ชื่อ-สกุล_____ห้อง_____รหัสนักศึกษา_____

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
วิชา Machine Learning Laboratory

การทดลองที่ 3 : การทดลองปรับค่าพารามิเตอร์เพื่อสร้างโมเดลการจัดกลุ่มข้อมูลด้วยเทคนิคการจัดกลุ่มข้อมูลแบบต่างๆและเปรียบเทียบประสิทธิภาพของโมเดลทดสอบ

วัตถุประสงค์

1. เพื่อศึกษาและทดลองการใช้งานโมเดลการจัดกลุ่มข้อมูลแบบต่างๆ
2. เพื่อศึกษาและทดลองการปรับค่าพารามิเตอร์ที่เหมาะสมกับโมเดลสำหรับชุดข้อมูลทดสอบ
3. เพื่อศึกษาและทดลองเทคนิคการวัดประสิทธิภาพโมเดลการจัดกลุ่มข้อมูล
4. เพื่อศึกษาและทดลองการเปรียบเทียบประสิทธิภาพของโมเดลต่างๆเชิงกราฟ

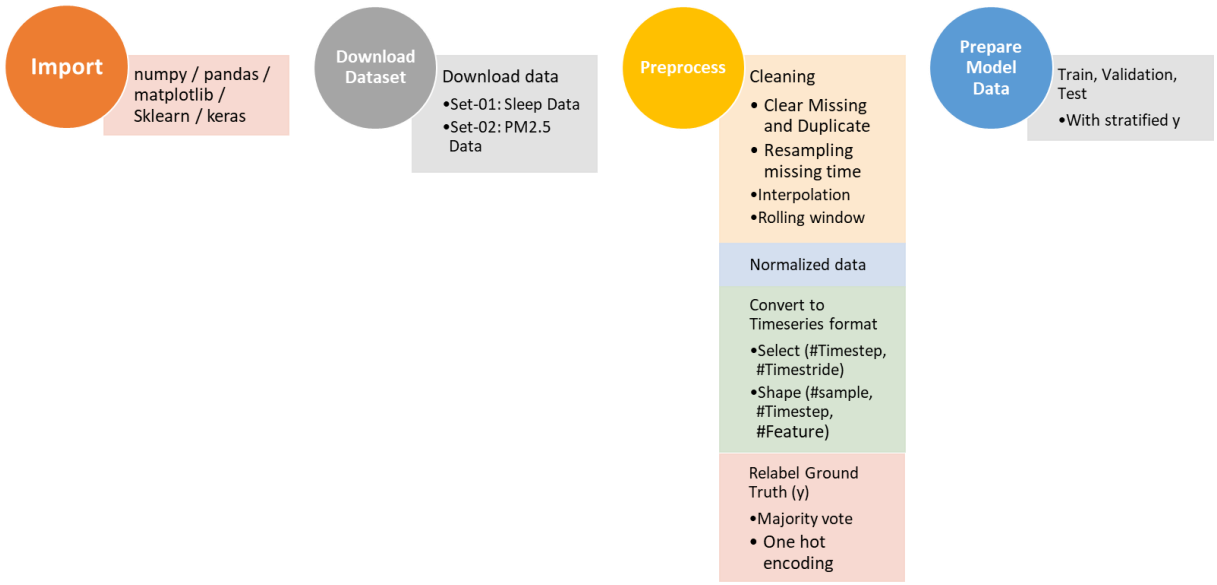
อุปกรณ์ และเครื่องมือที่ใช้ในการทดลอง

1. โปรแกรม python

ข้อกำหนดในการตรวจการทดลอง

1. แสดง source code และภาพผลการทดลองที่ทำพร้อมอธิบาย โดยกลุ่มใดพร้อม ให้ Post แจ้งชื่อกลุ่มและ Email ที่ต้องการให้ติดต่อไว้ใน Facebook group เพื่อตรวจการทดลองตามหัวข้อส่งงาน โดยจะทำการตรวจผ่าน google hangout -> share screen ตามลำดับที่แจ้งไว้
2. นศ.ที่ได้รับการตรวจจากอาจารย์เรียบร้อย อาจารย์จะเช็คส่งงานในระบบ
3. ให้นศ. นำ source code และ ภาพ figure ที่ให้แสดงทุกภาพ โพสลงใน google form พร้อมตอบคำถามท้ายการทดลอง ส่งภายในวันที่ 31 มีค. 2563 เวลา 17.00 น.

