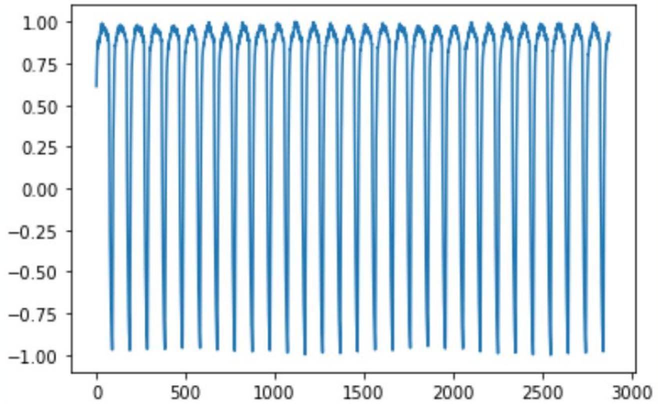
CCBDA-HW3-Anomaly Detection (Autoencoder)

Host: TA Wei-Lun Tseng (曾偉倫)
Email: erictseng.eed06g@nctu.edu.tw / E3 email system

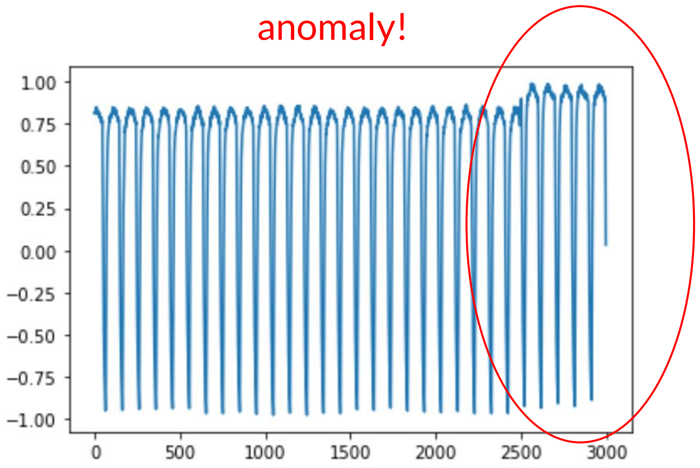
- Kaggle deadline: 05/18/2022 11:59 PM (GMT+8)
- E3 submission deadline: 05/19/2022 08:59 AM (GMT+8)

Time-series Anomaly Detection

normal



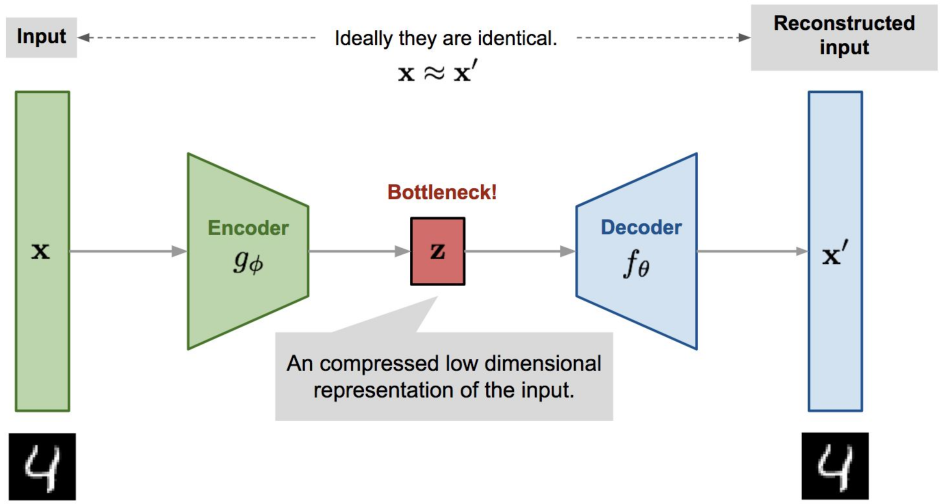
anomaly!



