color, fill, group, linetype, size  
**j + geom\_errorbar**() - x, y, ymax, ymin, alpha, color, group, linetype, size, width  
 Also **geom\_errorbarh**()  
**j + geom\_linerange**() - x, y, ymax, ymin, alpha, color, group, linetype, size  
**j + geom\_pointrange**() - x, y, ymax, ymin, alpha, color, fill, group, linetype, shape, size

#### maps

```
k <- ggplot(data, aes(fill = murder))
```

**k + geom\_map**(aes(map\_id = state), map = map) + **expand\_limits**(x = map\$long, y = map\$lat)  
 map\_id, alpha, color, fill, linetype, size

#### THREE VARIABLES

```
sealsSz <- with(seals, sqrt(delta_long^2 + delta_lat^2)); l <- ggplot(seals, aes(long, lat))
```

**l + geom\_contour**(aes(z = z)) - x, y, z, alpha, color, group, linetype, size, weight  
**l + geom\_contour\_filled**(aes(fill = z)) - x, y, alpha, color, fill, group, linetype, size, subgroup  
**l + geom\_raster**(aes(fill = z), hjust = 0.5, vjust = 0.5, interpolate = FALSE) - x, y, alpha, fill  
**l + geom\_tile**(aes(fill = z)) - x, y, alpha, color, fill, linetype, size, width

Note : alpha อยู่ระหว่าง 0-1

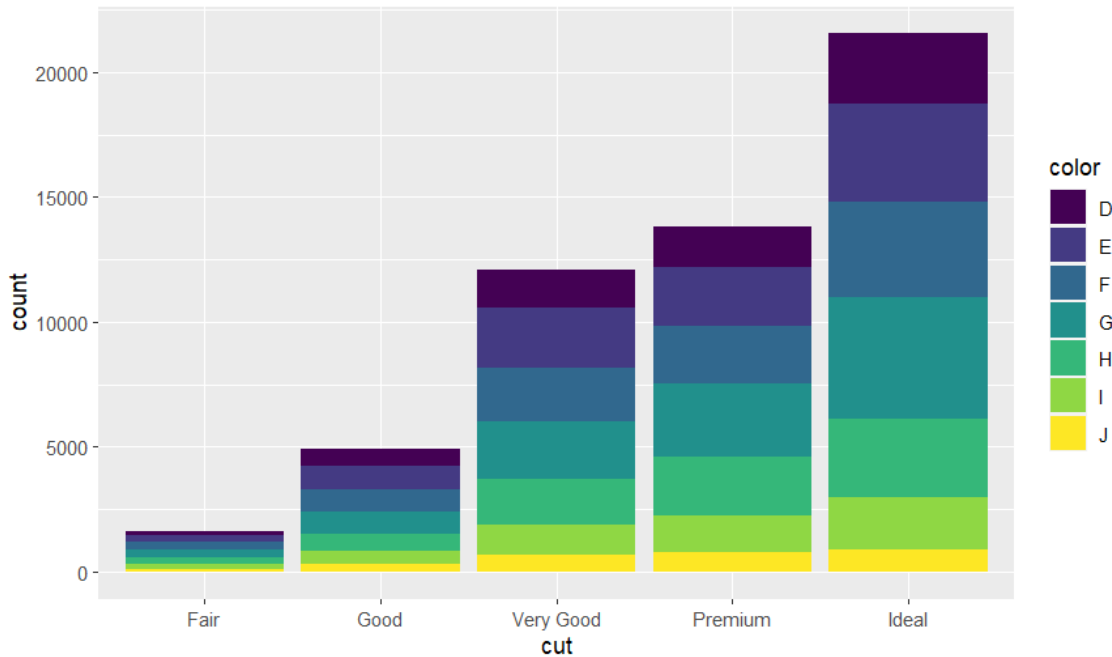
Note : ตัวแปร ord คือ ตัวแปรกลุ่มที่เรียงสูงกลางต่ำได้

### ▼ bar chart

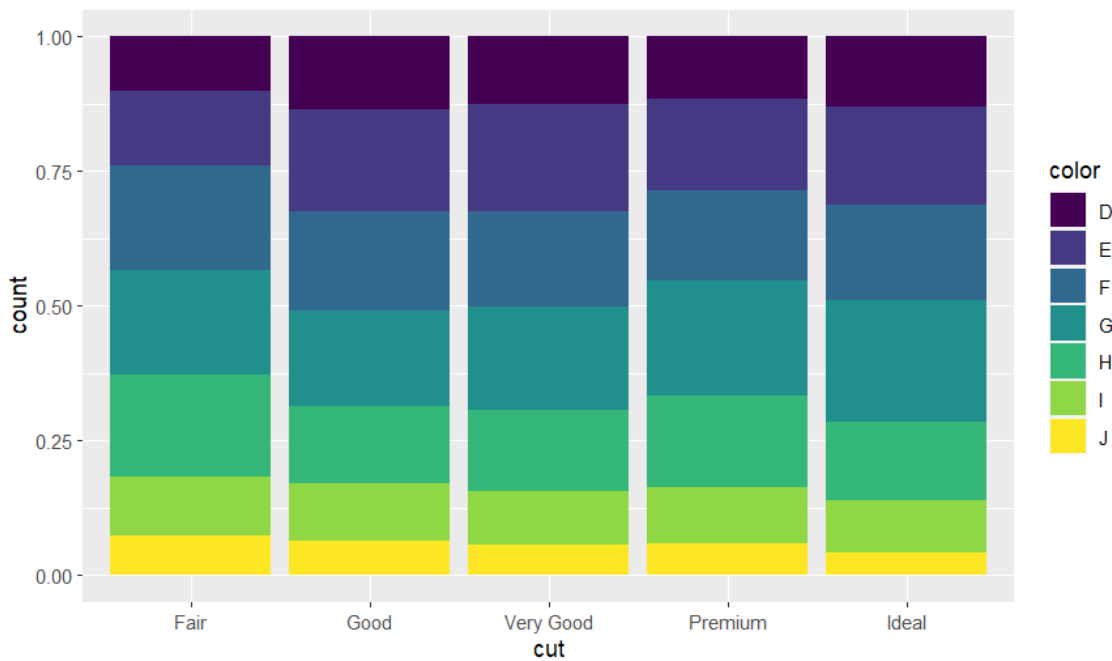
```
ggplot(diamonds, aes(cut, fill = color)) +
 geom_bar(position = "stack")
```

