กลุ่ม เด็กดีขี้เมา

60010113 คณิศร พิทักษ์วงศ์, 60010479 ธีระสาร มินทะขัด

ผลการทดลอง Lab 1

ตอนที่ 1: การทดลองอ่านไฟล์ข้อมูล การแก้ปัญหา ข้อมูลหาย และการปรับช่วงค่าของข้อมูล แสดงข้อมูลเชิง กราฟ และ การจัดเตรียมรูปแบบข้อมูลเพื่อนำเข้าโมเดล

1. ขั้นตอนการทดลองในการนำเข้าข้อมูล

import lib

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        from mpl_toolkits.mplot3d import Axes3D
        import seaborn as sns
```

read data file

```
In [2]: df = pd.read_csv('watch_test2_sample.csv')
```

แปลงชนิดของข้อมูล และทำการcopyข้อมูลไว้เพื่อใช้งานในการplot gps จากค่าจริง

```
In [3]: df['uts'] = pd.to_datetime(df['uts'])
        df.sort_values('uts', inplace = True)
        df_copy = df.copy()
```

In [1] ทำการ import library ที่จำเป็นต่อการทดลอง

In [2] อ่านไฟล์จาก .csv โดยใช้ pandas แล้วเก็บลง df

In [3] แปลง format ของ feature “uts” จาก string เป็น datetime และทำการ copy ข้อมูลไว้เพื่อใช้งานในการplot GPS จากค่าจริง

กลุ่ม เด็กดีขี้เมา

60010113 คณิศร พิทักษ์วงศ์, 60010479 ธีระสาร มินทะขัด

```
In [4]: df
```

Out[4]:

	uts	accelerateX	accelerateY	accelerateZ	compass	gps.x	gps.y	gyro.x	gyro.y	gyro.z	heartrate	light	pressure
0	2018-11-18 08:18:41+07:00	-3.957379	-14.204506	2.303692	355.85300	13.621563	100.369093	-1.304740	-2.642471	3.315061	117.0	12	1013.201
1	2018-11-18 08:18:41+07:00	-3.957379	-14.204506	2.303692	355.85300	13.621563	100.369093	-1.304740	-2.642471	3.315061	117.0	12	1013.201
2	2018-11-18 08:18:41+07:00	-3.957379	-14.204506	2.303692	355.85300	13.621563	100.369093	-1.304740	-2.642471	3.315061	117.0	12	1013.201
3	2018-11-18 08:18:41+07:00	-3.957379	-14.204506	2.303692	355.85300	13.621563	100.369093	-1.304740	-2.642471	3.315061	117.0	12	1013.201
4	2018-11-18 08:18:41+07:00	-3.957379	-14.204506	2.303692	355.85300	13.621563	100.369093	-1.304740	-2.642471	3.315061	117.0	12	1013.201
...	...	...	...	...	...	...	...	...	...	...	...	...	...
6265	2018-11-18 16:09:04+07:00	0.480334	-8.700976	-4.669516	232.32242	13.553518	100.279728	0.065982	-0.058532	0.052147	63.0	0	1008.726
6264	2018-11-18 16:09:04+07:00	0.480334	-8.700976	-4.669516	232.32242	13.553518	100.279728	0.065982	-0.058532	0.052147	63.0	0	1008.726
6274	2018-11-18 16:09:04+07:00	0.480334	-8.700976	-4.669516	232.32242	13.553518	100.279728	0.065982	-0.058532	0.052147	63.0	0	1008.726
6268	2018-11-18 16:09:04+07:00	0.480334	-8.700976	-4.669516	232.32242	13.553518	100.279728	0.065982	-0.058532	0.052147	63.0	0	1008.726
6275	2018-11-18 16:09:04+07:00	0.480334	-8.700976	-4.669516	232.32242	13.553518	100.279728	0.065982	-0.058532	0.052147	63.0	0	1008.726

6276 rows × 13 columns

ln [4] นำข้อมูลมาแสดงผลเพื่อตรวจสอบความถูกต้องของ feature ใน table

2. ทำการลดปัญหาข้อผิดพลาดของข้อมูล

In [5]:
```
df = df.drop_duplicates(keep="first")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 271 entries, 0 to 6273
Data columns (total 13 columns):
uts             271 non-null datetime64[ns, pytz.FixedOffset(420)]
accelerateX     267 non-null float64
accelerateY     267 non-null float64
accelerateZ     267 non-null float64
compass         270 non-null float64
gps.x           271 non-null float64
gps.y           271 non-null float64
gyro.x          268 non-null float64
gyro.y          269 non-null float64
gyro.z          269 non-null float64
heartrate       270 non-null float64
light           271 non-null int64
pressure        269 non-null float64
dtypes: datetime64[ns, pytz.FixedOffset(420)](1), float64(11), int64(1)
memory usage: 29.6 KB
```

ln [5] ทำการลบข้อมูลที่มีค่าซ้ำซ้อน โดยเลือกเก็บข้อมูลชุดแรกที่เจอ

In [6]:
```
df = df.fillna(df.median())
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 271 entries, 0 to 6273
Data columns (total 13 columns):
uts             271 non-null datetime64[ns, pytz.FixedOffset(420)]
accelerateX     271 non-null float64
accelerateY     271 non-null float64
accelerateZ     271 non-null float64
compass         271 non-null float64
gps.x           271 non-null float64
gps.y           271 non-null float64
gyro.x          271 non-null float64
gyro.y          271 non-null float64
gyro.z          271 non-null float64
heartrate       271 non-null float64
light           271 non-null int64
pressure        271 non-null float64
dtypes: datetime64[ns, pytz.FixedOffset(420)](1), float64(11), int64(1)
memory usage: 29.6 KB
```

ln [6] จัดการข้อมูลที่เป็น NaN โดยใช้ค่า median ของแต่ละ feature

60010113 คณิศร พิทักษ์วงศ์, 60010479 ธีระสาร มินทะขัด