ตอนที่ 1: การทดลองเตรียมข้อมูล ปรับค่าข้อมูล และจัดแบ่งชุด Train, Test เพื่อสอนโมเดล



1.1 Import Lib (numpy, pandas, matplotlib, sklearn, keras)

1.2 โหลดข้อมูล Timeseries Dataset file

ข้อมูลมี 2 ชุด ให้นศ.โหลดข้อมูลในชุดที่กลุ่มตนเองได้รับ

Set#1: Sleep data

AccX, AccY, AccZ,

Heartrate,

Sleep_Stage: (0-5, wake = 0, N1 = 1, N2 = 2, N3 = 3, REM = 5)

Set#2: PM2.5 data (เลือก 4 features)

HUMI: Humidity (%)

PRES: Pressure (hPa)

Iws: Cumulated wind speed (m/s)

Iprec: Cumulated precipitation (mm)

PM2.5_level: Level#0: particle concentration < 10 ug/m^3

Level#1: particle concentration 10 – 25 ug/m^3

Level#0: particle concentration > 25 ug/m^3

1.3 Preprocess data

1.3.1 Cleaning

Clear Missing and Duplicate data,

Resampling and interpolated data,

Moving average (Rolling window)

1.3.2 แสดง**รูปภาพ**เปรียบเทียบข้อมูลก่อนและหลัง Cleaning ในรูปของ

- Feature (Y-axis) – Time (X-axis)
- Statistical Plot

```
import seaborn as sns
g = sns.PairGrid(df2,hue='models', hue_kws={"cmap":list_of_cmaps},)
g.map_upper(plt.scatter)
g.map_lower(sns.kdeplot)
g.map_diag(sns.distplot)
```

1.3.3 Normalizing data

1.3.3 Convert sample-based data to timeseries data format

Select #Timestep, #Timestride

Organizing to shape -> (#ชุด timeseries, #Timestep, #features)

1.4 Prepare Label Ground Truth (y) สำหรับแต่ละชุด timeseries ด้วย majority vote และ ทำ **One-Hot Encoding** โดยจัดข้อมูล ground truth จาก 1D -> binary N-D class เช่น ข้อมูลเดิม y มี 3 class (1,2,3) ให้จัดรูปแบบ y ใหม่ เป็น binary output ซึ่งให้ค่า ตำแหน่ง class คำตอบที่ถูกต้องเป็น 1 สำหรับ class ที่ผิดที่เหลือจะมีค่าเป็น 0 โดยใช้

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

→

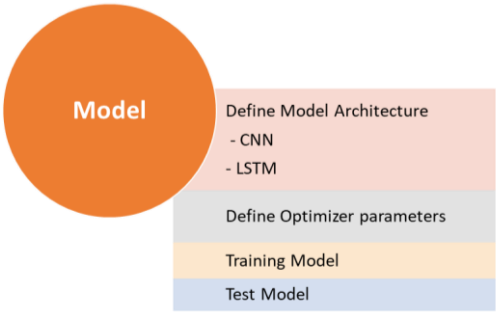
One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

1.5 Prepare training, validation, and test data (Train_test_split())

stratify = y, กำหนด random seed,

ตอนที่ 2: การทดลองสร้าง สอน และ ทดสอบโมเดล เพื่อดูค่าความแม่นยำเบื้องต้นของแต่ละโมเดล (CNN, LSTM)



2.1 กำหนดโครงสร้างโมเดล (model architecture) ที่ต้องการนำมาทดสอบ

```
# keras CNN:
    model = Sequential()
    #L1
        model.add(Conv2D(CNN_L1,kernel_size=Ker_size,
            activation=Act_func, input_shape=Input_shape,
            padding='same'))
        # Optional: model.add(BatchNormalization())
        model.add(MaxPooling2D(pool_size=P_size))
        model.add(Dropout(Prate))
    ....
    #Ln
        model.add(Conv2D(CNN_L,n kernel_size=Ker_size, activation=Act_func,
        input_shape=Input_shape,  padding='same'))
        # Optional: model.add(BatchNormalization())
        model.add(MaxPooling2D(pool_size=P_size))
        model.add(Dropout(Prate))

    model.add(Flatten())
    model.add(Dense(Dense_size , activation= Act_func ))
    model.add(Dense(#output_class, activation='sigmoid'))
    model.summary()

# keras LSTM:
    model = Sequential()
    #L1
        model.add(LSTM(n_hidden1, input_shape=(t_window, Nfeature)))
        # Optional: model.add(BatchNormalization())
        model.add(Dropout(Prate))
    ....
    #Ln
        model.add(LSTM(n_hiddenN))
```

```
# Optional: model.add(BatchNormalization())
model.add(Dropout(pvN))


model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

2.2 กำหนด Optimizer Parameters