Sprint 05

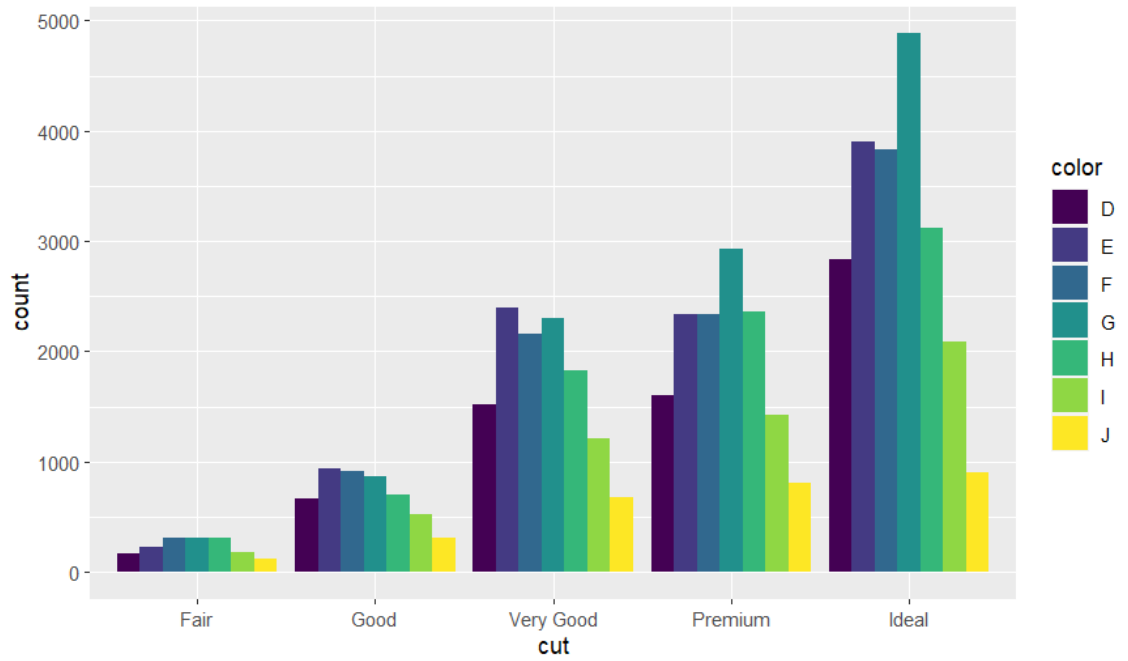
14



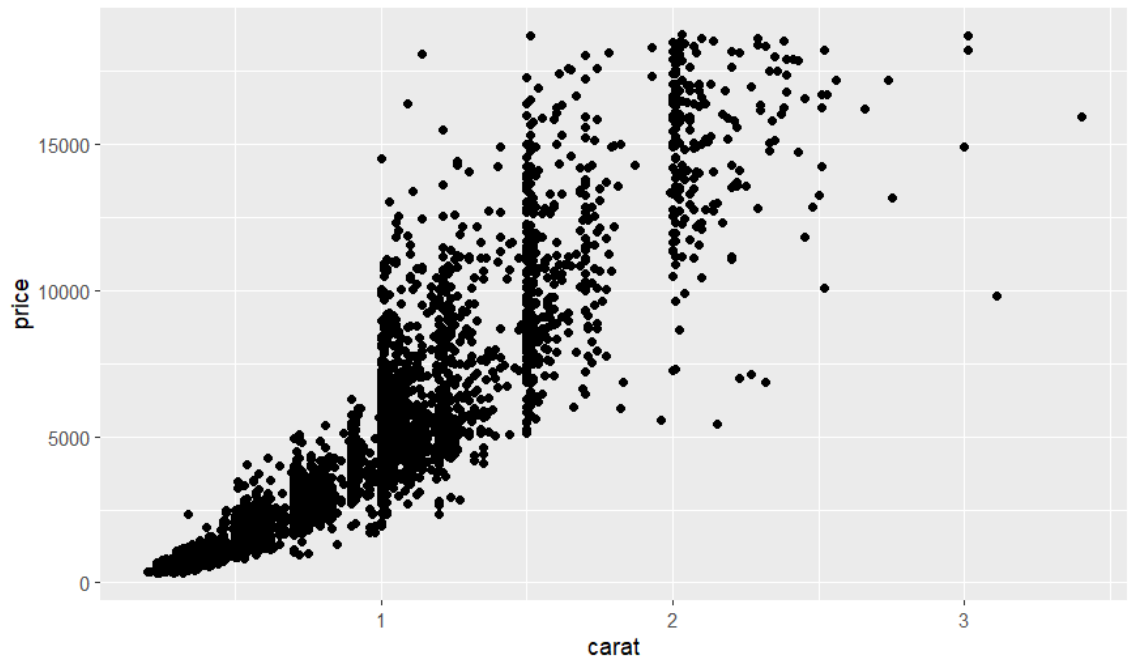
```
ggplot(diamonds, aes(cut, fill = color)) +
 geom_bar(position = "fill")
```



