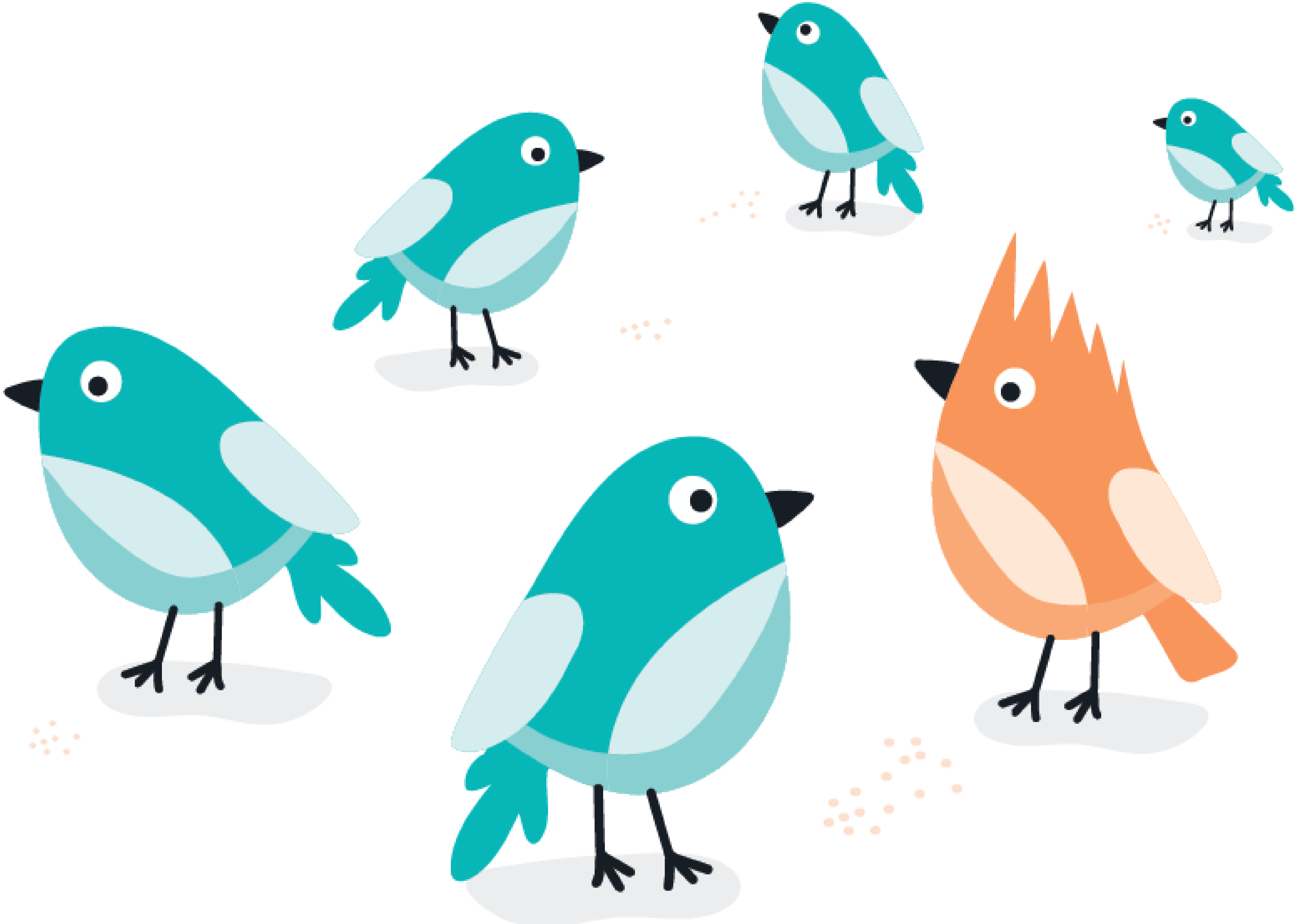
Time Series Anomaly Detection

GAN/Transformer/GRU

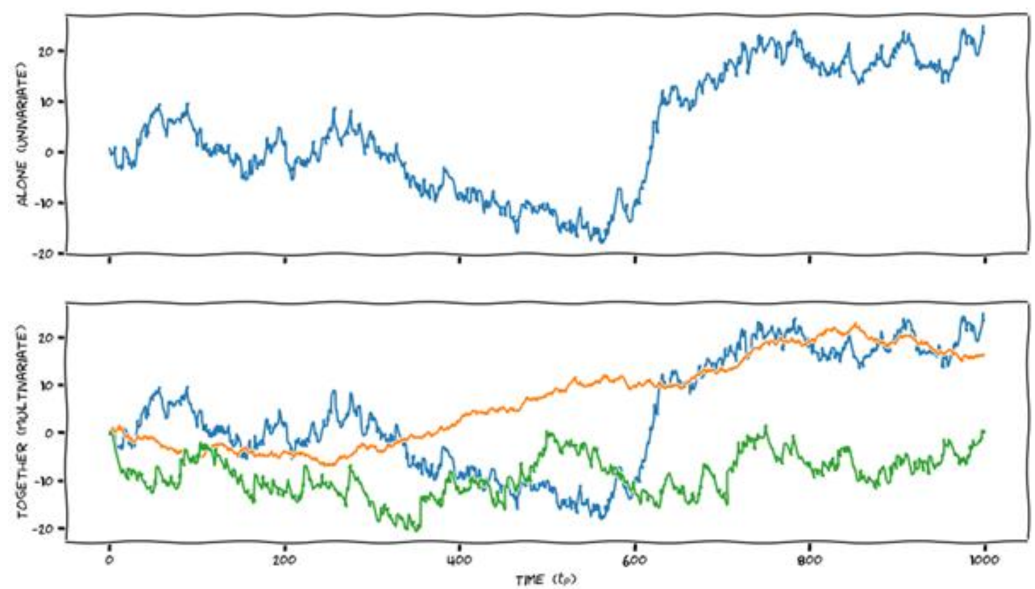
https://zindi.africa/learn/introduction-to-anomaly-detection-using-machine-learning-with-a-case-study