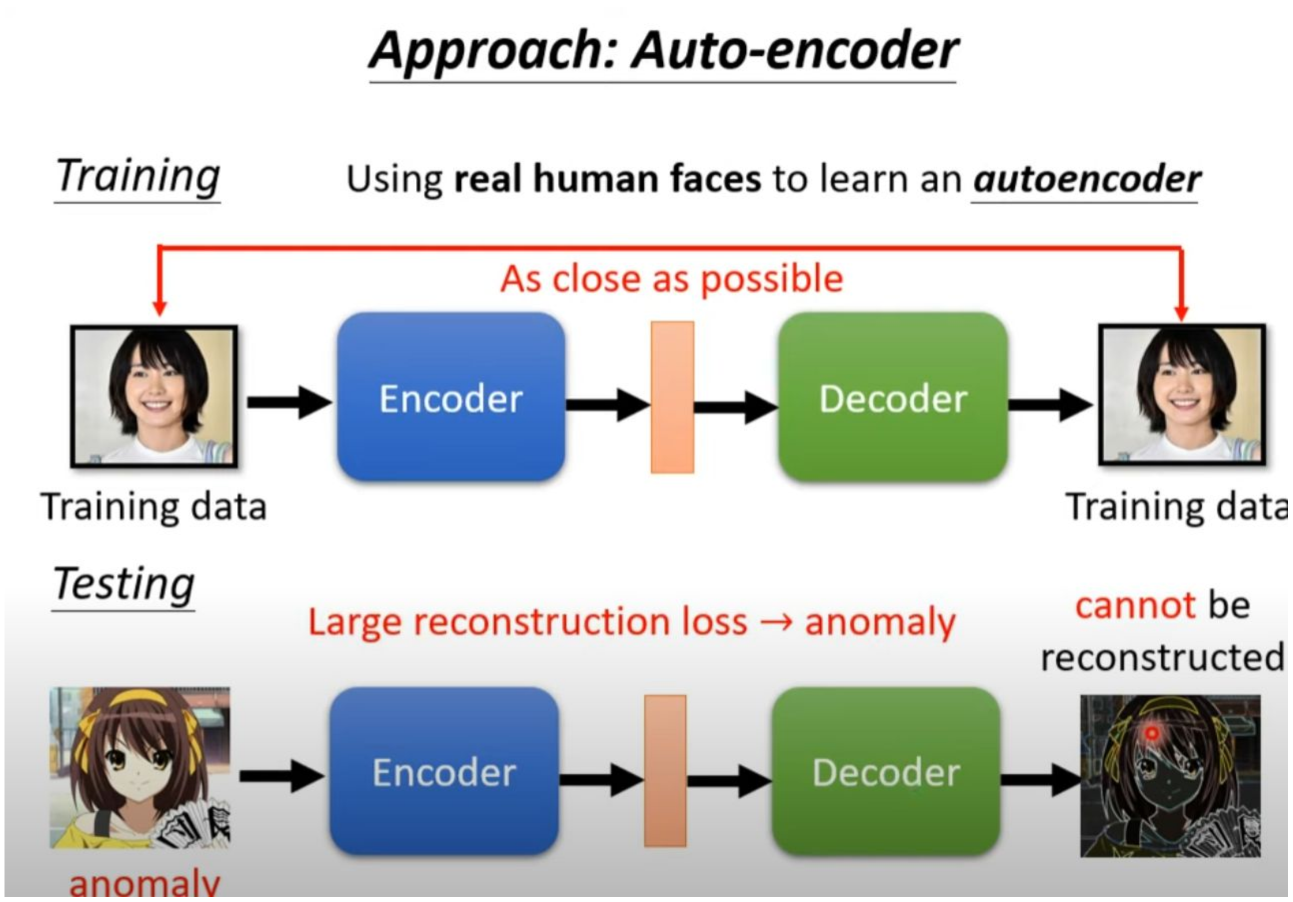
Autoencoder

● input	x
● output	x’

- reconstruction error = abs(x’-x)


Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
$\mathbf{x}$
Encoder
g_ϕ
Bottleneck!
$\mathbf{z}$
Decoder
f_θ
$\mathbf{x}'$
An compressed low dimensional representation of the input.