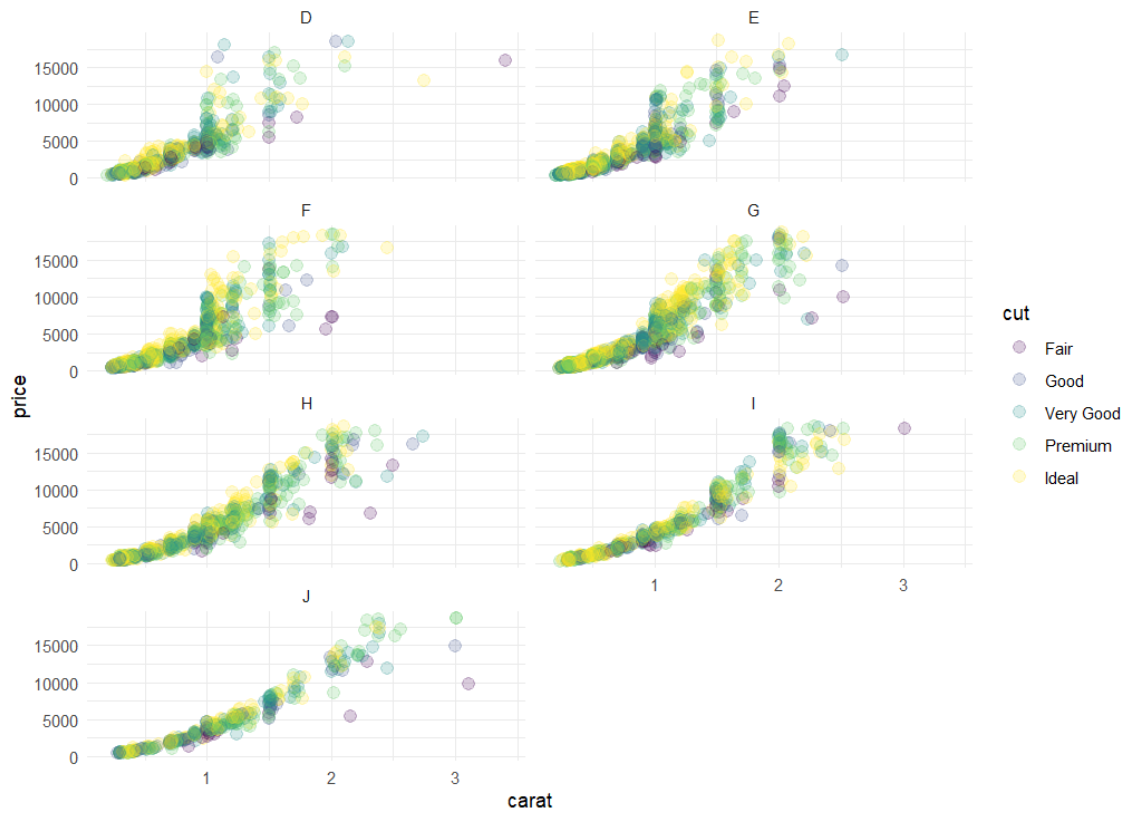
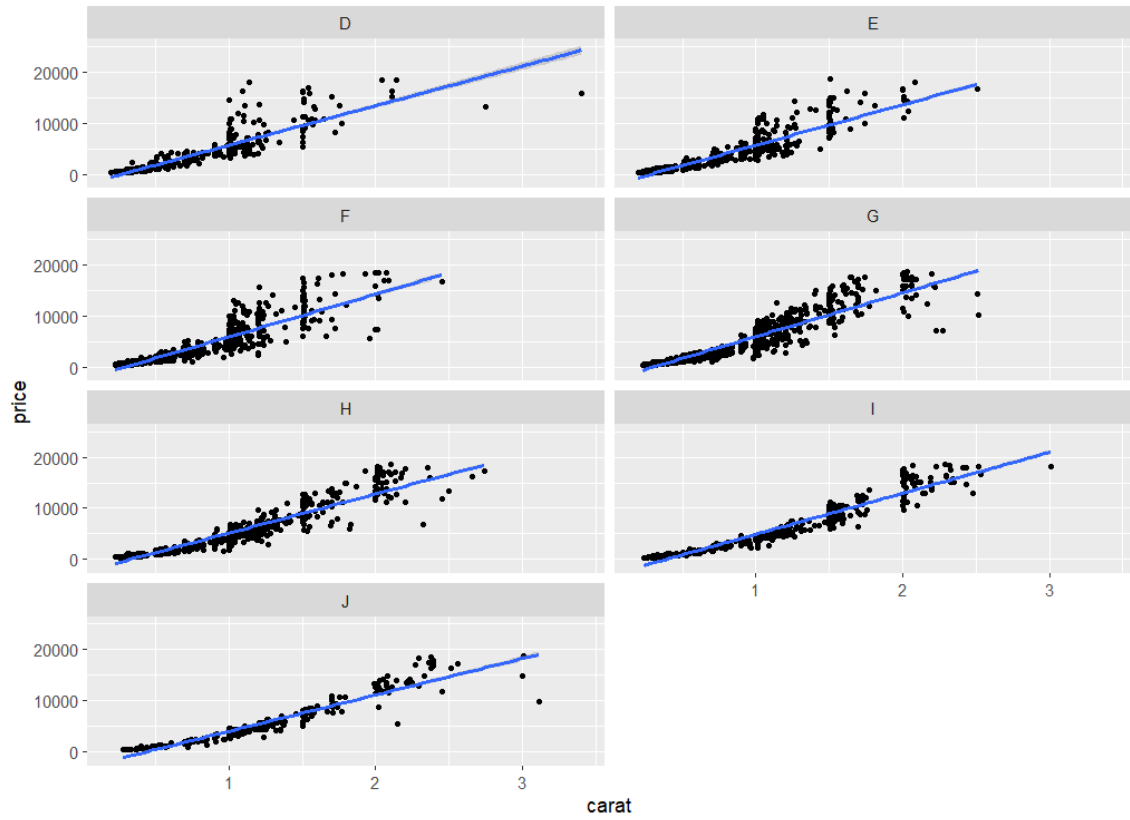
```
ggplot(diamonds, aes(cut, fill = color)) +
 geom_bar(position = "dodge")
```