```
In [8]: df = df.set_index('uts').interpolate(method="nearest")
        df
```

Out[8]:

uts	accelerateX	accelerateY	accelerateZ	compass	gps.x	gps.y	gyro.x	gyro.y	gyro.z	heartrate	light	pressure
2018-11-18 08:18:41+07:00	-3.957379	-14.204506	2.303692	355.85300	13.621563	100.369093	-1.304740	-2.642471	3.315061	117.0	12	1013.20100
2018-11-18 08:19:03+07:00	-0.038236	-1.156625	1.883101	355.85300	13.621482	100.369133	1.873036	2.521149	-1.295162	81.0	12	1013.16500
2018-11-18 08:19:45+07:00	1.906998	-4.361242	-4.358852	351.80853	13.621563	100.369088	-0.676847	-3.687540	1.123822	104.0	10	1013.21800
2018-11-18 08:20:13+07:00	-0.265259	-10.149148	3.042116	354.06730	13.621562	100.369103	-0.437397	1.558026	0.047890	122.0	10	1013.20795
2018-11-18 08:20:33+07:00	-2.098175	-11.195846	1.754056	354.06730	13.621545	100.369100	-2.849995	-1.562282	3.178840	78.5	11	1013.20600
...	...	...	...	...	...	...	...	...	...	...	...	...
2018-11-18	4.803340	0.050184	-8.263658	225.21982	13.551022	100.280217	-0.031927	-0.094716	0.037248	66.0	0	1008.64700

ln [8] ทำการแทรกข้อมูลในช่วงที่หายไปด้วยค่าใกล้เคียง โดยก่อนแทรกต้องกำหนด index เป็น feature ของ uts เพราะ ข้อมูลถูกเก็บแบบ timestamp ต้องแทรกด้วยช่วงเวลา

