

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

60010113 คณิศร พิทักษ์วงศ์, 60010479 ชีระสาร มินทะขันธ์

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 x 13 columns

In [4] นำข้อมูลมาแสดงผลเพื่อตรวจสอบความถูกต้องของ feature ใน table

2. ทำการลบปัญหาข้อผิดพลาดของข้อมูล

```
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
```

In [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
```

In [6] จัดการข้อมูลที่เป็น NaN โดยใช้ค่า median ของแต่ละ feature

60010113 คณิศร พิทักษ์วงศ์, 60010479 ชีระสาร มินทะขันธ์

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

Out[8]:

	accelerateX	accelerateY	accelerateZ	compass	gps.x	gps.y	gyro.x	gyro.y	gyro.z	heartrate	light	pressure
uts												
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 16:08:42+07:00	4.803340	0.050184	-8.263658	225.21982	13.551022	100.280217	-0.031927	-0.094716	0.037248	66.0	0	1008.64700

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

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

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	10.0	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	11.0	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 x 12 columns

In [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
```

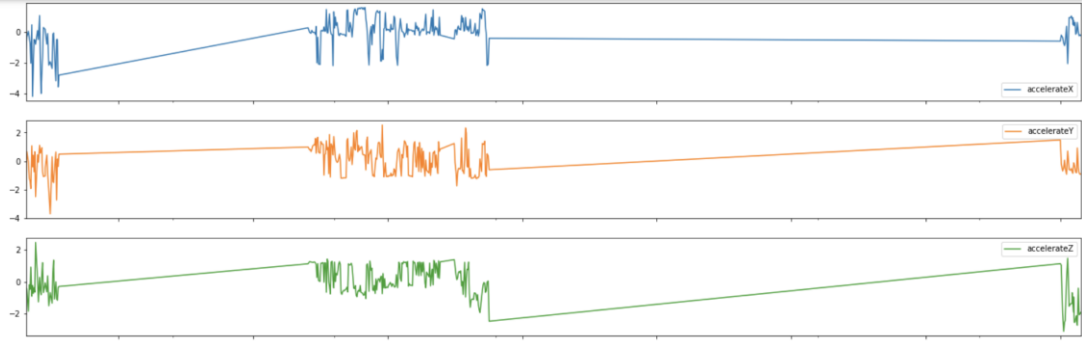
```
Out[11]:
```

	accelerateX	accelerateY	accelerateZ	compass	gps.x	gps.y	gyro.x	gyro.y	gyro.z	heartrate	light	pressure
uts												
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

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

3. แสดงข้อมูลในรูปภาพ

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



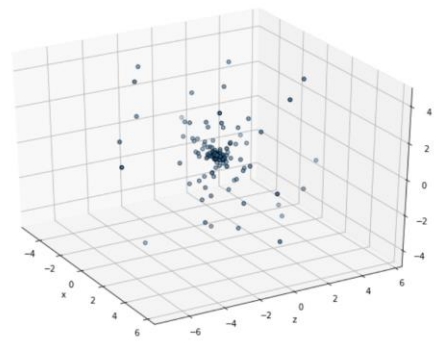
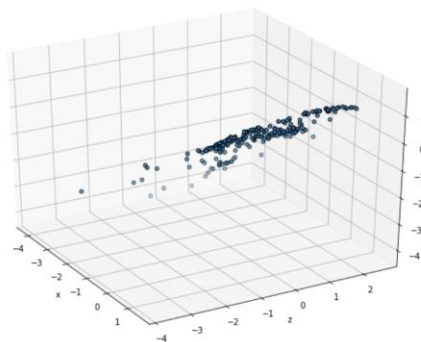
In [12] แก่ index กลับเป็นแบบเดิม แล้ว plot graph โดยแกน x เป็นแกนของ uts และ แกน y เป็น feature ที่เหลือ

