SWINBURNE UNIVERSITY OF TECHNOLOGY

COS40007

Portfolio Week 3

Studio 3

(DUE 20/01/25 - 00:00 A.M)

NAME & STUDENTID	:	**Nguyen Gia Binh - 104219428**
CLASS	:	**COS40007 - AI for Engineering**
LECTURER	:	**Dr. Bui Ngoc Dung** (dbui@swin.edu.au)
TUTOR	:	**Dr. Bui Ngoc Dung** (dbui@swin.edu.au)

Hanoi, January 19, 2025

TABLE OF CONTENT

1) Studio 3

Activity 6 table:

SVM model	Train-test split	Cross validation
Original feature	88.82%	89.18%
With hyper parameter tuning	84.20%	84.26%
With feature selection and hype parameter tuning	85.38%	85.61%
With PCA and hyper parameter tuning	84.21%	84.36%

Activity 7 table:

Model	Train-test split	Cross validation
SVM	88.82%	89.18%
SGD	61.13%	87.83%
RandomForest	90.37%	92.54%
MLP	82.43%	87.02%

2) Portfolio

a) Data collection

```
boning = pd.read_csv('/kaggle/input/ampc2-dataset/Boning.csv')
slicing = pd.read_csv('/kaggle/input/ampc2-dataset/Slicing.csv')
```

```
columns = ['Frame', 'Right Upper Arm x', 'Right Upper Arm y', 'Right Upper Arm z',
                      'Left Upper Arm x', 'Left Upper Arm y', 'Left Upper Arm z']
boning_data = boning[columns]
slicing_data = slicing[columns]
```

```python
print(boning_data)
```

```
       Frame  Right Upper Arm x  Right Upper Arm y  Right Upper Arm z  \
0          0           0.559333           0.024451           0.523876   
1          1           0.364502           0.174362           0.578967   
2          2          -0.041012           0.134000           0.285496   
3          3           0.007940           0.223349           0.133206   
4          4           0.418177           0.374025           0.080194   
...      ...                ...                ...                ...   
54175  54175          -0.859456          -0.295704          -2.242624   
54176  54176           0.154228           0.177665          -2.178331   
54177  54177           0.121470          -0.017822          -1.919329   
54178  54178          -0.145355           0.002809          -1.044747   
54179  54179          -0.177527           0.383617          -0.046803   

       Left Upper Arm x  Left Upper Arm y  Left Upper Arm z  
0              0.005296         -0.301528         -0.231850  
1             -0.139028          0.029267          0.051904  
2              0.066277         -0.211549          0.132967  
3              0.173529         -0.107682          0.040102  
4              0.017176         -0.232074          0.278534  
...                 ...               ...               ...  
54175         -0.621395          0.651749         -0.907248  
54176          0.510283          0.431242         -0.824876  
54177          0.681681          0.251320         -0.406565  
54178          0.427238          0.112021         -0.055451  
54179          0.548915          0.148727          0.108037  

[54180 rows x 7 columns]
```

```python
print(slicing_data)
```

```
       Frame  Right Upper Arm x  Right Upper Arm y  Right Upper Arm z  \
0          0          -0.081934          -0.063509          -0.194105   
1          1          -0.017001           0.060680          -0.165873   
2          2          -0.097286           0.002338          -0.117991   
3          3          -0.150787          -0.041678          -0.051735   
4          4          -0.180658          -0.111853          -0.084678   
...      ...                ...                ...                ...   
17875  17875          -0.114328           0.439828          -0.081698   
17876  17876          -0.370897           0.463411          -0.037791   
17877  17877          -0.495828           0.108677          -0.183645   
17878  17878          -0.473204          -0.212558          -0.433903   
17879  17879          -0.337285          -0.520867          -0.449577   

       Left Upper Arm x  Left Upper Arm y  Left Upper Arm z  
0              0.029982         -0.124462          0.040935  
1              0.067401         -0.042730          0.058972  
2              0.067550         -0.074310          0.094963  
3              0.075417         -0.134344          0.106930  
4             -0.000695         -0.187848          0.029711  
...                 ...               ...               ...  
17875          0.178778          0.022925          0.315181  
17876          0.047531         -0.041651          0.352262  
17877         -0.168810         -0.536649          0.279052  
17878         -0.125721         -0.807797          0.018725  
17879         -0.285581         -0.778978         -0.134281  

[17880 rows x 7 columns]
```

```
combined = [(boning_data, 0), (slicing_data, 1)]

combined_data = pd.concat([data.assign(Class=label) for data, label in combined], ignore_index=True)
```

```
print(combined_data)
```

```
       Frame  Right Upper Arm x  Right Upper Arm y  Right Upper Arm z  \
0          0           0.559333           0.024451           0.523876
1          1           0.364502           0.174362           0.578967
2          2          -0.041012           0.134000           0.285496
3          3           0.007940           0.223349           0.133206
4          4           0.418177           0.374025           0.080194
...      ...                ...                ...                ...
72055  17875          -0.114328           0.439828          -0.081698
72056  17876          -0.370897           0.463411          -0.037791
72057  17877          -0.495828           0.108677          -0.183645
72058  17878          -0.473204          -0.212558          -0.433903
72059  17879          -0.337285          -0.520867          -0.449577

       Left Upper Arm x  Left Upper Arm y  Left Upper Arm z  Class
0              0.005296         -0.301528         -0.231850      0
1             -0.139028          0.029267          0.051904      0
2              0.066277         -0.211549          0.132967      0
3              0.173529         -0.107682          0.040102      0
4              0.017176         -0.232074          0.278534      0
...                 ...               ...               ...    ...
72055          0.178778          0.022925          0.315181      1
72056          0.047531         -0.041651          0.352262      1
72057         -0.168810         -0.536649          0.279052      1
72058         -0.125721         -0.807797          0.018725      1
72059         -0.285581         -0.778978         -0.134281      1

[72060 rows x 8 columns]
```