https://ff12.fastforwardlabs.com/

▪ Time series types

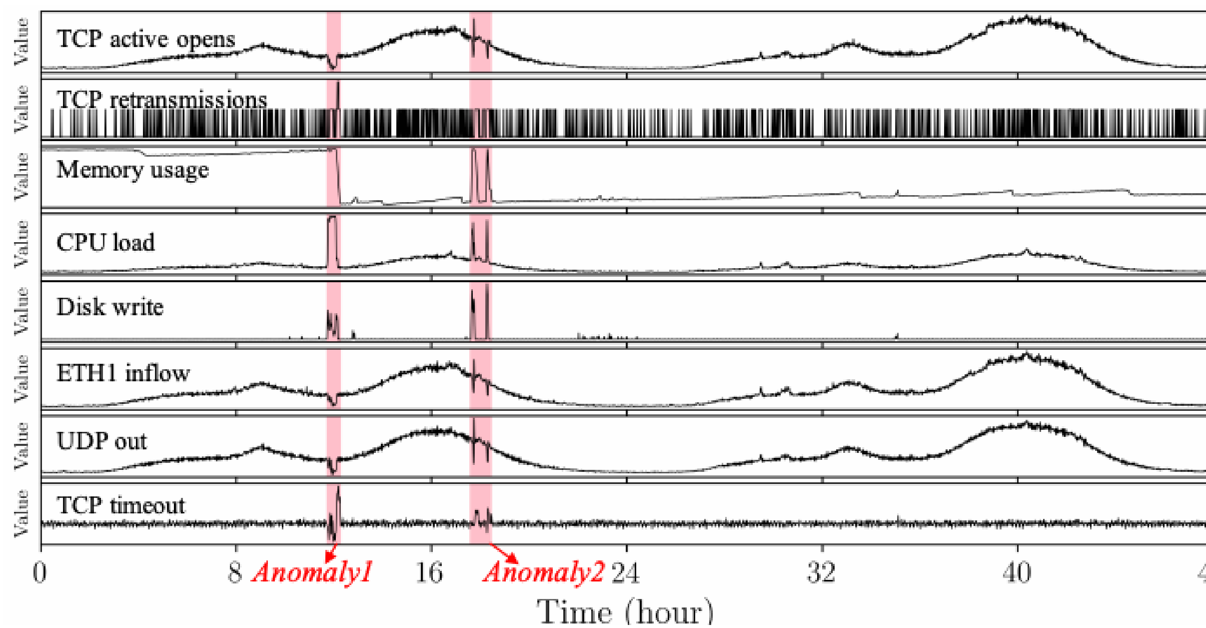
- **Univariate**
  - 단변량 시계열 데이터
  - 주식가격, 유가, 전력 수요
- **Multivariate**
  - 다변량 시계열 데이터
  - 공정 센서 데이터



