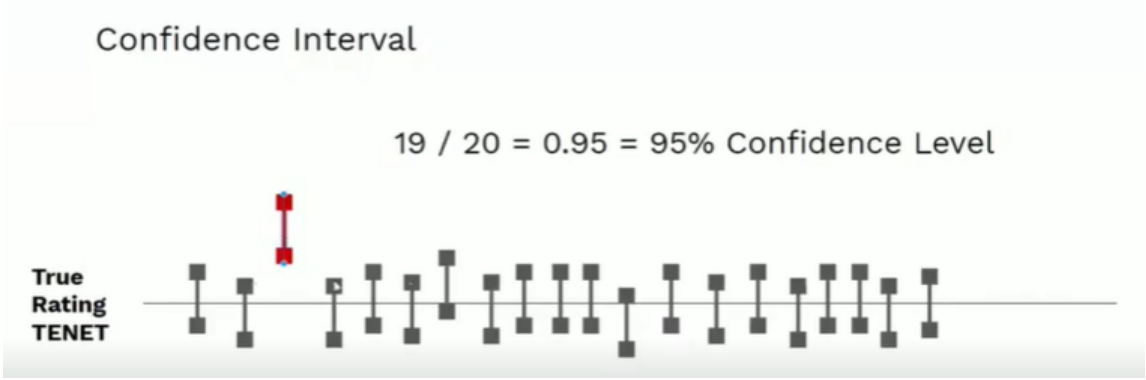
Sprint 06

Tags	Statistics
Status	Not started

▼ Basic Statistics 101

Simulate data in google sheet

- `RAND()` : การสุ่มตัวเลข 0 - 1
- `NORMINV(probability, mean, SD)` : การเปลี่ยนตัวเลขความน่าจะเป็นให้กลายเป็นตัวที่อยู่ใน Normal Distribution
- Confidence Interval : ค่าความเชื่อมั่นว่าค่าเฉลี่ยของข้อมูลชุดนี้ควรจะอยู่ช่วงไหน เช่น Confidence Interval เท่ากับ 95% คือ การมั่นใจ 95% ว่าถ้าหากชุดข้อมูลมีการเปลี่ยนตัวเลขไปค่าเฉลี่ยที่ได้จะอยู่ในช่วงที่กำหนด
- `CONFIDENCE.NORM(error, SD, sample_size)` : เรียกว่า Margin Error โดยจะนำค่าที่ได้ไปบวกลบกับค่าเฉลี่ยเพื่อหา ช่วงความเชื่อมั่นออกมา (Lower & Upper Bound)



▼ Build Model in R

LM : Linear Model

- ใช้กับตัวแปรที่เป็น numeric

```
lmfit <- lm(mpg ~ hp, data = mtcars)

note : lm(ตัวแปรตาม ~ ตัวแปรต้น, table ที่ใช้)
```

```
Call:
lm(formula = mpg ~ hp, data = mtcars)

Coefficients:
(Intercept)           hp
   30.09886     -0.06823
```

lm : เป็นฟังก์ที่ใช้สร้าง model linear regression โดยจะให้ผลลัพธ์เป็น Coefficients ออกมา เช่น ค่า intercept กับ slope

Summary

```
summary(lmfit)

note : summary(model ที่สร้างขึ้นมา)
```

```
Call:
lm(formula = mpg ~ hp, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max
-5.7121 -2.1122 -0.8854  1.5819  8.2360

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 30.09886    1.63392  18.421  < 2e-16 ***
hp          -0.06823    0.01012  -6.742 1.79e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.863 on 30 degrees of freedom
Multiple R-squared:  0.6024,    Adjusted R-squared:  0.5892
F-statistic: 45.46 on 1 and 30 DF,  p-value: 1.788e-07
```