Autoencoder



reference:

[(7)【機器學習2021】自編碼器 (Auto-encoder) (下) – 領結變聲器與更多應用 - YouTube]

Dataset

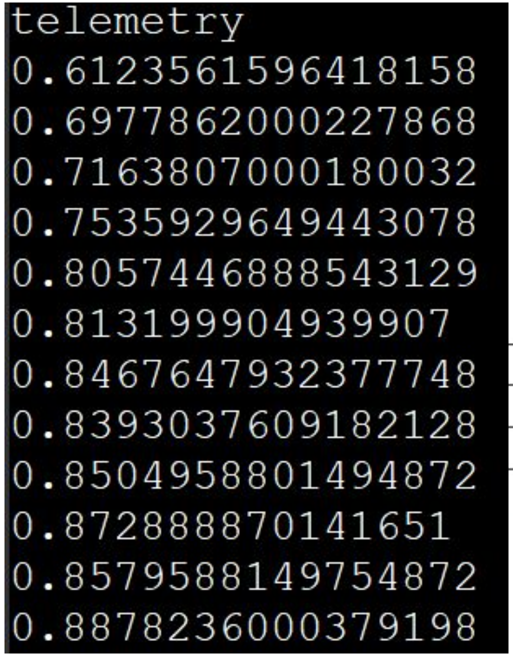
5 sensor data from airplane (single channle)

objective: detect anomaly sequence for each sensor

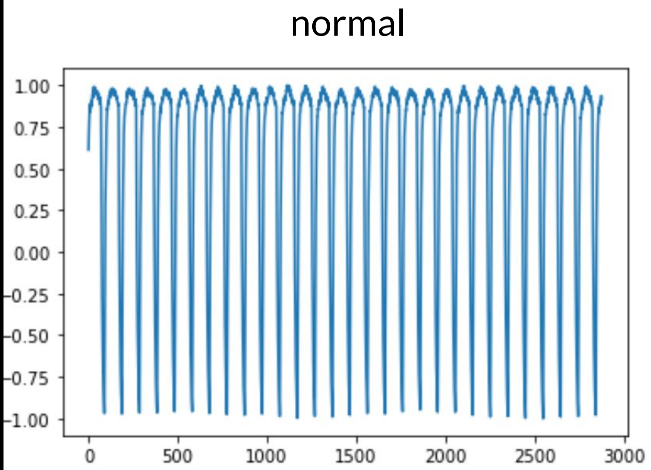
- given:
  - Normal data (each sensor has one normal data)
    - sensor_A_normal.csv / sensor_B_normal.csv / sensor_C_normal.csv / sensor_D_normal.csv / sensor_E_normal.csv (.csv format)
  - Public test data (with label) (normal: 0 / anomaly: 1)
    - sensor_A_public.csv / sensor_B_public.csv / sensor_C_public.csv / sensor_D_public.csv / sensor_E_public.csv (.csv format)
  - Public test data (no label)
    - sensor_A_private.csv / sensor_B_private.csv / sensor_C_private..csv / sensor_D_private..csv / sensor_E_private.csv (.csv format)

Dataset - Normal Data

- sensor_A_normal.csv

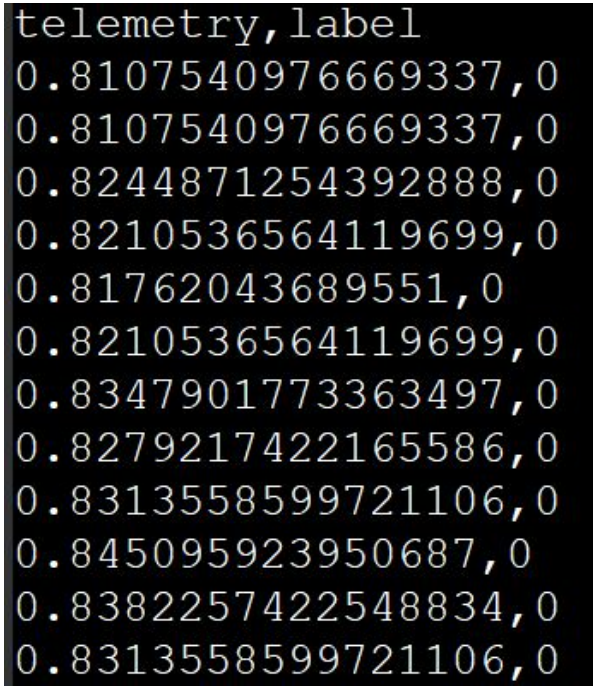


```
telemetry
0.6123561596418158
0.6977862000227868
0.7163807000180032
0.7535929649443078
0.8057446888543129
0.813199904939907
0.8467647932377748
0.8393037609182128
0.8504958801494872
0.872888870141651
0.8579588149754872
0.8878236000379198
```