```
In [9]: #mva
        df = df.rolling("3s").mean()
        df
```

uts	accelerateX	accelerateY	accelerateZ	compass	gps.x	gps.y	gyro.x	gyro.y	gyro.z	heartrate	light	pressure
08:20:13+07:00												
2018-11-18 08:20:33+07:00	-2.098175	-11.195846	1.754056	354.06730	13.621545	100.369100	-2.849995	-1.562282	3.178840	78.5	11.0	1013.20600
...	...	...	...	...	...	...	...	...	...	...	...	...
2018-11-18 16:07:32+07:00	4.803340	0.050184	-8.263658	225.21982	13.551022	100.280217	-0.031927	-0.094716	0.037248	66.0	0.0	1008.64700
2018-11-18 16:07:58+07:00	1.780343	-6.609970	2.081448	228.24623	13.552975	100.280140	-1.199382	0.211781	-0.054275	70.0	99.0	1008.63794
2018-11-18 16:08:19+07:00	0.399084	-8.364026	-5.357756	230.59320	13.553595	100.279633	-0.019156	-0.062789	0.100037	66.0	114.0	1008.72500
2018-11-18 16:08:42+07:00	0.745593	-8.820463	-5.106835	232.32242	13.553518	100.279728	0.013835	0.034055	0.017028	57.0	44.0	1008.69300
2018-11-18 16:09:04+07:00	0.480334	-8.700976	-4.669516	232.32242	13.553518	100.279728	0.065982	-0.058532	0.052147	63.0	0.0	1008.72600

271 rows × 12 columns

ln [9] ลด noise ของข้อมูลด้วยการ moving average ทุก ๆ timestamp ที่ 3 วินาที

In [10]:
```
#scaler
means = df.mean()
stds = df.std()
```

In [11]:
```
df = (df - means) / stds
df
```

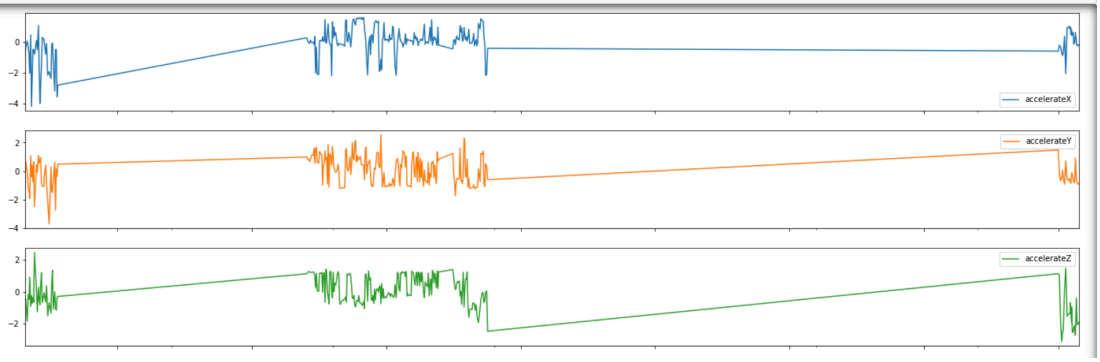
Out[11]:

uts	accelerateX	accelerateY	accelerateZ	compass	gps.x	gps.y	gyro.x	gyro.y	gyro.z	heartrate	light	pressure
2018-11-18 08:18:41+07:00	-1.118282	-2.036701	-0.353763	2.096929	1.876887	1.872687	-1.750671	-3.573744	4.089767	0.513937	-1.065275	0.077559
2018-11-18 08:19:03+07:00	-0.342711	0.672244	-0.448671	2.096929	1.876873	1.872688	2.391790	3.445797	-1.622874	-0.048798	-1.065275	0.076974
2018-11-18 08:19:45+07:00	0.042237	0.006915	-1.857185	2.061940	1.876887	1.872687	-0.932167	-4.994435	1.374548	0.310727	-1.075608	0.077836
2018-11-18 08:20:13+07:00	-0.387637	-1.194745	-0.187136	2.081481	1.876887	1.872688	-0.620026	2.136505	0.041335	0.592095	-1.075608	0.077672
2018-11-18 08:20:33+07:00	-0.750358	-1.412056	-0.477790	2.081481	1.876884	1.872688	-3.765023	-2.105312	3.920973	-0.087877	-1.070442	0.077641
...	...	...	...	...	...	...	...	...	...	...	...	...
2018-11-18 16:07:32+07:00	0.615403	0.922796	-2.738315	0.966798	1.864411	1.870558	-0.091467	-0.110269	0.028148	-0.283271	-1.127276	0.003507
2018-11-18 16:07:58+07:00	0.017173	-0.459956	-0.403913	0.992980	1.864757	1.870556	-1.613329	0.306389	-0.085261	-0.220745	-0.615764	0.003360
2018-11-18 16:08:19+07:00	-0.256168	-0.824126	-2.082590	1.013284	1.864866	1.870544	-0.074819	-0.066867	0.105952	-0.283271	-0.538262	0.004776
2018-11-18 16:08:42+07:00	-0.187597	-0.918889	-2.025969	1.028244	1.864853	1.870546	-0.031813	0.064785	0.003092	-0.423955	-0.899937	0.004255
2018-11-18 16:09:04+07:00	-0.240089	-0.894082	-1.927287	1.028244	1.864853	1.870546	0.036164	-0.061080	0.046610	-0.330166	-1.127276	0.004792

ln [10], [11] ทำการ Normalization ด้วย standardized Norm

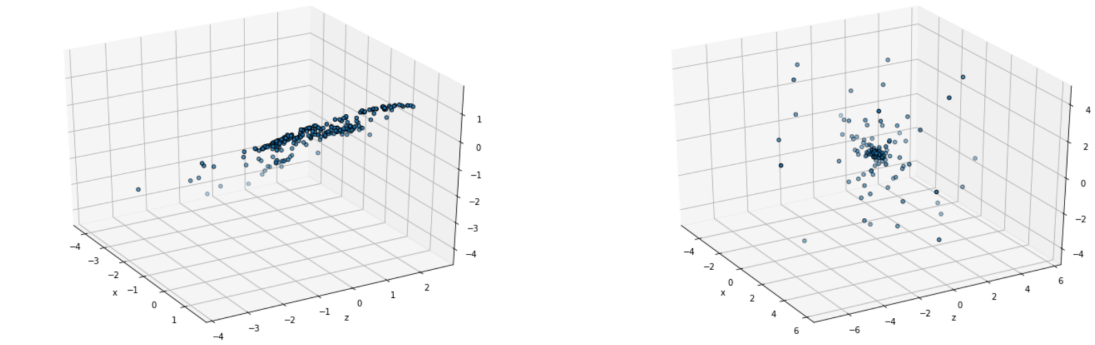
3. แสดงข้อมูลในรูปกราฟ