https://medium.com/geekculture/vector-auto-regression-for-multivariate-time-series-forecasting-9334d29591f3

▪ Multivariate Time series Anomaly Detection

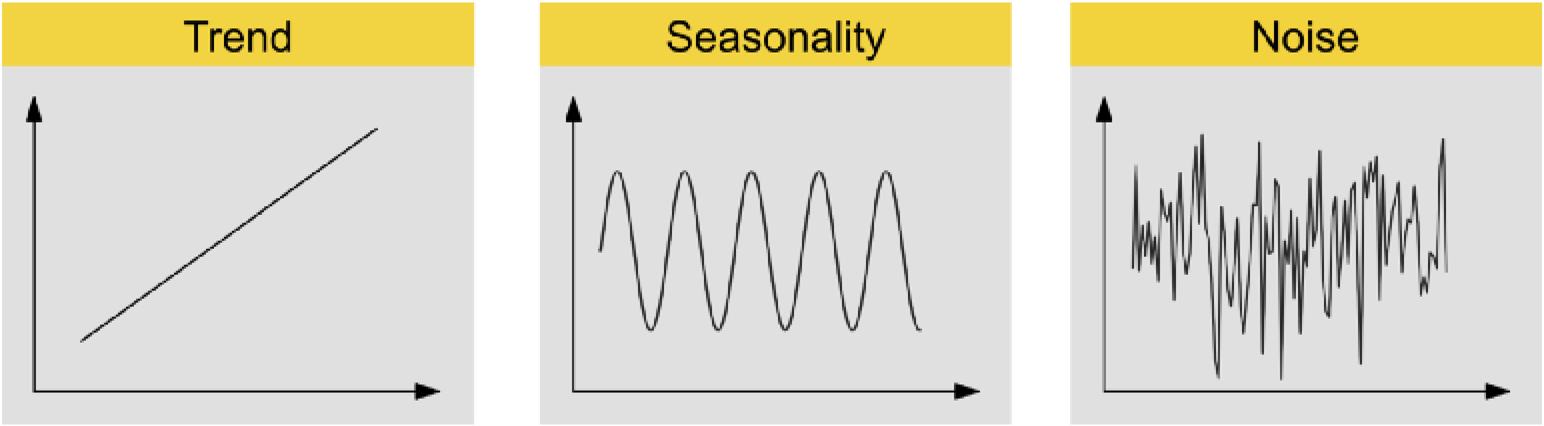
- **Dataset**
  - m개의 변수로 이루어진 t시점 m차원 벡터가 총 시간 T 만큼 존재
- **Task**
  - 길이가 K인 time window $W_t = \{x_{t-k}, \dots, x_{t-1}, x_t\}$를 입력으로 t 시점의 normal/abnormal 여부를 예측