Dataset - Public Test Data

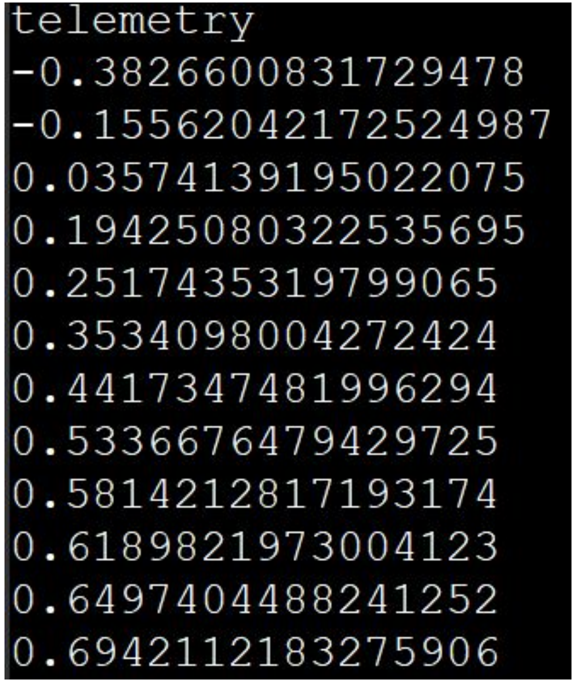
- sensor_A_public.csv



```
telemetry,label
0.8107540976669337,0
0.8107540976669337,0
0.8244871254392888,0
0.8210536564119699,0
0.817620436895 51,0
0.8210536564119699,0
0.8347901773363497,0
0.8279217422165586,0
0.8313558599721106,0
0.8450959239506 87,0
0.8382257422548834,0
0.8313558599721106,0
```

Dataset - Private Test Data

- sensor_A_private.csv



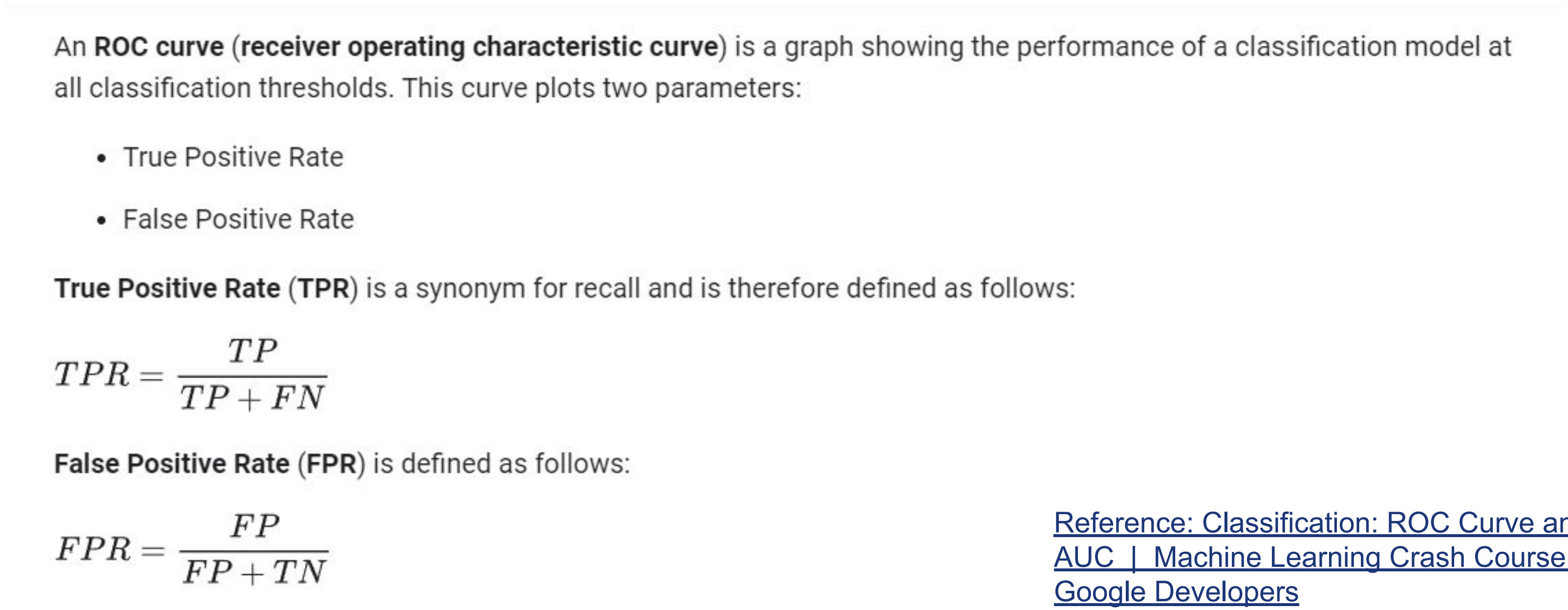
```
telemetry
-0.3826600831729478
-0.15562042172524987
0.03574139195022075
0.1942508032253569 5
0.251743531979906 5
0.353409800427242 4
0.441734748199629 4
0.533667647942972 5
0.581421281719317 4
0.618982197300412 3
0.649740448824125 2
0.694211218327590 6
```