▼ scatter chart



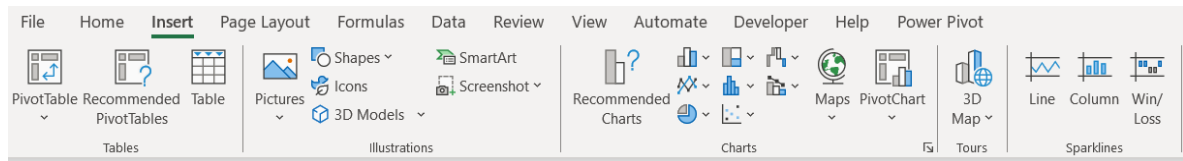




## ▼ Data Visualization in Excel

### ▼ Sparkline

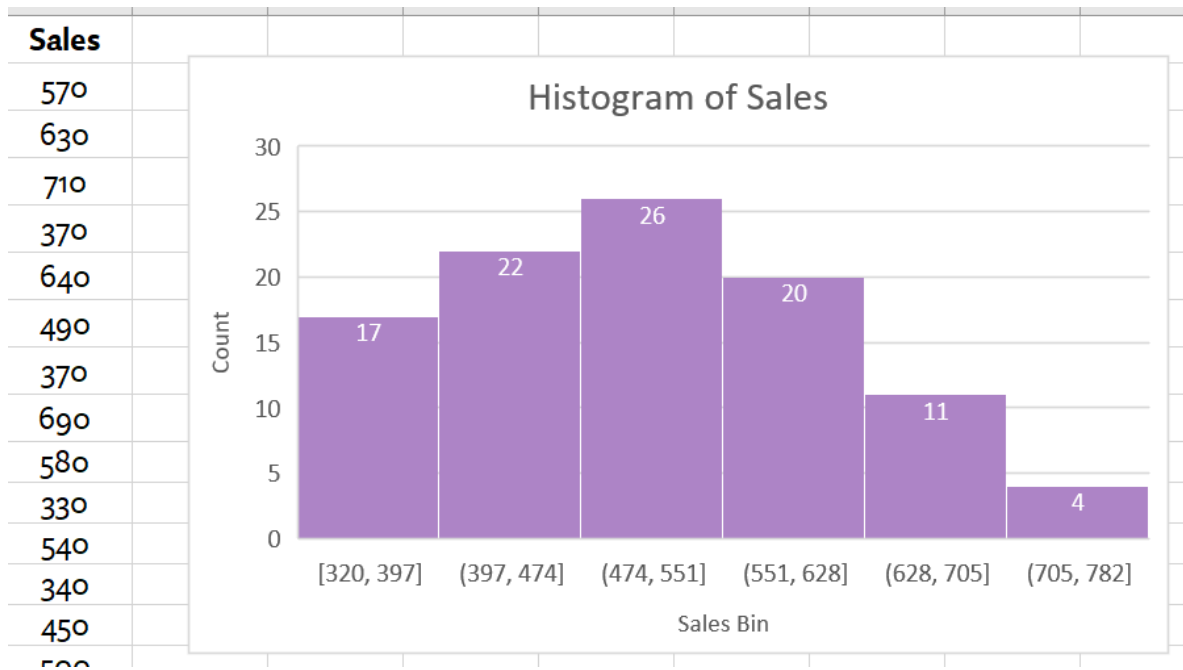
การสร้าง Chart ขนาดเล็กภายใน Cell



|          | Q1   | Q2   | Q3    | Q4    |  |  |
|----------|------|------|-------|-------|--|--|
| Apple    | 10   | 15   | 25    | 50    |  |  |
| Banana   | 40   | 25   | 30    | 40    |  |  |
| Carrot   | 50   | 45   | 20    | 5     |  |  |
|          |      |      |       |       |  |  |
|          | Q1   | Q2   | Q3    | Q4    |  |  |
| % Growth | 2.5% | 4.0% | -2.5% | -5.0% |  |  |