```
In [12]: df.reset_index().plot(x='uts', subplots=True, figsize=(24, 36))
```



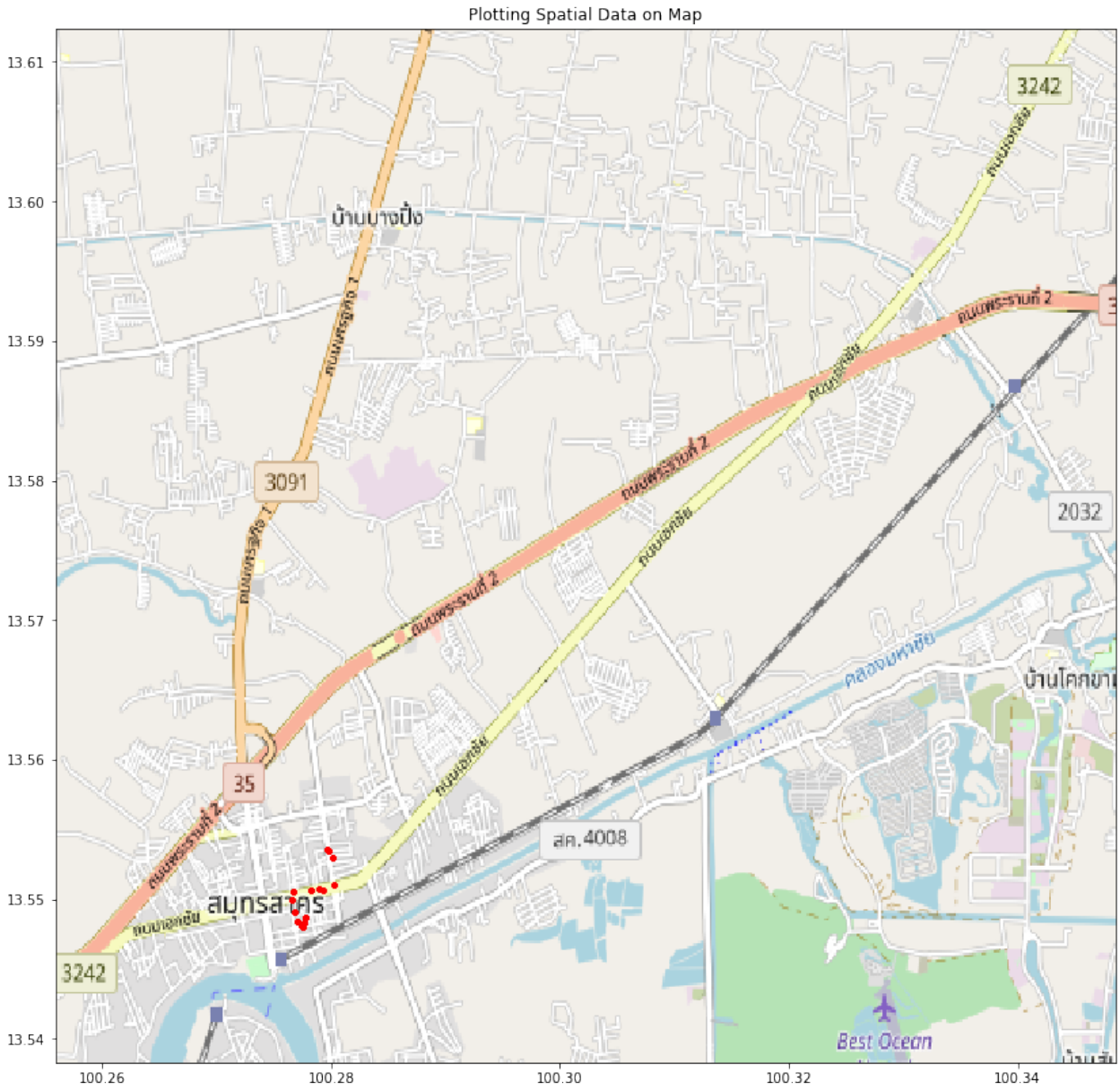
ln [12] แก้ index กลับเป็นแบบเดิม แล้ว plot graph โดยแกน x เป็นแกนของ uts และ แกน y เป็น feature ที่เหลือ

```
In [13]: fig = plt.figure(figsize=(24, 8))
ax = fig.add_subplot(1, 2, 1, projection='3d')

ax.scatter(df['accelerateX'],df['accelerateY'],df['accelerateX'],s=20,edgecolor='k')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_ylabel('z')
ax.view_init(30, -30)

ax = fig.add_subplot(1, 2, 2, projection='3d')

ax.scatter(df['gyro.x'],df['gyro.y'],df['gyro.z'],s=20,edgecolor='k')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_ylabel('z')
ax.view_init(30, -30)
```



ln [13] แสดงกราฟข้อมูลความสัมพันธ์ระหว่างคู่ features ด้วย 2D Scatter Pair Plot หรือ 2D sns.jointplot หรือ 3D Scatter Plot เพื่อดูความสัมพันธ์ของข้อมูลเชิง 3 มิติ (accelerateX, accelerateY, accelerateZ) หรือ (gyro.x, gyro.y, gyro.z)

```
In [14]: map_im = plt.imread('map.png')
         fig, ax = plt.subplots(figsize=(14,14))
         BBox = [100.2559,100.3486,13.5383,13.6124]
         ax.scatter(df_copy['gps.y'], df_copy['gps.x'], zorder=1, alpha=0.5, c='r', s=10)
         ax.set_title('Plotting Spatial Data on Map')
         ax.set_xlim(BBox[0], BBox[1])
         ax.set_ylim(BBox[2], BBox[3])
         ax.imshow(map_im, zorder=0, extent=BBox, aspect='auto')
```



ln [14] แสดงข้อมูลเชิงพิกัด Geolocation ของ ข้อมูล (gps.x, gps.y)

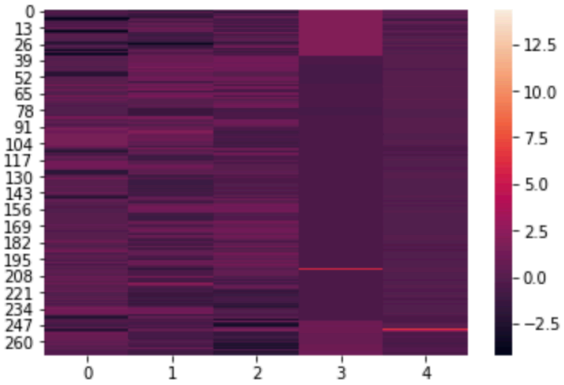
4. ขั้นตอนการจัดเตรียมข้อมูลเพื่อนำเข้าโมเดล