summary : จะให้ข้อมูลออกมามากกว่า lm

Predict()

```
predict(lmfit, newdata = new_cars)

note : predict(coefficient, ตัวแปรต้นที่ต้องนำมา prediction)
note : ตัวแปรต้นที่นำมาใช้ data type ต้องเป็น data frame or list
```

predict : ใช้ในการทำนายค่าตัวแปรตามจากตัวแปรต้นที่ input เข้าไป

Multiple Linear Regression

```
## Root mean square error
## Multiple Linear Regression
## mpg = f(hp, wt, am)
## mpg = intercept + b0*hp + b1*wt + b2*am

lmfit_V2 <- lm(mpg ~ hp + wt + am, data = mtcars)
```

```
coefs <- coef(lmfit_V2)
coefs[[1]] + coefs[[2]]*200 + coefs[[3]]*3.5 + coefs[[4]]*
```

Build Full Model

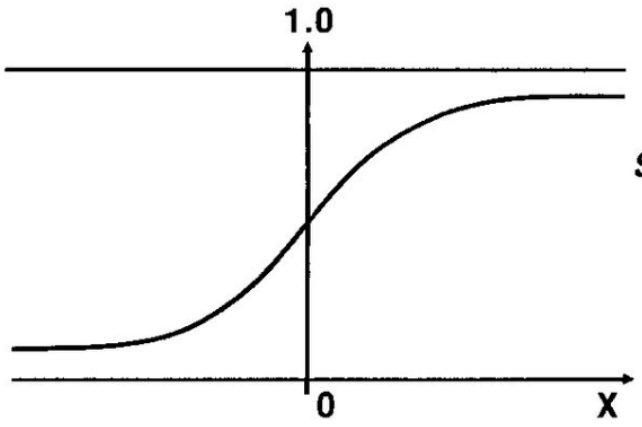
```
## Build Full Model
lmfit_Full <- lm(mpg ~ ., data = mtcars)
## if you don't want to some independent variable
lmfit_Full <- lm(mpg ~ . -gear, data = mtcars)
lmfit_Full

mtcars$predicted <- predict(lmfit_Full)
head(mtcars)

## Train RMSE
squared_error <- (mtcars$mpg - mtcars$predicted)**2
(rmse <- sqrt(mean(squared_error)))

note : ค่า Error ของ data set ที่ใช้ Train model นี้ขึ้นมา
```

Logistic Regression

- ใช้กับตัวแปรที่เป็น factor

Logistic Function

เราใช้สูตรคณิตศาสตร์ เพื่อปรับผลลัพธ์ให้อยู่ระหว่าง 0-1 (เหมือนความน่าจะเป็น)



Sigmoid

$$f(x) = \frac{1}{1+e^{-x}}$$

1

2

`e^z / (1 + e^z)`

สองสูตรได้ผลลัพธ์เหมือนกัน

```
logit_model <- glm(am ~ mpg, data = train_data, family = "
note : family เป็นตัวบอกว่าจะใช้ model ในรูปแบบที่ให้ค่าตัวแปรตามเป็น 0,1

p_train <- predict(logit_model,type = "response")
note : type จะเป็นตัวบีบค่าทำนายให้อยู่ระหว่าง 0,1 หรือ Probability
```

```
## Logistic Regression

library(dplyr)
mtcars %>% head()

str(mtcars$am)

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

class(mtcars$am)
table(mtcars$am)
```

```
## Split Data
set.seed(450)
n <- nrow(mtcars)
id <- sample(1:n, size = n*0.7)
train_data <- mtcars[id, ]
test_data <- mtcars[-id, ]

## train model
logit_model <- glm(am ~ mpg, data = train_data, family = "
p_train <- predict(logit_model,type = "response") ## proba
train_data$pred <- if_else(p_train >= 0.5, "Manual", "Auto
mean(train_data$am == train_data$pred)

## test model

p_test <- predict(logit_model,newdata = test_data, type =
test_data$pred <- if_else(p_test >= 0.5, "Manual", "Auto")
mean(test_data$am == test_data$pred)
```