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

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

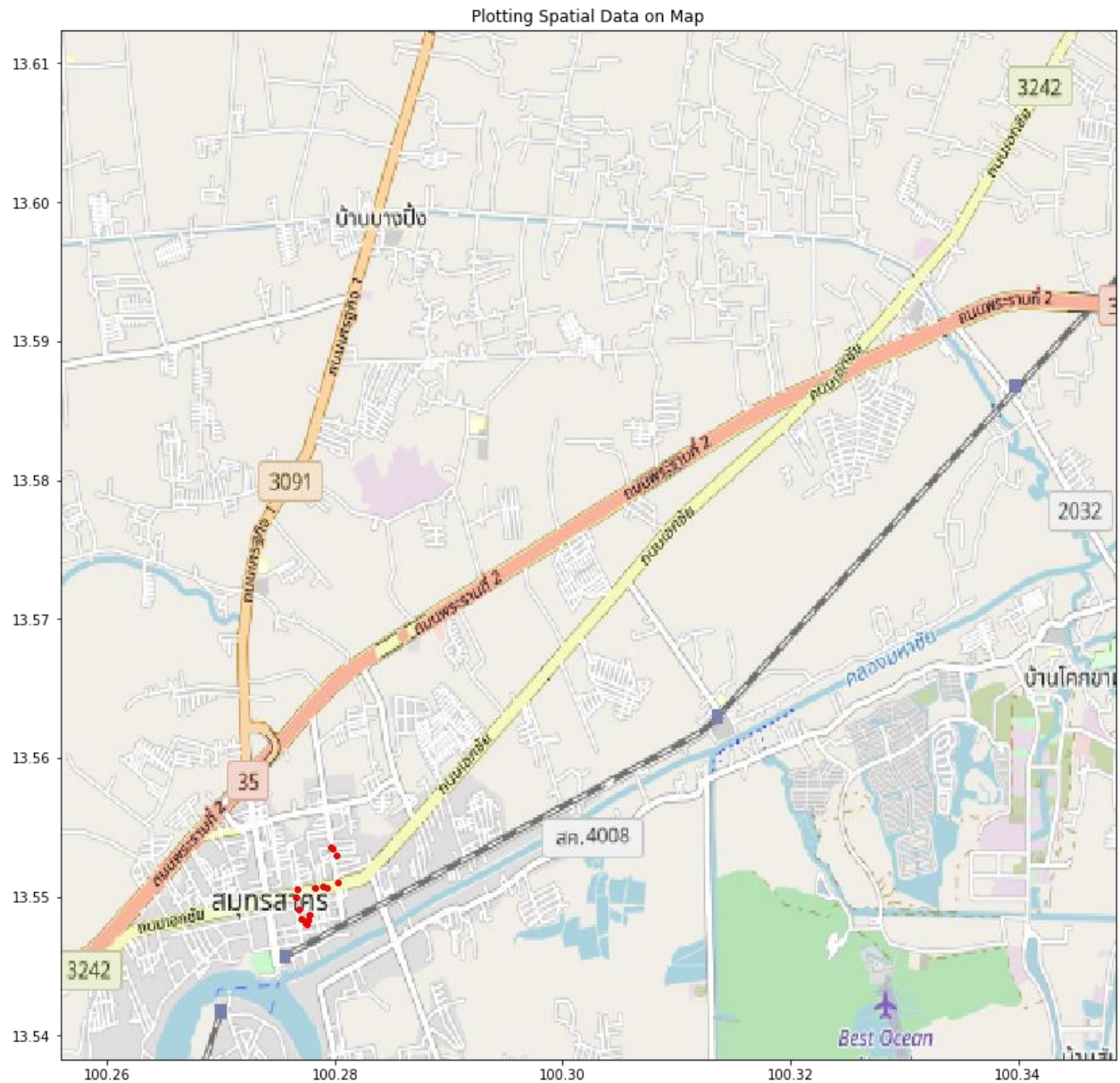
ax = fig.add_subplot(1, 2, 2, projection='3d')

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_zlabel('z')
ax.view_init(30, -30)
```



In [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.x'], df_copy['gps.y'], 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')
```



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

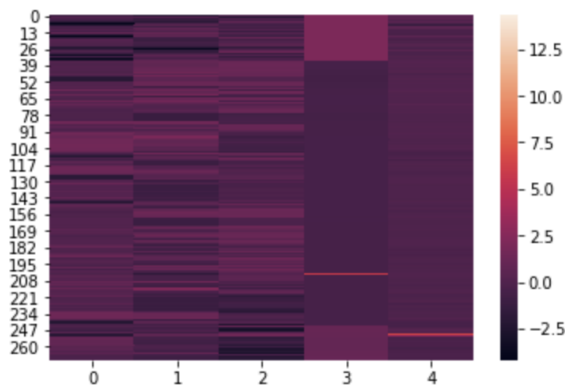
4. ขั้นตอนการจัดเตรียมข้อมูลเพื่อนำเข้าโมเดล

```
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>



In [16] สร้าง table ที่มีข้อมูล 5 Features

[accelerateX, accelerateY, accelerateZ, compass, heartrate]

In [17] นำ array ที่สร้างจาก In [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)

In [18] จัดเรียงข้อมูล time series โดยต้องการตัด ข้อมูลตาม time series เกลื่อนไข time step ที่ 3 และ time stride ที่ 1

In [19] ทำข้อมูลใน In [18] เป็น array 3 มิติ ที่มี 5 feature โดยใช้ array จาก In [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]],
```

In [20] แสดงผลลัพธ์จาก In [19]

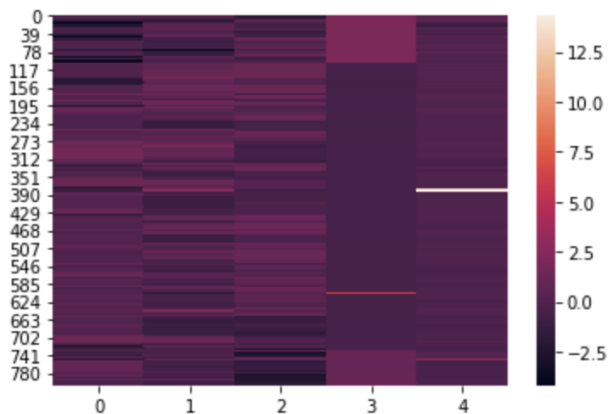
```
In [21]: data = np.concatenate(data)
```

```
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>
```



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

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