```
In [16]: columns = ['accelerateX', 'accelerateY', 'accelerateZ', 'compass', 'heartrate']
         arr = df[columns].to_numpy()
         arr.shape
```

```
Out[16]: (271, 5)
```

```
In [17]: sns.heatmap(arr)
```

```
Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x136debc50>
```



ln [16] สร้าง table ที่มีข้อมูล 5 Features

[accelerateX, accelerateY, accelerateZ, compass, heartrate]

ln [17] นำ array ที่สร้างจาก ln [16] มา plot เป็น heatmap

```
In [18]: timestep = 3
         stride = 1
         data = []
         for i in range(0, len(df)-timestep+1, stride):
             data.append(df[columns].iloc[i: i+timestep].to_numpy())
```

```
In [19]: data = np.array(data)
         data.shape
```

```
Out[19]: (269, 3, 5)
```

ln [18] จัดเรียงข้อมูล time series โดยต้องการตัด ข้อมูลตาม time series เงื่อนไข time step ที่ 3 และ time stride ที่ 1

ln [19] ทำข้อมูลใน ln [18] เป็น array 3 มิติ ที่มี 5 feature โดยใช้ array จาก ln [16]

```
In [20]: data
```

```
Out[20]: array([[[-1.11828156, -2.03670093, -0.35376317,  2.09692929,
                   0.51393742],
                  [-0.34271073,  0.67224354, -0.4486707 ,  2.09692929,
                   -0.0487981 ],
                  [ 0.04223724,  0.00691484, -1.85718468,  2.06193985,
                    0.31072737]],

                 [[-0.34271073,  0.67224354, -0.4486707 ,  2.09692929,
                   -0.0487981 ],
                  [ 0.04223724,  0.00691484, -1.85718468,  2.06193985,
                    0.31072737],
                  [-0.3876371 , -1.19474507, -0.18713573,  2.08148088,
                    0.59209513]],

                 [[ 0.04223724,  0.00691484, -1.85718468,  2.06193985,
                    0.31072737],
                  [-0.3876371 , -1.19474507, -0.18713573,  2.08148088,
                    0.59209513],
                  [-0.75035831, -1.41205597, -0.47779004,  2.08148088,
                   -0.08787695]],

                 ...,

                 [[ 0.61540298,  0.92279609, -2.73831488,  0.96679796,
                   -0.28327123],
                  [ 0.01717307, -0.45995626, -0.40391315,  0.99297999,
                   -0.22074506],
                  [-0.25616838, -0.82412577, -2.08259004,  1.01328405,
                   -0.28327123]],

                 [[ 0.01717307, -0.45995626, -0.40391315,  0.99297999,
                   -0.22074506],
                  [-0.25616838, -0.82412577, -2.08259004,  1.01328405,
                   -0.28327123],
                  [-0.18759656, -0.91888925, -2.02596911,  1.02824384,
                   -0.42395511]],
```

ln [20] แสดงผลลัพธ์จาก ln [19]

60010113 คณิศร พิทักษ์วงศ์, 60010479 ธีระสาร มินทะขัด

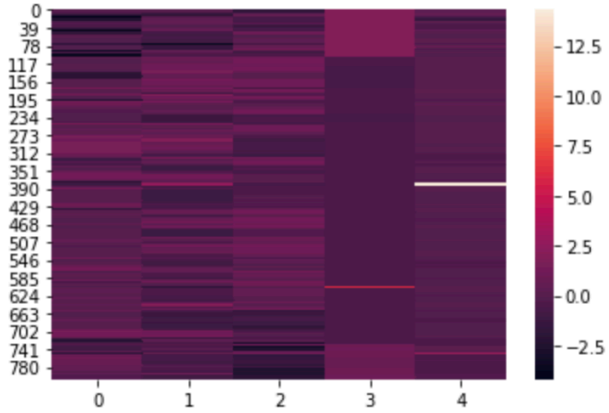
```
In [21]: data = np.concatenate(data)
```

```
In [22]: data
```

```
Out[22]: array([[-1.11828156, -2.03670093, -0.35376317,  2.09692929,  0.51393742],
                [-0.34271073,  0.67224354, -0.4486707 ,  2.09692929, -0.0487981 ],
                [ 0.04223724,  0.00691484, -1.85718468,  2.06193985,  0.31072737],
                ...,
                [-0.25616838, -0.82412577, -2.08259004,  1.01328405, -0.28327123],
                [-0.18759656, -0.91888925, -2.02596911,  1.02824384, -0.42395511],
                [-0.24008947, -0.89408187, -1.92728678,  1.02824384, -0.33016586]])
```