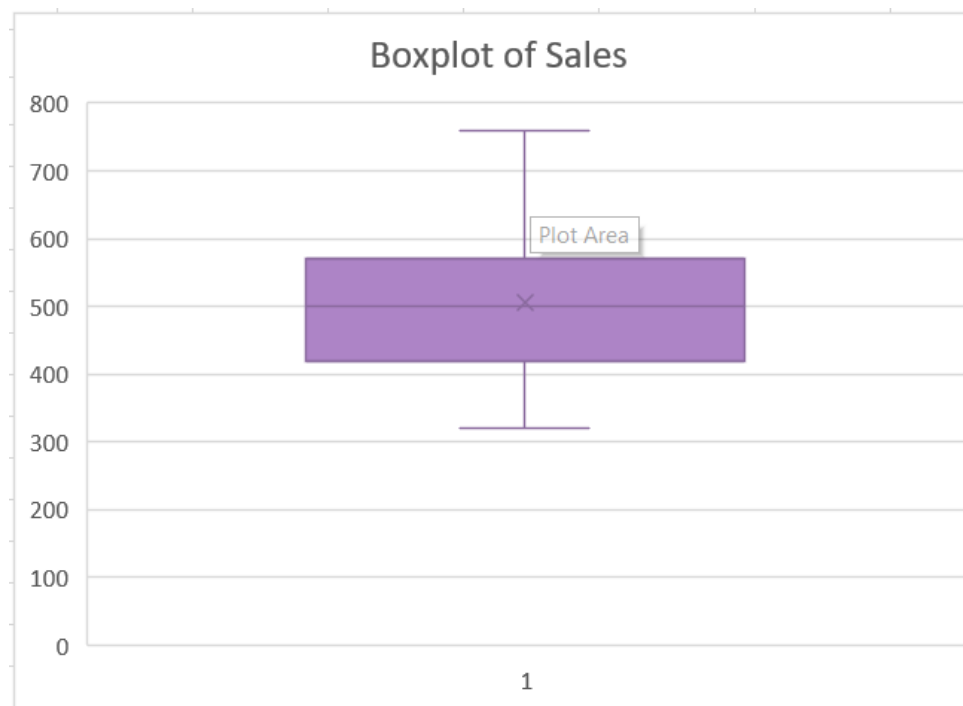
### ▼ Histogram

- แสดงผลข้อมูล 1 column ที่เป็นตัวเลข
- ดูการกระจายของข้อมูลเป็นช่วงๆ



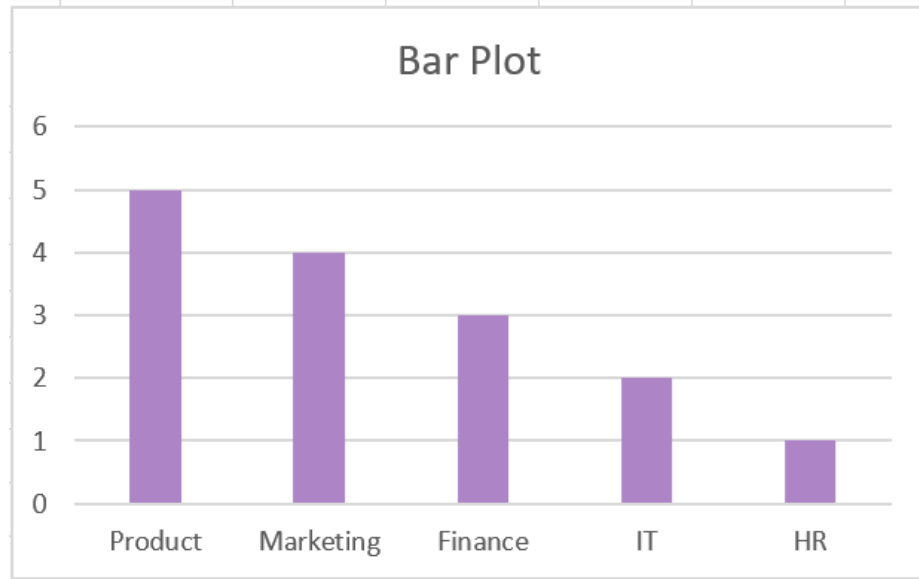
## ▼ Boxplot

ไว้หาข้อมูลที่ Extreme มากๆ(outlier) แล้ว filter ออก



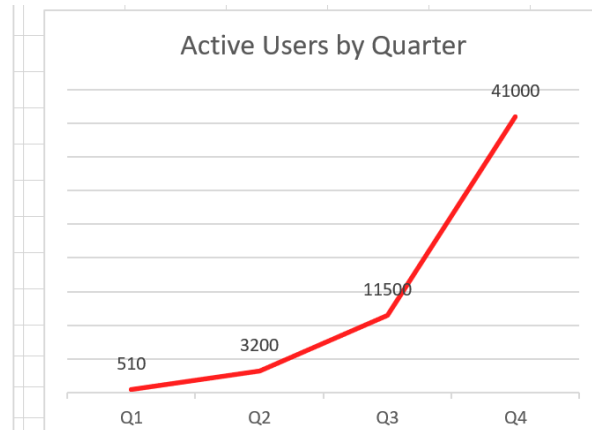
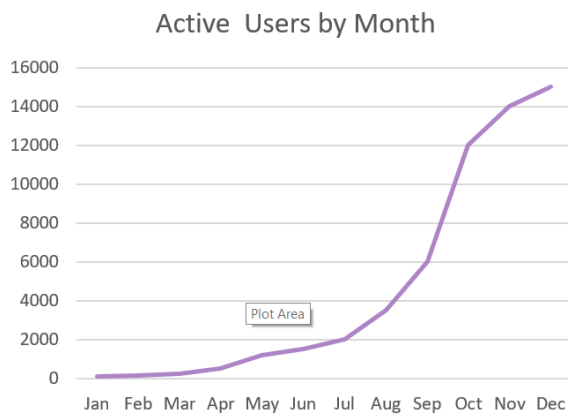
## ▼ Bar

|           |   |           |   |
|-----------|---|-----------|---|
| Marketing | 4 | Product   | 5 |
| HR        | 1 | Marketing | 4 |
| Finance   | 3 | Finance   | 3 |
| IT        | 2 | IT        | 2 |
| Product   | 5 | HR        | 1 |

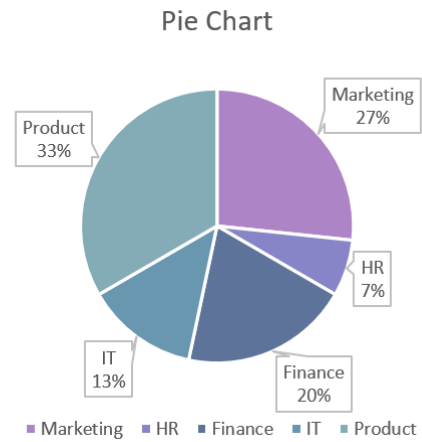
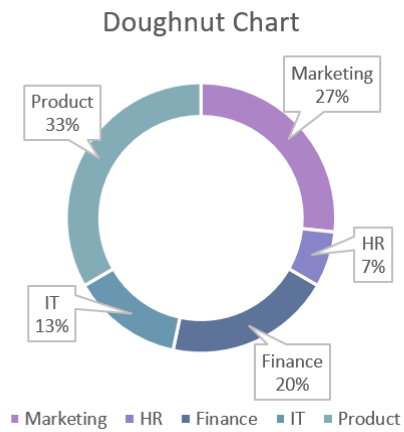