Problem Description

Time-series anomaly detection

- Use reconstruction error value from autoencoder as anomaly score
- In this homework, you need to train 5 autoencoder models to detect each sensor data.
- Given:
    - Normal time-series sensor data
- Objective:
    - Training an autoencoder with normal data
    - Detect anomaly base on anomaly score (reconstruction error)

Evaluation Metric



An **ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

True Positive Rate (**TPR**) is a synonym for recall and is therefore defined as follows:

$$TPR = \frac{TP}{TP + FN}$$

False Positive Rate (**FPR**) is defined as follows:

$$FPR = \frac{FP}{FP + TN}$$

Reference: Classification: ROC Curve and AUC | Machine Learning Crash Course | Google Developers

Evaluation Metric

An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives. The following figure shows a typical ROC curve.

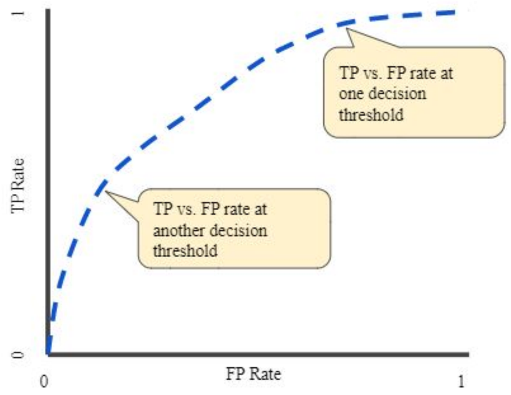


Figure 4. TP vs. FP rate at different classification thresholds.

[Reference: Classification: ROC Curve and AUC | Machine Learning Crash Course | Google Developers]

Evaluation Metric

- ROC AUC Score
  - Compute Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1).

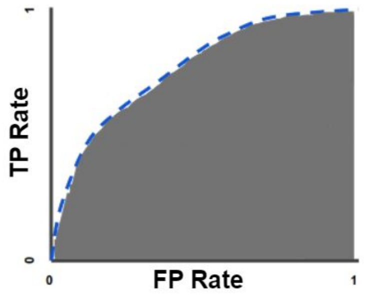
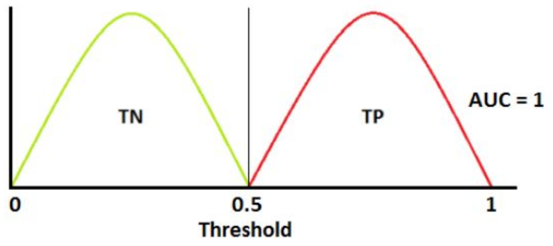


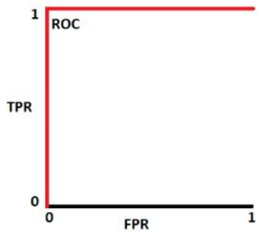
Figure 5. AUC (Area under the ROC Curve).

Reference: Classification: ROC Curve and AUC | Machine Learning Crash Course | Google Developers

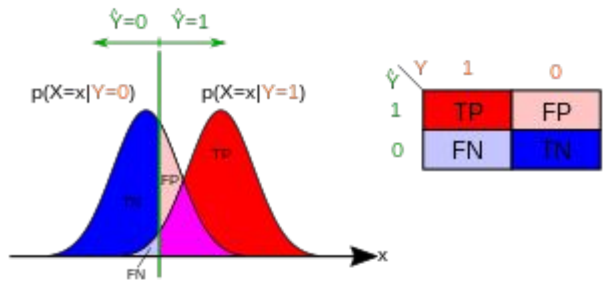
Why ROC AUC score ?