```
In [23]: sns.heatmap(data)
```

```
Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x12d62af10>
```



ln [21] ปรับ array จาก ln [19] ให้เป็น 2 มิติ

ln [23] นำ array ที่ปรับแล้วจาก ln [21] มา plot เป็น heatmap

กลุ่ม เด็กดีขี้เมา

60010113 คณิศร พิทักษ์วงศ์, 60010479 ธีระสาร มินทะขัด

ตอนที่ 2: การทดลองการลดมิติของข้อมูลด้วยค่า Principle Component Analysis

1. เตรียมชุดข้อมูล feature3 ค่า $\mathcal{X}$(accelerateX,accelerateY,accelerateZ)

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('watch_test2_sample.csv')
df['uts'] = pd.to_datetime(df['uts'])
```

In [3]:
```
df.drop_duplicates(inplace=True)
```

In [4]:
```
df.set_index('uts', inplace=True)
df.fillna(df.mean(), inplace=True)
```

In [5]:
```
accelerator_df = df[['accelerateX', 'accelerateY', 'accelerateZ']].copy()
accelerator_df
```

In [1] ทำการ import library ที่จำเป็นต่อการทดลอง

In [2] อ่านไฟล์จาก .csv โดยใช้ pandas แล้วเก็บลง df แล้วแปลง format ของ feature “uts” จาก string เป็น datetime และทำการ copy ข้อมูลไว้เพื่อใช้งานในการplot GPS จากค่าจริง

In [3] ทำการลบข้อมูลที่มีค่าซ้ำซ้อน

In [4] กำหนด index เป็น feature ของ uts เพราะ ข้อมูลถูกเก็บแบบ timestamp แล้วแทนข้อมูลที่หายไป หรือ NaN ด้วย mean ของแต่ละ feature

In [5] จัด array ให้มี feature 3 คือ [accelerateX, accelerateY, accelerateZ]

Out[5]:

uts	accelerateX	accelerateY	accelerateZ
2018-11-18 08:18:41+07:00	-3.957379	-14.204506	2.303692
2018-11-18 08:19:03+07:00	-0.038236	-1.156625	1.883101
2018-11-18 08:19:45+07:00	1.906998	-4.394027	-4.358852
2018-11-18 08:20:13+07:00	-0.265259	-10.149148	3.042116
2018-11-18 08:20:33+07:00	-2.098175	-11.195846	1.754056
...	...	...	...
2018-11-18 16:07:32+07:00	4.803340	0.050184	-8.263658
2018-11-18 16:07:58+07:00	1.780343	-6.609970	2.081448
2018-11-18 16:08:19+07:00	0.399084	-8.364026	-5.357756
2018-11-18 16:08:42+07:00	0.745593	-8.820463	-5.106835
2018-11-18 16:09:04+07:00	0.480334	-8.700976	-4.669516

271 rows × 3 columns

2. **ปรับให้เป็น Zero Mean และ คำนวณค่า covariance**

```
In [6]: accelerator_df = accelerator_df - accelerator_df.mean()
```

```
In [7]: accelerator_df
```

Out[7]:

uts	accelerateX	accelerateY	accelerateZ
2018-11-18 08:18:41+07:00	-5.618468	-9.810479e+00	-1.596343
2018-11-18 08:19:03+07:00	-1.699325	3.237402e+00	-2.016934
2018-11-18 08:19:45+07:00	0.245909	1.776357e-15	-8.258887
2018-11-18 08:20:13+07:00	-1.926348	-5.755121e+00	-0.857919
2018-11-18 08:20:33+07:00	-3.759265	-6.801819e+00	-2.145979
...	...	...	...
2018-11-18 16:07:32+07:00	3.142251	4.444211e+00	-12.163693
2018-11-18 16:07:58+07:00	0.119253	-2.215943e+00	-1.818587
2018-11-18 16:08:19+07:00	-1.262006	-3.969999e+00	-9.257791
2018-11-18 16:08:42+07:00	-0.915496	-4.426436e+00	-9.006870
2018-11-18 16:09:04+07:00	-1.180755	-4.306949e+00	-8.569551

271 rows × 3 columns

```
In [8]: accelerator = accelerator_df.to_numpy()
```

```
In [9]: cov = accelerator.T.dot(accelerator)/(len(accelerator) - 1)
        cov
```

```
Out[9]: array([[25.4476783 ,  3.15359882, -1.82430517],
               [ 3.15359882, 23.14423757,  7.02704944],
               [-1.82430517,  7.02704944, 19.5368561 ]])
```

```
In [10]: eigen_values, eigen_vectors = np.linalg.eig(cov)
         eigen_values, eigen_vectors
```

```
Out[10]: (array([13.15046863, 25.75818909, 29.22011425]),
          array([[ 0.26701286, -0.87106272,  0.41225462],
                 [-0.60926567,  0.17885949,  0.77253131],
                 [ 0.74665888,  0.45744838,  0.48295082]]))
```

```
In [11]: sorted_indexes = np.argsort(eigen_values)[::-1]
         eigen_values = eigen_values[sorted_indexes]
         eigen_vectors = eigen_vectors[:, sorted_indexes]
         eigen_values, eigen_vectors
```

```
Out[11]: (array([29.22011425, 25.75818909, 13.15046863]),
          array([[ 0.41225462, -0.87106272,  0.26701286],
                 [ 0.77253131,  0.17885949, -0.60926567],
                 [ 0.48295082,  0.45744838,  0.74665888]]))
```