https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbUZYQ0JZSU5kNnc4NGpsTk5qWjB1aGg5V2xTd3xBQ3Jtc0ttQnJhUFpmMGRGNlBHa2czTmJDcTRDdC00aGlYMGg3bjJjZUIwS21kSGMtelJoNmd2NlQ2aVNhcUQ5SkkxSFpBRmJVQUpSMk0tTk1zck9YSUN2TEVLYnhtWW9FalMxam04QV9ENzBJbEJZTWpuOEdvOA&q=http%3A%2F%2Fdsba.korea.ac.kr%2Fseminar%2F%3Fuid%3D1434%26mod%3Ddocument%26pageid%3D1&v=gCleQ9JxibI

Time series data components

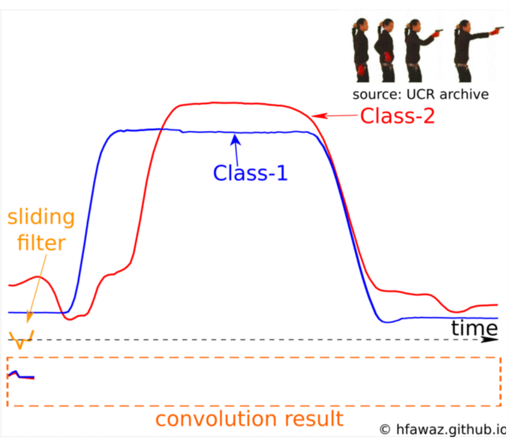
- **Trend**
- **Seasonality**
- **Noise**

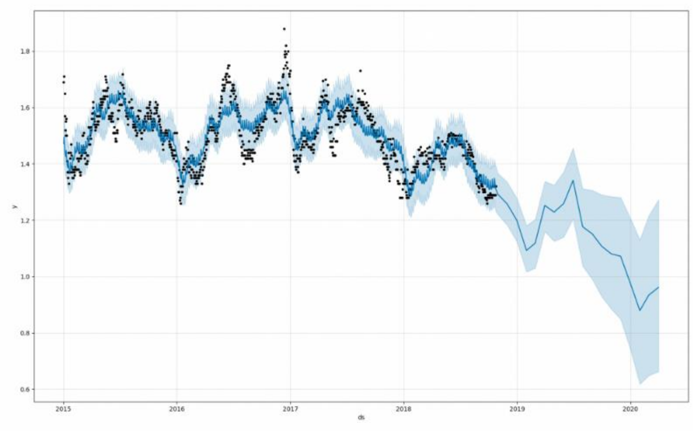


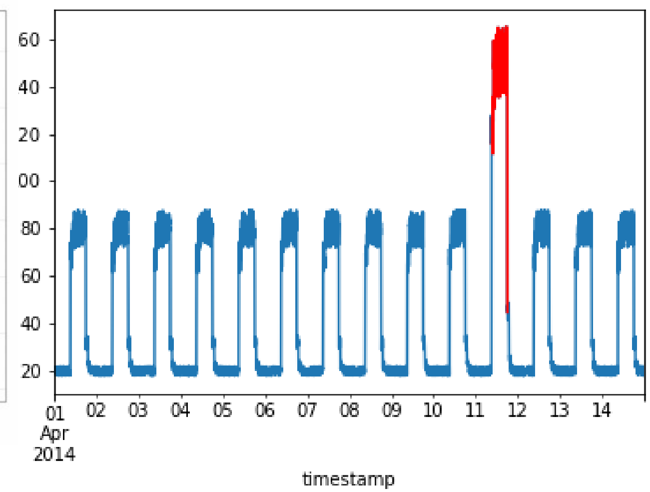
https://towardsdatascience.com/a-comprehensive-beginners-guide-to-the-diverse-field-of-anomaly-detection-8c818d153995