## ▼ Line

จะใช้กับ ข้อมูลที่สัมพันธ์กับเวลา

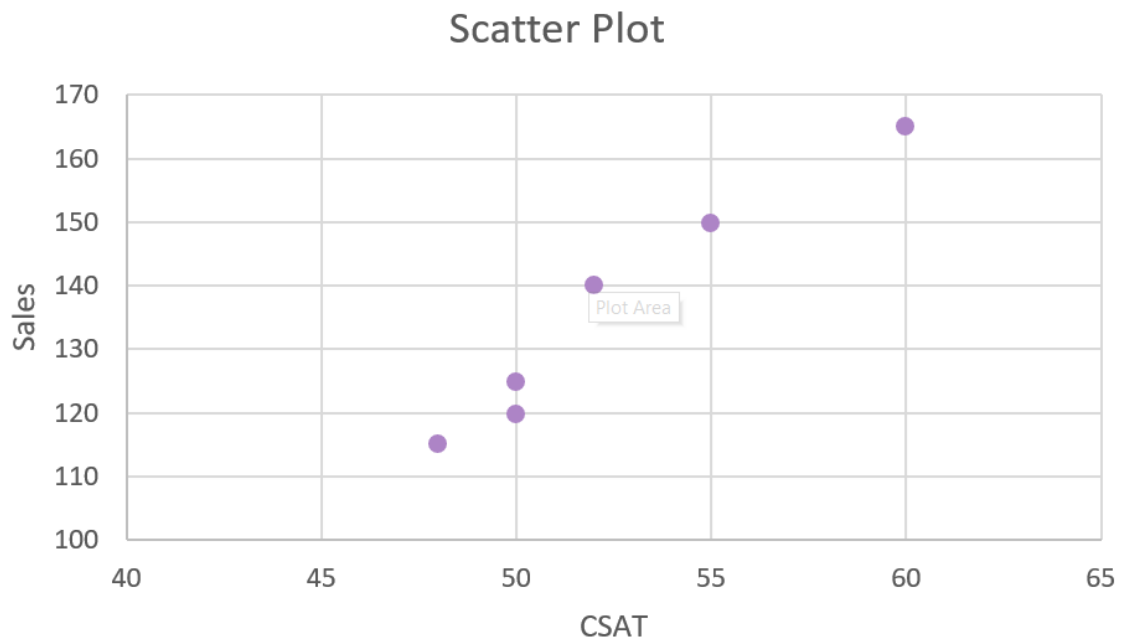


## ▼ Pie & Doughnut

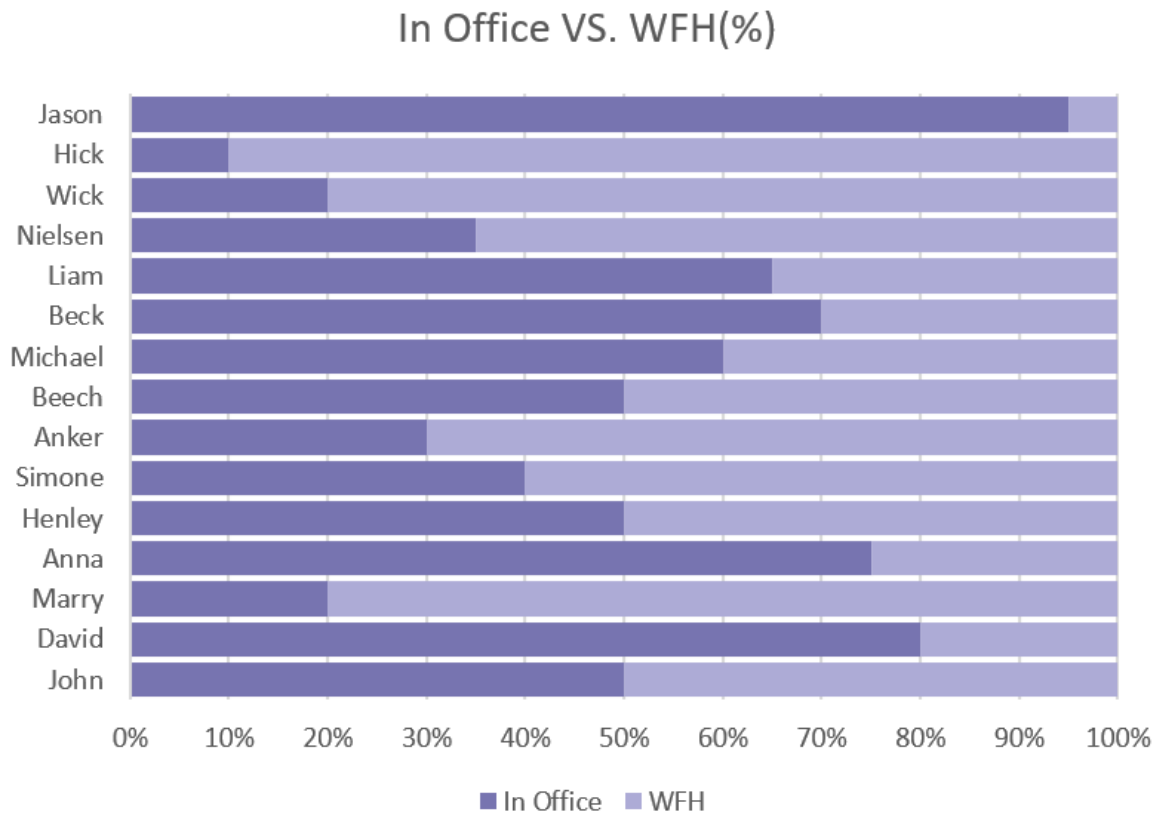


## ▼ Scatter Plot

ใช้ดูความสัมพันธ์ของชุดข้อมูลที่เป็นตัวเลขเชิงปริมาณ ทั้งคู่

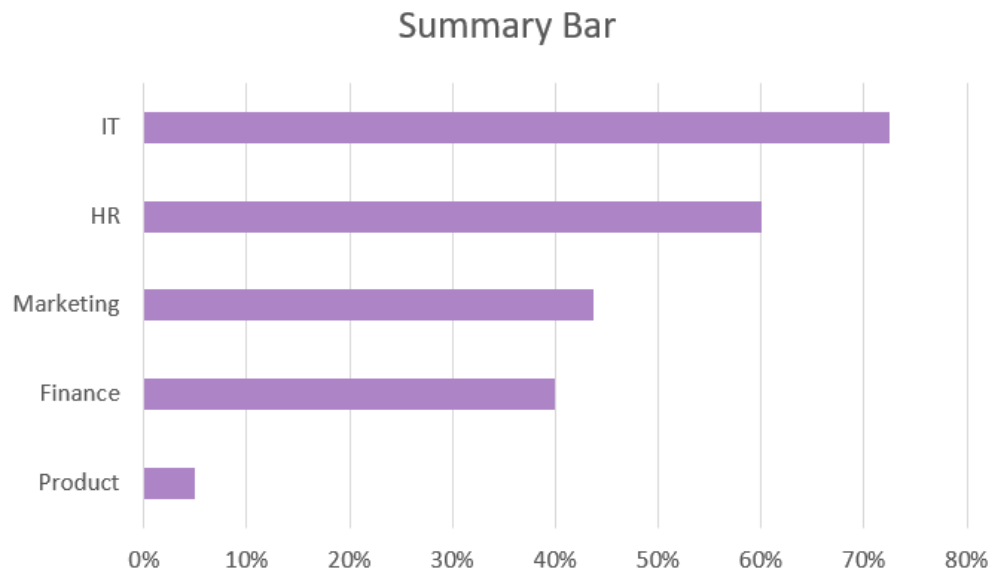


## ▼ Stacked Bar



## ▼ Summary Bar

จะมีการใช้ฟังก์ชัน unique, sort มีช่วยในการจัดการข้อมูล



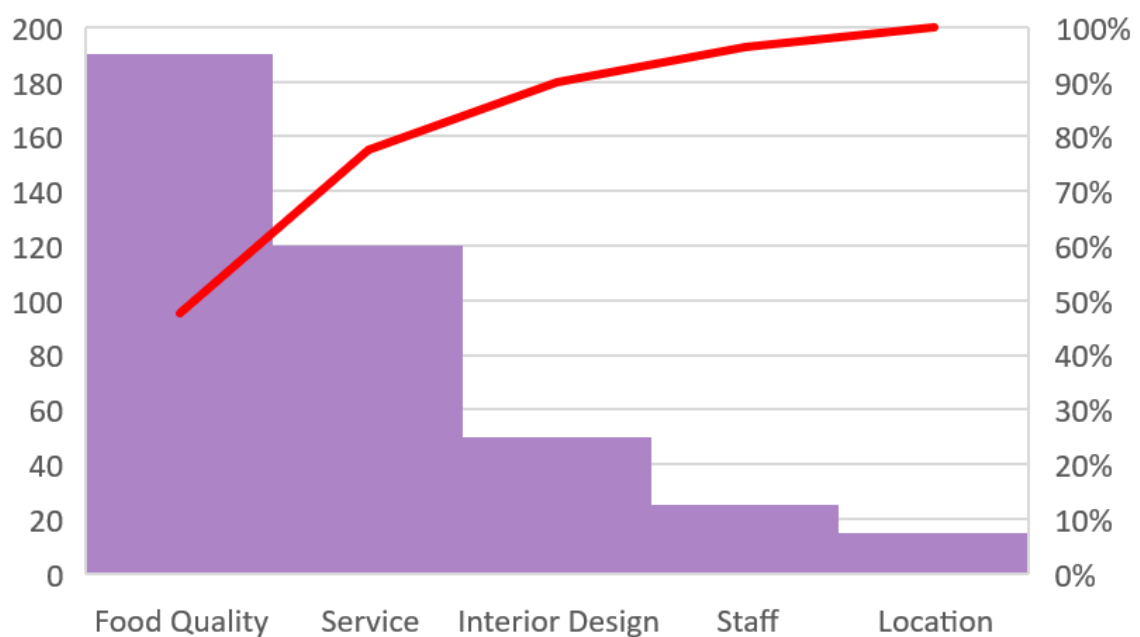
## ▼ Pareto

- ใช้บ่อยมากในงาน Quality control หรือ ปัจจัยในสิ่งที่ส่งผลกับความพอใจของลูกค้ามากที่สุดจากการทำ Survey