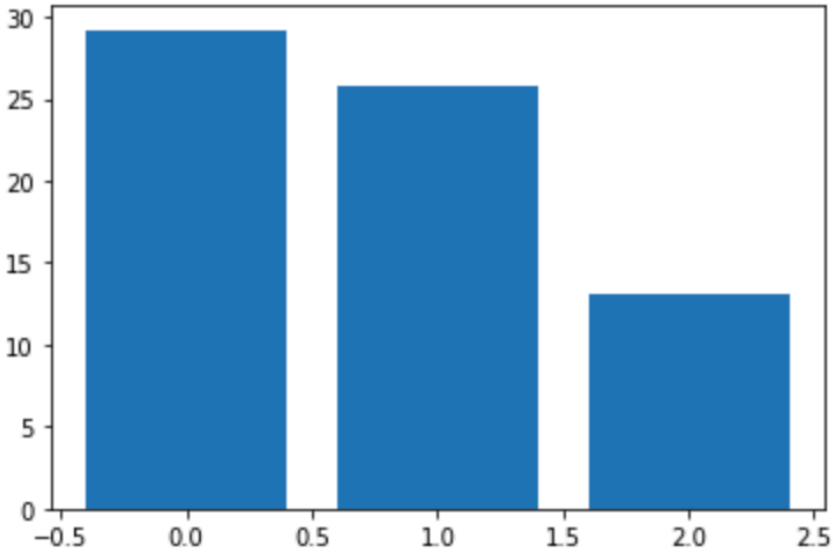
ln [9] คำนวณค่า covariance matrix ของชุดข้อมูล

ln [10] คำนวณค่า eigenvalue / eigenvector จาก covariance matrix ที่คำนวณได้จาก ln [9]

ln [11] sort array ที่ได้จาก ln [10] เพื่อเตรียม plot graph ในขั้นต่อไป

3. แสดงกราฟ Eigen Space (Eigenvalue, Eigenvector)

```
In [12]: plt.bar(np.arange(len(eigen_values)), eigen_values)
Out[12]: <BarContainer object of 3 artists>
```



In [12] แสดงกราฟแท่ง (Bar graph) ของค่า Eigenvalue ที่จัดเรียงค่าจากมากไปน้อย

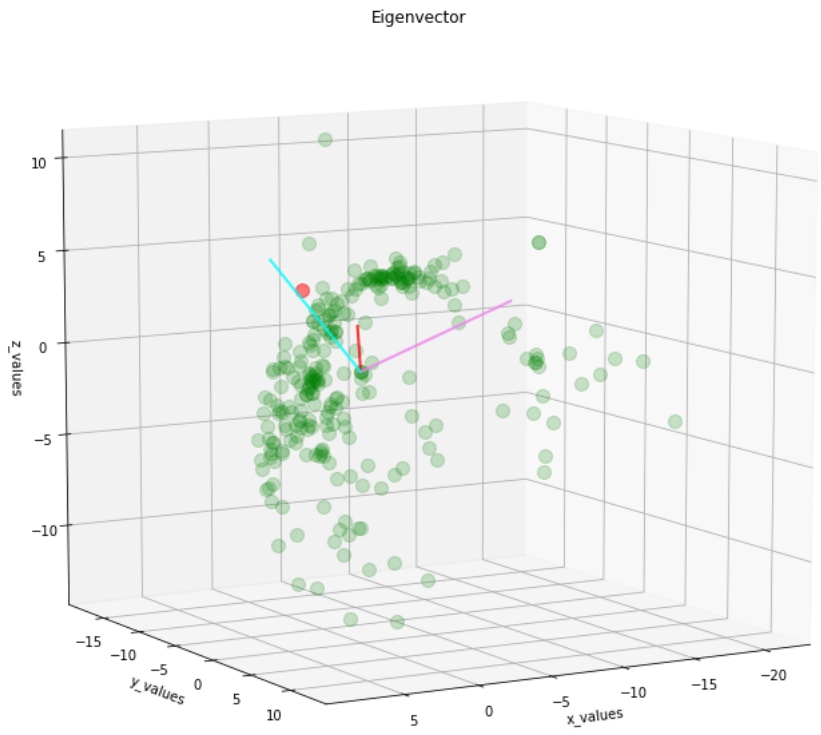
```
In [13]: np.sqrt(eigen_values)
Out[13]: array([5.40556327, 5.07525261, 3.62635749])

In [14]: eigen_vectors
Out[14]: array([[ 0.41225462, -0.87106272,  0.26701286],
                [ 0.77253131,  0.17885949, -0.60926567],
                [ 0.48295082,  0.45744838,  0.74665888]])

In [15]: eigen_vectors = (eigen_vectors.T * np.sqrt(eigen_values)).T
```

