

### กิจกรรมที่ 3 : SVM Hyperparameter Tuning

3.1 : การทดลองจัดกลุ่มข้อมูล (Classification) ด้วย RBF Kernel สำหรับ SVM แบบ Hard Decision (No Soft Margin) และการเลือกพารามิเตอร์ที่เหมาะสม (จาก Activity: Classification (RBF kernel hard decision))

- Library ที่ใช้

```
import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix, classification_report
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

- อ่านไฟล์ข้อมูล “data\_sample.csv”
- เก็บข้อมูลที่อ่านใน Pandas DataFrame โดยกำหนดให้ใส่ชื่อคอลัมน์ เพื่อความเข้าใจข้อมูล
- สร้างฟังก์ชันคำนวณ RBF Kernel จาก support vectors (All Training Data:  $x_i$ )
  - $\text{RBF\_Kernel}(x_j, x_i)$

$$K_i(x_j, x_i) = \exp(-\gamma \|x_j - x_i\|^2)$$

$$\hat{y} = y_{\text{predict}} = h(\theta) = \theta_0 + \sum_{i=1}^N \theta_i K_i$$

$$\hat{y}_{\text{class}} = \begin{cases} 1 & \text{sign}(\hat{y}) \geq 0 \\ 0 & \text{sign}(\hat{y}) < 0 \end{cases}$$

กำหนดพารามิเตอร์  $\theta_0 - \theta_N$  ( $N$ : จำนวน *Training Data*)

- ทำการประมาณค่า  $\hat{y} = y_{\text{predict}}$  และ  $\hat{y}_{\text{class}}$  ของ  $X_{\text{train}}$  และ  $X_{\text{test}}$  โดยใช้ฟังก์ชัน RBF และพารามิเตอร์  $\theta_0 - \theta_N$  ที่กำหนดด้านบน
- สร้างฟังก์ชันและทำการคำนวณค่าตาราง Confusion Matrix, Precision, และ Recall

**NOTE** : From scratch only.

	Predicted Negative	Predicted Positive
Actual Negative	True Negative (TN)	False Positive (FP)
Actual Positive	False Negative (FN)	True Positive (TP)

- ตอบคำถามท้ายการทดลอง

### 3.2 : สร้าง Regression Model เพื่อประมาณราคาหุ้น Microsoft

- Library ที่ใช้

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn import preprocessing
from sklearn import metrics
import pandas_datareader.data as web
import yfinance as yf
from sklearn import model_selection
```

- อ่านข้อมูล

- ทำเช่นเดียวกับ Lab#2
- ราคาใกล้ปิด (Adj close) ของหุ้นกลุ่ม Technology 3 ตัว
  - `stk_tickers = ['MSFT', 'IBM', 'GOOGL']` จาก yahoo finance
- อัตราแลกเปลี่ยนสกุลเงิน
  - `ccy_tickers = ['USD/JPY', 'GBP/USD']` จาก fred
    - USD/JPY (US dollar , Japan Yen)
    - GBP/USD (British Pound, US Dollar)
- ค่าดัชนีตลาดหุ้น
  - `idx_tickers = ['S&P500', 'Dowjones', 'VIX']` จาก fred
- ในช่วงวัน '2018-12-31' ถึง วันปัจจุบัน
- ทำ Data Exploration เพื่อดูรายละเอียดของข้อมูล และ Standardization โดยเปลี่ยนข้อมูล date column ให้เป็น index และไม่ต้องทำ standardization สำหรับ index