Time series tasks

- **1) Classification**
- **2) Forecasting**
- **3) Anomaly Detection**



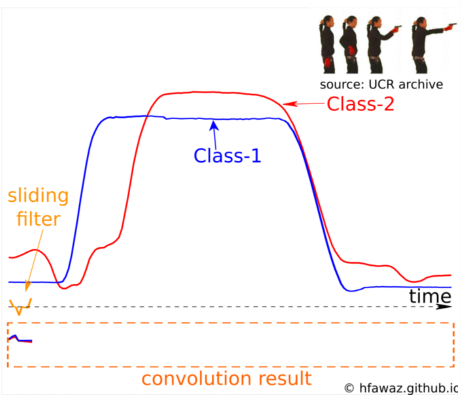


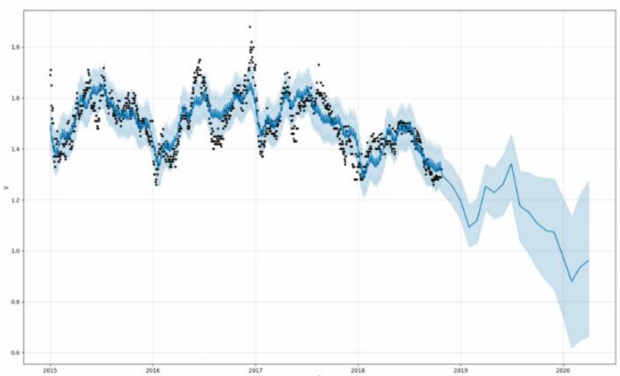


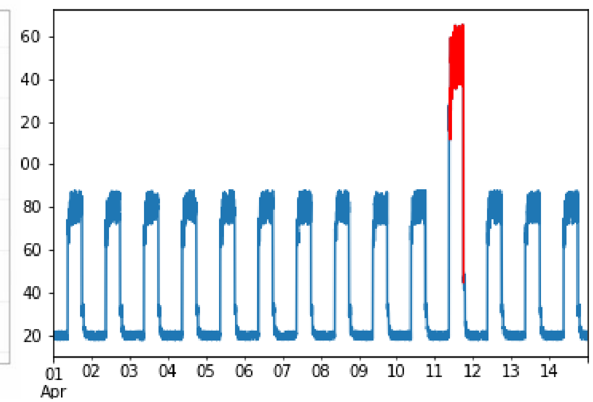
https://medium.com/@hassanismailfawaz/deep-learning-for-time-series-classification-a-brief-overview-73b58767ed0f
https://keras.io/examples/timeseries/timeseries_anomaly_detection/
https://syllepsis.live/2019/10/08/time-series-forecasting-a-quick-reference/

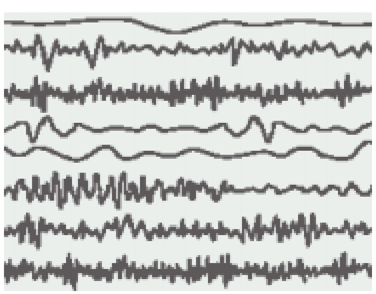
Time Series Tasks

- **1) Classification**
- **2) Forecasting**
- **3) Anomaly Detection**
  - 과거 data를 기반으로 t시점에서의 abnormal 여부 예측