In [15] ปรับขนาดของ Eigenvector ด้วยค่า Eigenvalue จากสูตร

$$\text{Eigenvector} * \sqrt{Eigenvalue}$$

In [16]:
```
scale = 2
ev1, ev2, ev3 = eigen_vectors.T * scale
```

In [17]:
```
fig = plt.figure(figsize=(60, 12))

ax = fig.add_subplot(141, projection='3d')

ax.plot(accelerator[:, 0],
        accelerator[:, 1],
        accelerator[:, 2],
        'o',
        markersize=10,
        color='green',
        alpha=0.2
       )

ax.plot([df['accelerateX'].mean()],
        [df['accelerateY'].mean()],
        [df['accelerateZ'].mean()],
        'o',
        markersize=10,
        color='red',
        alpha=0.5
       )

ax.plot([0, ev1[0]], [0, ev1[1]], [0, ev1[2]],
        color='red', alpha=0.8, lw=2
       )
ax.plot([0, ev2[0]], [0, ev2[1]], [0, ev2[2]],
        color='violet', alpha=0.8, lw=2
       )
ax.plot([0, ev3[0]], [0, ev3[1]], [0, ev3[2]],
        color='cyan', alpha=0.8, lw=2
       )
ax.set_xlabel('x_values')
ax.set_ylabel('y_values')
ax.set_zlabel('z_values')

plt.title('Eigenvector')

ax.view_init(10, 60)
```



ln [16] เตรียมค่าเพื่อนำในสร้าง graph ตามสูตรด้านล่าง

```
# Split each eigenvector and scale with its sqrt(eigenvalue)
ev1 = eig_vecs[:,0]*np.sqrt(eig_vals[0])
ev2 = eig_vecs[:,1]*np.sqrt(eig_vals[1])
ev3 = eig_vecs[:,2]*np.sqrt(eig_vals[2])
```

ln [17] แสดงกราฟความสัมพันธ์ของ feature และ eigen vector จาก ln [16]

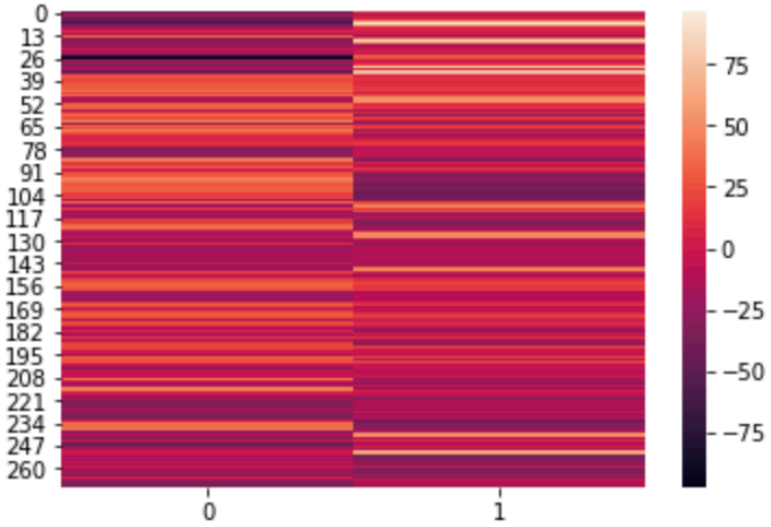
```
In [18]: K = 2

         pca = accelerator.dot( eigen_vectors[:, :K])

In [19]: pca
```

```
Out[19]: array([[-5.37811822e+01,  1.49013734e+01],
                [ 5.37392288e+00,  7.59435471e+00],
                [-1.39162216e+01, -1.48583130e+01],
                [-2.83599554e+01,  2.42294504e+00],
                [-3.88042832e+01,  7.96651355e+00],
                [-5.21090850e+01,  4.66019934e+01],
                [ 1.83454424e+01, -1.31337096e+01],
                [-5.67352295e+01,  9.60914170e+01],
                [-2.55461835e+01,  5.46388549e+01],
                [-2.25228300e+01,  5.60560563e+00],
                [ 1.22171468e+00,  1.06332249e+01],
                [-3.69782231e+01,  2.61911370e+01],
                [ 1.07145241e+01,  1.06019548e+00],
                [-1.68749714e+01,  1.98803977e+00],
                [ 3.51453509e+01, -1.84538192e+01],
                [-1.20574507e+01,  3.78674641e+01],
                [-3.75640957e+01,  9.23175298e+01],
                [-9.81386740e+00,  7.20369008e+01],
                [-2.40529866e+01,  4.55844588e+01],
```

ln [18] ลดมิติของข้อมูลจาก 3D features $\mathcal{X}$ (accelerateX, accelerateY, accelerateZ) ลงเหลือ 2D โดย เลือก eigenvector 2 vector แรก ที่สัมพันธ์กับ eigenvalue ที่มีค่าสูงสุด 2 อันดับแรก [np.dot()]

```
In [20]: sns.heatmap(pca)
```

```
Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x12af3add0>
```



ln [20] แสดง graph heatmap จากข้อมูลที่ได้ใน ln [18]