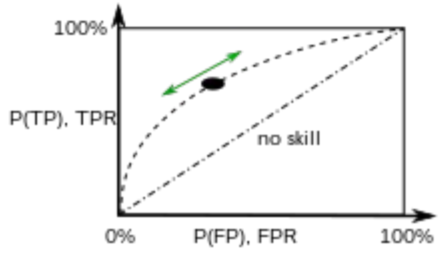
Decision threshold will cause result deference!





[Image 6 and 7] (Image courtesy: My Photoshopped Collection)

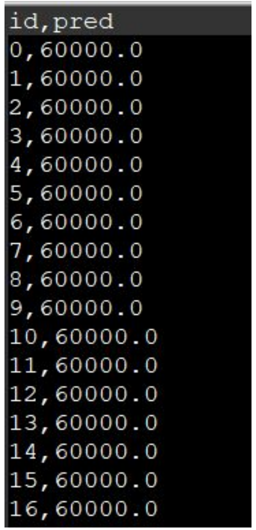




Kaggle Submission

concatenate 10 results (5 public, 5 private) in one csv file.

- sample.csv



```
id,pred
0,60000.0
1,60000.0
2,60000.0
3,60000.0
4,60000.0
5,60000.0
6,60000.0
7,60000.0
8,60000.0
9,60000.0
10,60000.0
11,60000.0
12,60000.0
13,60000.0
14,60000.0
15,60000.0
16,60000.0
```

*** id:

● 0~3999:	public_A	pred score
● 4000~7999:	public_B	pred score
● 8000~11999:	public_C	pred score
● 12000~15999:	public_D	pred score
● 16000~19999:	public_E	pred score
● 20000~23999:	private_A	pred score
● 24000~27999:	private_B	pred score
● 28000~31999:	private_C	pred score
● 32000~35999:	private_D	pred score
● 36000~39999:	private_E	pred score

Grading Policy

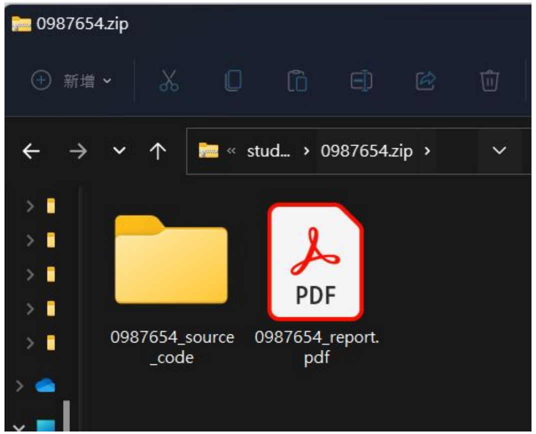
- Total : 100 pts
    - Kaggle Submission: (90 pts)
        - Kaggle Public Score (70 pts)
            - Public Score >= Baseline: 70 pts
        - Kaggle Private Score Ranking: (20 pts)
            - Top 0~25 % : 20 pts
            - Top 25~50 % : 15 pts
            - Top 50~75 % : 10 pts
            - Top 75% ~ : 5 pts
    - E3 Submission (10 pts)
        - report (with template)
        - source code (in one folder)
- Don't cheat!
- Apply yourself!

Link

- Kaggle: https://www.kaggle.com/competitions/ccbda2022spring-hw3
    - Daily submission times limit: 4
- E3
    - Start from 5/5 16:00
    - HW3.zip
        - sensor_A_normal.csv / sensor_B_normal.csv / sensor_C_normal.csv / sensor_D_normal.csv / sensor_E_normal.csv
        - sensor_A_public.csv / sensor_B_public.csv / sensor_C_public.csv / sensor_D_public.csv / sensor_E_public.csv (.csv format)
        - sensor_A_private.csv / sensor_B_private.csv / sensor_C_private..csv / sensor_D_private..csv / sensor_E_private.csv (.csv format)
        - sample_submission.csv
        - report_template.docx
        - CCBDA-HW3-Anomaly Detection (Autoencoder).pdf

Important dates

- Kaggle deadline: 05/18/2022 11:59 PM (GMT+8)
- E3 submission deadline: 05/19/2022 08:59 AM (GMT+8)
  - put source code and report in one zip file (file name: [student ID].zip)
  - ex. 0987654.zip
    - [folder] — 0987654_source_code
    - [pdf] —0987654_report.pdf
- NOT allow any late submissions



Appendix

- Time Series Anomaly Detection using LSTM Autoencoders with PyTorch in Python | Curiousily - Hacker's Guide to Machine Learning