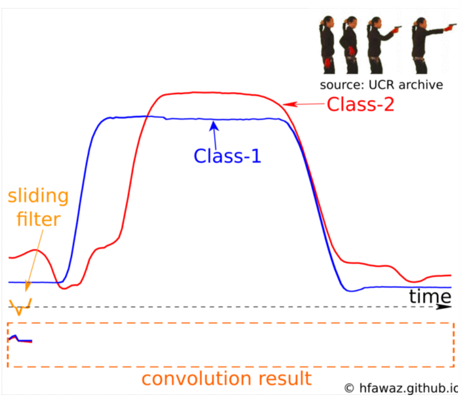


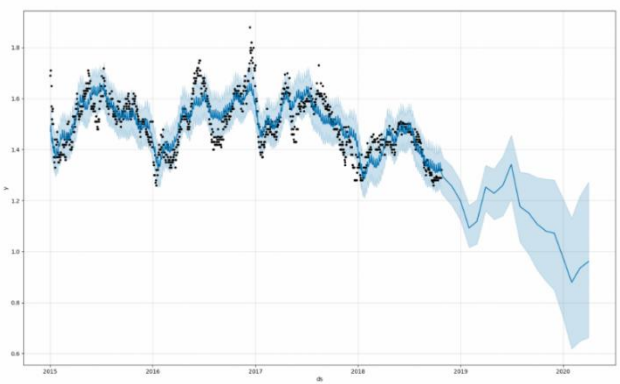
→ Unsupervised ML Model →

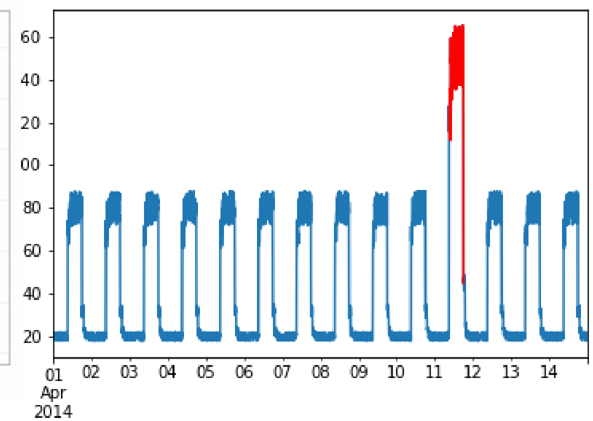
Anomalies	t_{start}	t_{stop}
1	Jan 10th, 2019 - 8:16 am	Jan 10th, 2019 - 3:34 pm
2	Jan 16th, 2019 - 11:16 am	Jan 17th, 2019 - 2:34 am
...	...	...
18	Mar 24th, 2019 - 2:12 pm	Mar 28th, 2019 - 3:19 pm

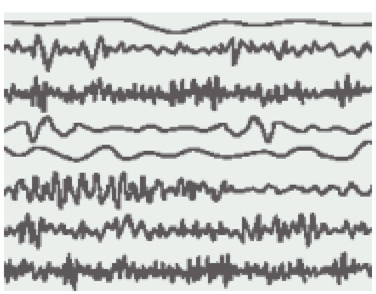
Time Series Tasks

- **3) Anomaly Detection**
  - 과거 data를 기반으로 t시점에서의 abnormal 여부 예측
  - 제조 공정 과정에서 불량품이나 기계 고장을 탐지
  - System Security 분야에서 보안이 위협받는 상황을 판별



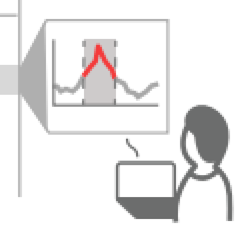






Unsupervised ML Model

Anomalies	t_{start}	t_{stop} timestamp
1	Jan 10th, 2019 - 8:16 am	Jan 10th, 2019 - 3:34 pm
2	Jan 16th, 2019 - 11:16 am	Jan 17th, 2019 - 2:34 am
...	...	...
18	Mar 24th, 2019 - 2:12 pm	Mar 28th, 2019 - 3:19 pm