- เส้นสีแดงเป็น cumulative percentage

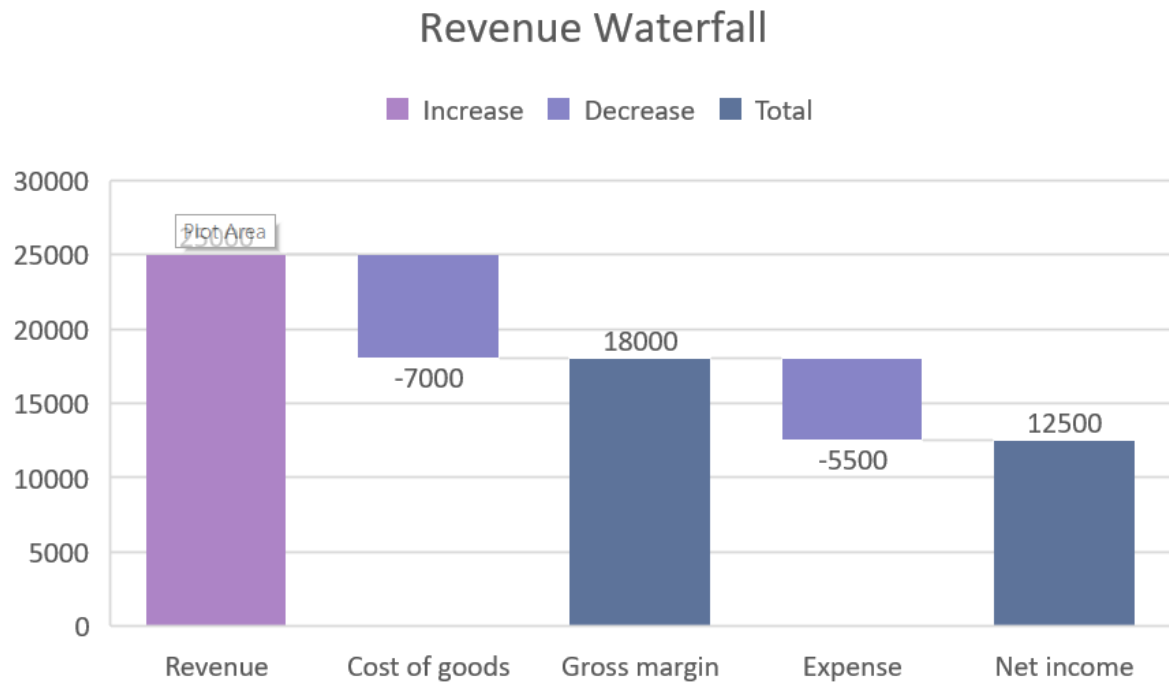
| Customer Satisfaction |     | percentage | cumulative |
|-----------------------|-----|------------|------------|
| Food Quality          | 190 | 48%        | 48%        |
| Service               | 120 | 30%        | 78%        |
| Interior Design       | 50  | 13%        | 90%        |
| Staff                 | 25  | 6%         | 96%        |
| Location              | 15  | 4%         | 100%       |