b) Composite column

I didnt read the requirement carefully enough and since my friend is working on number 4, i feel fit to also do number 4

Set 1:

combined_data['RMS_right_xy'] = np.sqrt((combined_data['Right Upper Arm x']**2 + combined_data['Right Upper Arm y']**2) / 2)

combined_data['RMS_right_yz'] = np.sqrt((combined_data['Right Upper Arm y']**2 + combined_data['Right Upper Arm z']**2) / 2)

combined_data['RMS_right_zx'] = np.sqrt((combined_data['Right Upper Arm z']**2 + combined_data['Right Upper Arm x']**2) / 2)

combined_data['RMS_right_xyz'] = np.sqrt((combined_data['Right Upper Arm x']**2 + combined_data['Right Upper Arm y']**2 + combined_data['Right Upper Arm z']**2) / 3)

combined_data['right_roll'] = 180 * np.arctan2(combined_data['Right Upper Arm y'], np.sqrt(combined_data['Right Upper Arm x'] ** 2 + combined_data['Right Upper Arm z'] ** 2)) / np.pi

combined_data['right_pitch'] = 180 * np.arctan2(combined_data['Right Upper Arm x'], np.sqrt(combined_data['Right Upper Arm y'] ** 2 + combined_data['Right Upper Arm z'] ** 2)) / np.pi

Set 2:

```
combined_data['RMS_left_xy'] = np.sqrt((combined_data['Left Upper Arm x']**2 +
combined_data['Left Upper Arm y']**2) / 2)

combined_data['RMS_left_yz'] = np.sqrt((combined_data['Left Upper Arm y']**2 +
combined_data['Left Upper Arm z']**2) / 2)

combined_data['RMS_left_zx'] = np.sqrt((combined_data['Left Upper Arm z']**2 +
combined_data['Left Upper Arm x']**2) / 2)

combined_data['RMS_left_xyz'] = np.sqrt((combined_data['Left Upper Arm x']**2 +
combined_data['Left Upper Arm y']**2 + combined_data['Left Upper Arm z']**2) / 3)

combined_data['left_roll'] = 180 * np.arctan2(combined_data['Left Upper Arm y'],
np.sqrt(combined_data['Left Upper Arm x'] ** 2 + combined_data['Left Upper Arm z'] ** 2)) /
np.pi

combined_data['left_pitch'] = 180 * np.arctan2(combined_data['Left Upper Arm y'],
np.sqrt(combined_data['Left Upper Arm x'] ** 2 + combined_data['Left Upper Arm z'] ** 2)) /
np.pi
```

```
combined_data.head()
```