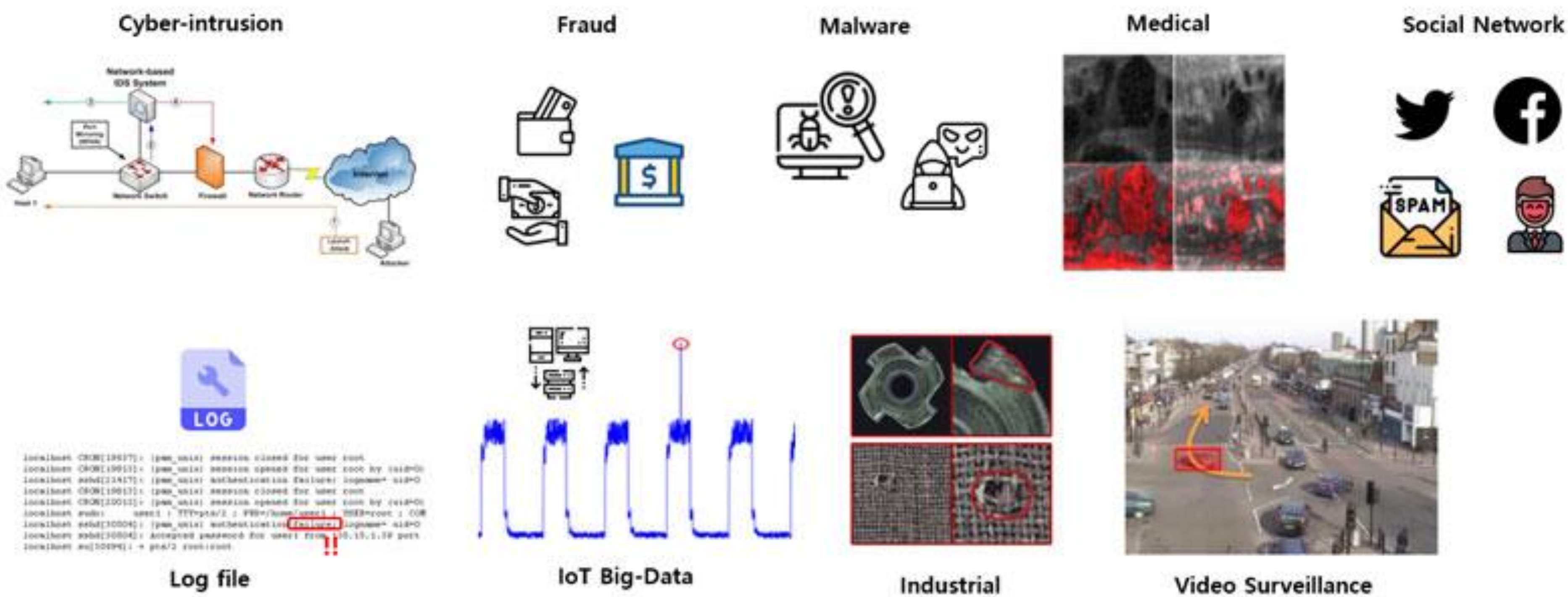
Anomaly

- **Anomaly Detection**
  - 정상 데이터와 본질적으로 다름
- **Novelty Detection**
  - 정상 데이터와 본질적으로 같지만 유형이 다름
- **Outlier Detection**
  - outlier: 일반적인 데이터 생성 매커니즘을 위배해서 만들어진 데이터
  - noise: 데이터 수집 관점에서 자연적으로 발생하는 변동성 (random error or variance)