Pareto Chart

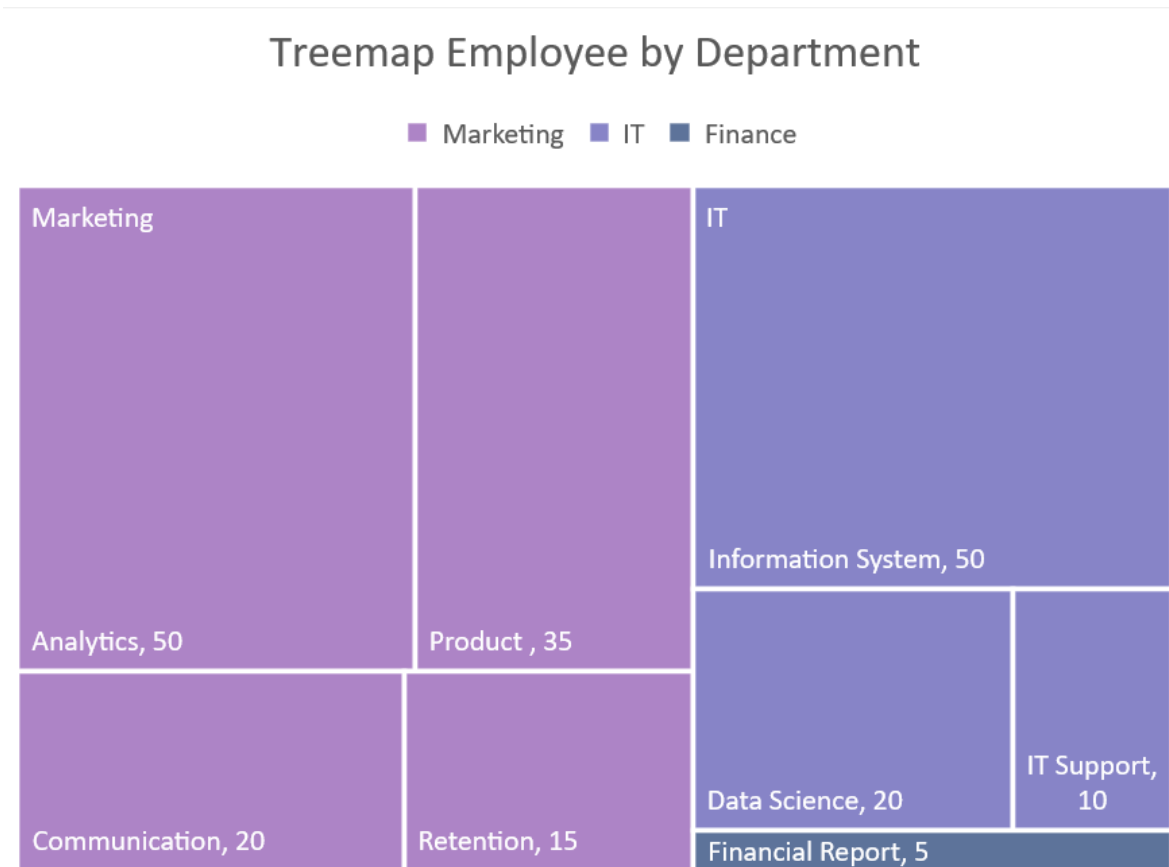


## ▼ Waterfall

เลือก Set as total เพื่อให้ Gross margin and Net income ลงไปอยู่ที่ Base line



#### ▼ Treemap



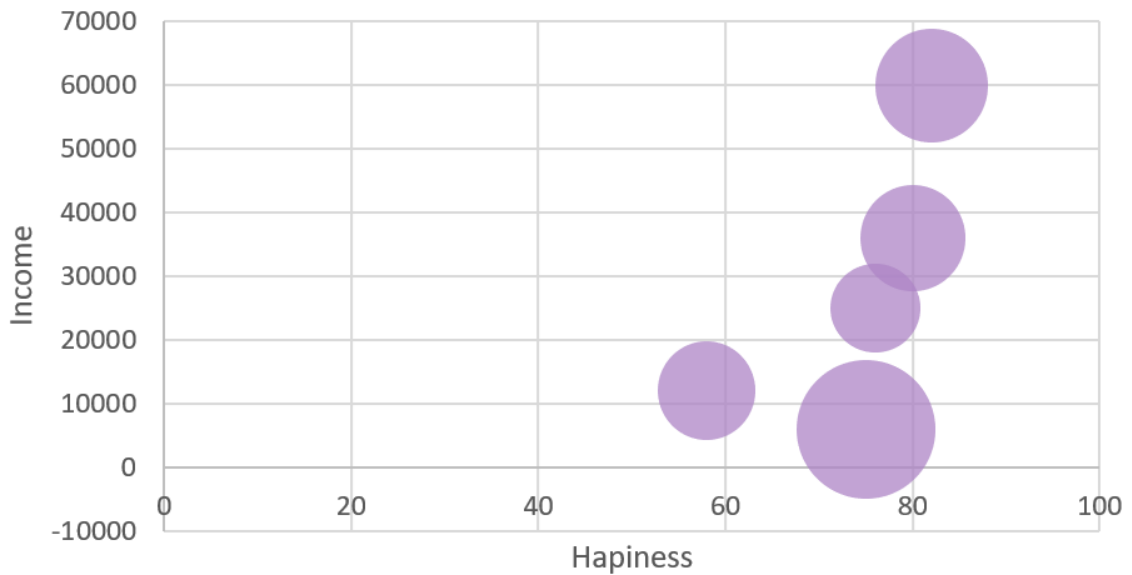


## ▼ Bubble

คือ Scatter Plot ที่เราเพิ่มตัวแปรที่ 3 เข้าไป ซึ่งก็คือขนาดของตัว Bubble

|             | Happiness | Income | Pop Size |
|-------------|-----------|--------|----------|
| Teen        | 58        | 12000  | 6000     |
| Adult 25-35 | 76        | 25000  | 5000     |
| Adult 36-45 | 80        | 36000  | 7000     |
| Adult 46-60 | 82        | 60000  | 8000     |
| Old         | 75        | 6000   | 12000    |

Bubble Chart



## ▼ Bingmap

## ▼ Pivot Chart