	Frame	Right Upper Arm x	Right Upper Arm y	Right Upper Arm z	Left Upper Arm x	Left Upper Arm y	Left Upper Arm z	Class	RMS_right_xy	RMS_right_yz	RMS_right_zx	RMS_right_xyz	right_roll	right_pitch	RMS_left_xy	RMS_left_yz	RMS_left_zx	RMS_left_xyz	left_roll	left_pitch
0	0	0.559333	0.024451	0.523876	0.005296	-0.301528	-0.231850	0	0.395886	0.370839	0.541895	0.442680	1.827454	46.843726	0.213245	0.268955	0.163986	0.219622	-52.435408	-52.435408
1	1	0.364502	0.174362	0.578967	-0.139028	0.029267	0.051904	0	0.285713	0.427554	0.483769	0.407622	14.297917	31.082800	0.100462	0.042134	0.104935	0.087330	11.156428	11.156428
2	2	-0.041012	0.134000	0.285496	0.066277	-0.211549	0.132967	0	0.099090	0.223007	0.203949	0.183617	24.918982	-7.409126	0.156757	0.176682	0.105054	0.149249	-54.919971	-54.919971
3	3	0.007940	0.223349	0.133206	0.173529	-0.107682	0.040102	0	0.158031	0.183887	0.094358	0.150213	59.143265	1.748919	0.144409	0.081251	0.125938	0.120161	-31.157291	-31.157291
4	4	0.418177	0.374025	0.080194	0.017176	-0.232074	0.278534	0	0.396716	0.270486	0.301084	0.327209	41.296426	47.549452	0.164550	0.256359	0.197328	0.209551	-39.747551	-39.747551

c) Data processing

```
FRAMES_PER_MINUTE = 60

all_features = []

for start in range(0, len(combined_data), FRAMES_PER_MINUTE):
    end = start + FRAMES_PER_MINUTE
    if end > len(combined_data):
        break

    segment = combined_data.iloc[start:end]
    minute_features = {'Minute': start // FRAMES_PER_MINUTE + 1, 'Class':
segment['Class'].iloc[0]}

    for col in [c for c in combined_data.columns if c not in ['Frame', 'class']]:
        data = segment[col]
        minute_features.update({
            f'{col}_mean': data.mean(),
            f'{col}_std': data.std(),
            f'{col}_min': data.min(),
            f'{col}_max': data.max(),
            f'{col}_auc': simps(data),
            f'{col}_peaks': len(find_peaks(data)[0])
        })

    all_features.append(minute_features)

step3_data = pd.DataFrame(all_features)
```

d) Training

model	**Train-test split**	**Cross val**
1 and 2 with hyper parameter tuning	99.72%	96.50%
1 and 2 with hyper parameter tuning and 10 best features	77.28%	75.52%
1 and 2 with hyper parameter tuning and 10 principal components	77.84%	75.60%
SGD	100.00%	99.08%

Random Forest	99.72%	100.00%
MLP	96.95%	97.25%

e) Selection

Q1: Base on the result “1 and 2 with hyper parameter tuning” seems to be the best. I suspect because the varience between each feature is just too small. In my work, i actually have to remove some features because they have 0 variance across all other feature and would result in a division by 0

Q2: Random Forest would be the best, the testing result of Random Forest is the best in both Studio 3 and portfolio 3

4) **Appendix**

Here is the link to the kaggle notebook:

- Studio 3: https://www.kaggle.com/code/binhswinburnehn/cos40007-week3-studio
- Portfolio : https://www.kaggle.com/code/binhswinburnehn/cos40007-week-3-portfolio