```
model.compile(      loss='categorical_crossentropy',
                    optimizer='Adam', metrics=['accuracy'])
```

2.3 ทำการสอนโมเดล (Training model)

```
# Training the model
history = model.fit(    X_train, y_train,
                        batch_size=batch_size,
                        validation_data=(X_validate, y_validate),
                        epochs=epochs)
```

2.4 ทำการ predict ค่าจากโมเดลทั้ง 2 แบบ (CNN, LSTM) ด้วยข้อมูล X_test, y_test ที่แบ่งไว้

```
y_prediction = model. predict(X_test)
y_pred_single = [np.argmax(p) for p in y_prediction]
y_test_single=[np.argmax(p) for p in y_test]
```

2.5 คำนวณค่าตัววัดประสิทธิภาพของการทำนายจากโมเดล CNN และ LSTM โดยวัดค่า classification report/ confusion matrix

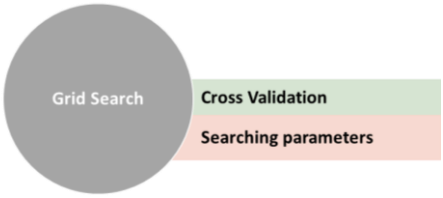
```
classification_report(y_test_single, y_pred_single)
confusion_matrix(y_test_single, y_pred_single)
```

2.6 แสดง**รูปภาพ**ของ classification_report และ confusion_matrix

_____

(อาจารย์ตรวจผลการทดลอง)

ตอนที่ 3: การทดลองการค้นหาพารามิเตอร์ที่ดีที่สุดสำหรับโมเดล



3.1 สร้างโมเดลรูปแบบ cross validation ที่ต้องการใช้ ในที่นี้ใช้ StratifiedKFold

```
seed  #กำหนดเพื่อให้ random ได้ชุดข้อมูลเดียวกันในทุกครั้งที่รัน
k # กำหนดจำนวนชุดข้อมูลย่อยที่จะถูกนำมาทำ StratifiedKFold Cross Validation

skfold = model_selection.StratifiedKFold(n_splits=k, shuffle = True,
        random_state=seed)
```

3.2 กำหนดรายการพารามิเตอร์ทั้งหมดที่ต้องการทดสอบหาค่าที่ดีที่สุดของโมเดล

```
batch_size = [10, 20, 40, 60, 80, 100]
epochs = [10, 50, 100]
optimizers = ['adam','adadelta']
param_grid = dict(batch_size=batch_size, epochs=epochs, optimizer=optimizers)
```

3.3 ใช้ฟังก์ชั่น GridSearchCV เพื่อทำ cross validation หาค่าพารามิเตอร์ที่ดีที่สุด

```
# ใช้ cross validation (cv) เป็น StratifyKfold ที่กำหนดไว้ในตอนข้อ 3.1
gsCV_model = model_selection.GridSearchCV(        estimator=model,
                                        param_grid=param_grid, n_jobs=2, cv=skfold)
```

3.4 นำค่าพารามิเตอร์ที่ดีที่สุดไปสอนโมเดล

```
        gsCV_model.fit(X_train, y_train)
```

3.5 save ผลลัพธ์จากการทำ GridSearchCV ลงบนไฟล์ .csv

```
        gridsearch_result = pd.DataFrame( gsCV_model.cv_results_)

        gridsearch_result.to_csv('filename.csv')
```

3.6 แสดงค่า**กราฟ**แท่งพารามิเตอร์ที่ดีที่สุด

```
        gsCV_model.best_params_, gsCV_model.best_score
```

3.7 ใช้โมเดลที่สอนจากพารามิเตอร์ที่ดีที่สุดมา predict ข้อมูล ชุด x_test

```
        y_predict_test = gsCV_model.predict(X_test)
```

3.8 คำนวณค่าตัววัดประสิทธิภาพของการทำนายจากโมเดลที่ทดสอบด้วย y _test

classification_report(y_future_test, y_pred_future)

confusion_matrix(y_future_test, y_pred_future)

3.9 แสดง**รูปภาพ**ของ classification_report และ confusion_matrix

_____

(อาจารย์ตรวจผลการทดลอง)