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
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

- อ่านไฟล์ข้อมูล “data_sample.csv”
- เก็บข้อมูลที่อ่านใน Pandas DataFrame โดยกำหนดให้ใส่ชื่อคอลัมน์ เพื่อความเข้าใจข้อมูล
- สร้างฟังก์ชันคำนวน RBF Kernel จาก support vectors (All Training Data: x_i)
  - RBF_Kernel(x_j, x_i)

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

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

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

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

- ทำการประมาณค่า $\hat{y} = y_{predict}$ และ $\hat{y}_{class}$ ของ X_train และ X_test โดยใช้ฟังก์ชัน RBF และพารามิเตอร์ $\theta_0 - \theta_N$ ที่กำหนดด้านบน
- สร้างฟังก์ชันและทำการคำนวนค่าตาราง 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 x return_period วัน
      - k = 3 จะได้ X4_3DT ผลต่างราคาย้อนหลัง 3 x 3 = 9 วัน
      - k = 6 จะได้ X4_6DT ผลต่างราคาย้อนหลัง 6 x 3 = 18 วัน
      - k = 12 จะได้ X4_12DT ผลต่างราคาย้อนหลัง 12 x 3 = 36 วัน
      - สามารถใช้ฟังก์ชันของ ของ pandas dataframe df.diff(period) เพื่อคำนวนค่า
        ผลต่างราคาปัจจุบันกับย้อนหลัง k วัน = df.diff(k)
        หรือ ผลต่างราคาปัจจุบันกับอนาคต k วัน = df.diff(k).shift(-k)
        เลือกอย่างใดอย่างหนึ่ง เพื่อใช้เป็น feature สร้างความสัมพันธ์ระหว่าง row ข้อมูล ที่มีความสัมพันธ์ตามลำดับเวลา
      - 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) แก้ไขครบถ้วยหรือไม่
- 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
- แสดงกราฟราคาหุ้นตามเวลาเพื่อเปรียบเทียบทิศทางราคาใน 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
- สร้างฟังก์ชันและทำการคำนวนค่า Model Performance: Confusion Matrix, Classification Report
- แสดงผลการคำนวน Model Performance ของแต่ละคลาส ในรูปของตาราง
- ตอบคำถามท้ายการทดลอง

การส่งงาน

1. ให้ Staff ตรวจในห้อง (อย่างน้อยข้อ 3.1)
2. ส่งเอกสารในฟอร์มส่ง Lab (https://forms.gle/pWUx2vHrZq29Axnx8)

   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