- เตรียมข้อมูล

- ทำเช่นเดียวกับ Lab#2
  - ราคาหุ้น Microsoft ที่ต้องการทำนาย (base)
  - ราคาหุ้น Microsoft เป็นคำตอบการทำนายในอีก 3 วันข้างหน้า (Y: y real)
  - ผลต่างราคาหุ้น Microsoft วันปัจจุบัน กับ ย้อนหลัง  $k \times \text{return\_period}$  วัน
    - $k = 3$  จะได้ X4\_3DT ผลต่างราคาย้อนหลัง  $3 \times 3 = 9$  วัน
    - $k = 6$  จะได้ X4\_6DT ผลต่างราคาย้อนหลัง  $6 \times 3 = 18$  วัน
    - $k = 12$  จะได้ X4\_12DT ผลต่างราคาย้อนหลัง  $12 \times 3 = 36$  วัน
  - สามารถใช้ฟังก์ชันของ pandas dataframe `df.diff(period)` เพื่อคำนวณค่าผลต่างราคาปัจจุบันกับย้อนหลัง  $k$  วัน = `df.diff(k)`  
หรือ ผลต่างราคาปัจจุบันกับอนาคต  $k$  วัน = `df.diff(k).shift(-k)`  
เลือกอย่างใดอย่างหนึ่ง เพื่อใช้เป็น feature สร้างความสัมพันธ์ระหว่าง row ข้อมูล ที่มี  
ความสัมพันธ์ตามลำดับเวลา
  - ราคาหุ้น Google และ IBM (X1)
  - อัตราแลกเปลี่ยนสกุลเงิน (ccy\_tickers: X2)
  - ค่าดัชนีตลาดหุ้น (idx\_tickers: X3)

- รวม X4 = [X4\_3DT, X4\_6DT, X4\_12DT] ตามแนว column (index = 1)
- รวม X = [X1, X2, X3, X4] เป็นข้อมูล feature input ตามแนว column (index = 1)
- รวม dataset = [X, Y] เป็นชุดข้อมูลทั้งหมด ตามแนว column (index = 1)
- ทำ Data Cleansing dataset ซึ่งประกอบด้วย X,Y ไปพร้อมกัน เช่นการจัดการ NA
- แสดง Data Exploration เพื่อดู missing values (NA) แก้ไขครบถ้วนหรือไม่
- แยกข้อมูล X กับ Y จากข้อมูล dataset
- ทำ Feature selection โดย Drop column ของ X ที่มีค่า correlation > 0.9
- แสดงคอลัมน์ที่ถูก Drop และ ข้อมูลที่เหลืออยู่ของ X
- เตรียมข้อมูล train 70% test 30% (train\_test\_split())
- Initialize Regression Model
  - LinearRegression()
  - Support Vector Regression (SVR())
- กำหนดพารามิเตอร์ของ Hyperparameter Tuning
  - GridSearchCV
    - เตรียม parameter\_dict ของ LinearRegression { 'fit\_intercept' = [True,False] }
    - เตรียม parameter\_dict ของ SVR parameters { 'kernel': [ ] , 'C': [ ] , 'gamma': [ ] , 'degree': [ ] }
    - โดยพารามิเตอร์ C, gamma ให้ใช้ตามที่กำหนดในตาราง
  - RandomizedSearchCV
    - เตรียม parameter\_dict ของ LinearRegression { 'fit\_intercept' = [True,False] }
    - เตรียม parameter\_dict ของ SVR parameters { 'kernel': [ ] , 'C': [ ] , 'gamma': [ ] , 'degree': [ ] }
    - โดยพารามิเตอร์ C, gamma ให้สุ่มหยิบในช่วงที่กำหนดในตาราง

[ตารางพารามิเตอร์ของแต่ละกลุ่ม](#) (link)

- ทำ Hyperparameter Tuning เพื่อได้ Best parameters โดยกำหนด Search parameters
  - GridSearchCV parameters: model, cv=k (K-Fold), param\_grid = parameter\_dict, scoring='neg\_mean\_squared\_error'
  - RandomizedSearchCV parameters: model, cv=k (K-Fold), param\_grid = parameter\_dict, n\_iter = N (number of parameters set), scoring= neg\_mean\_squared\_error'
- สร้างโมเดลและทำการ Train ด้วย Best Parameters ที่ได้จาก Hyperparameter Tuning
- แสดงกราฟราคาหุ้นตามเวลาเพื่อเปรียบเทียบทิศทางการราคาใน training และ testing data ผลลัพธ์จากการทำ Prediction จากแต่ละ model
- ตอบคำถามท้ายการทดลอง

### 3.3 : สร้าง Classification Model เพื่อประมาณระดับคุณภาพเมล็ดกาแฟ

- Library ที่ใช้

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVC
from sklearn import preprocessing
from sklearn import metrics
import pandas_datareader.data as web
from sklearn import model_selection
pd.options.display.float_format = '{:.3f}'.format
```

- อ่านข้อมูลจากไฟล์ "Coffee-modified.csv"
- เก็บข้อมูลที่อ่านใน Pandas DataFrame โดยกำหนดให้ใส่ชื่อคอลัมน์ เพื่อความเข้าใจข้อมูล
- ทำ Data Exploration เพื่อดูรายละเอียดของข้อมูล และ missing values (NA)
- ทำ Data Cleansing เพื่อแก้ไขปัญหาที่อาจเกิดขึ้นในข้อมูล
- เตรียมข้อมูล
  - แยก Y เป็นข้อมูลคอลัมน์ "Total.Cup.Points" และ X เป็นคอลัมน์ที่เหลือ
  - ทำการแบ่งระดับคุณภาพของเมล็ดกาแฟ จากคะแนน "Total.Cup.Points" ตามเงื่อนไขทางสถิติดังนี้
    - Labeling Bean\_grade value using percentile\*\*
      - Bean\_grade = 1 ; 'if Y < rating\_pctile[0] 75 percentile'
      - Bean\_grade = 2 ; 'if rating\_pctile[0] <= Y < rating\_pctile[1] 90 percentile'
      - Bean\_grade = 3 ; 'if Y >= rating\_pctile[1]'
  - แสดงกราฟแท่งเปรียบเทียบปริมาณข้อมูลในแต่ละ Bean\_grade
  - ทำ Standardization ของข้อมูล X
  - ทำ Feature selection โดย Drop column ของ X ที่มีค่า correlation > 0.8
    - แสดงคอลัมน์ที่ถูก Drop และ ข้อมูลที่เหลืออยู่ของ X
  - แปลงข้อมูล categorical data (non-numeric column) ให้เป็นตัวเลข ด้วย One Hot encoding
- เตรียมข้อมูล train 70% test 30% (train\_test\_split())
- Initialize Regression Model
  - Support Vector Machine for Classification (SVC())
- กำหนดพารามิเตอร์ของ Hyperparameter Tuning
  - GridSearchCV()
    - เตรียม parameter\_dict ของ  
SVC parameters: { 'kernel': ['linear','rbf','poly'] , 'C': [], 'gamma': [], 'degree': []}  
โดยพารามิเตอร์ C, gamma ให้ใช้ตามที่กำหนดในตารางของกลุ่ม
  - RandomizedSearchCV()

- เตรียม parameter\_dict ของ

SVC parameters {'kernel': [ 'linear','rbf','poly' ] , 'C': [random at least 5 ] ,  
'gamma': [random at least 5 ] , 'degree': [ ] }

โดยพารามิเตอร์ C, gamma ให้สุ่มหยิบในช่วงที่กำหนดในตารางของกลุ่ม

- RandomizedSearchCV() parameters:

model, scoring='accuracy', param\_grid=parameter\_dict, n\_iter  
cv=StratifiedKFold(n\_splits=k, shuffle=True, random\_state)

— ตารางพารามิเตอร์ของแต่ละกลุ่ม (link)

- ทำ Hyperparameter Tuning เพื่อได้ Best parameters โดยกำหนด Search parameters

— GridSearchCV parameters: model, scoring='accuracy', param\_grid=parameter\_dict,  
cv=StratifiedKFold(n\_splits=k, shuffle=True, random\_state)

— RandomizedSearchCV parameters: model, param\_grid = parameter\_dict,  
n\_iter = N (number of parameters set) scoring='accuracy',  
cv=StratifiedKFold(n\_splits=k, shuffle=True, random\_state)

- สร้างโมเดลและทำการ Train ด้วย Best Parameters ที่ได้จาก Hyperparameter Tuning
- สร้างฟังก์ชันและทำการคำนวณค่า Model Performance: Confusion Matrix, Classification Report
- แสดงผลการคำนวณ Model Performance ของแต่ละคลาส ในรูปของตาราง
- ตอบคำถามท้ายการทดลอง

## การส่งงาน

1. ให้ Staff ตรวจในห้อง (อย่างน้อยข้อ 3.1)
2. ส่งเอกสารในฟอร์มส่ง Lab (<https://forms.gle/pWUx2vHrZq29Axx8>)
  - 2.1 source code
  - 2.2 เอกสาร (pdf) อธิบายการทำงานของ source code (ถ้าไม่ได้อธิบายในห้อง)
  - 2.3 การตั้งชื่อไฟล์ “Lab#3\_ชื่อกลุ่ม\_รหัสสมาชิก#1\_รหัสสมาชิก#2.pdf”
3. กำหนดส่ง ก่อนวันทำแลปครั้งต่อไป