https://yupsung.blogspot.com/2021/01/anomaly-detection-overview.html

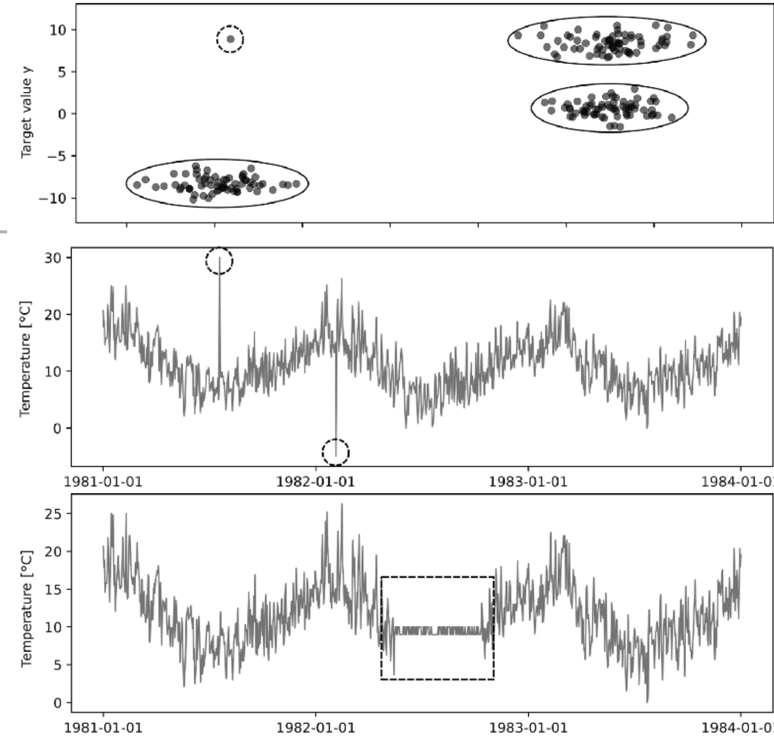
Anomaly Detection 적용 사례

- **Anomaly Detection**
  - 정상 데이터와 본질적으로 다름



https://arxiv.org/abs/1901.03407
https://www.cognex.com/ko-kr/blogs/deep-learning/research/anomaly-detection-overview-1-introduction-anomaly-detection

Types of Anomalies



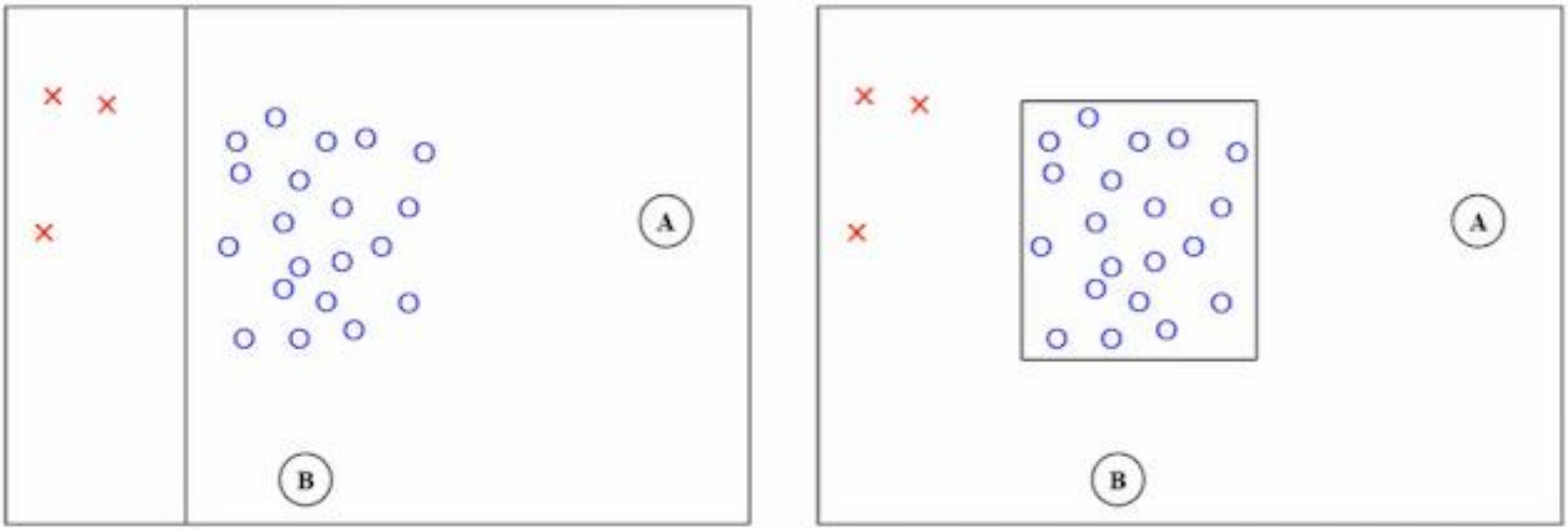
- **Point/Global Anomalies**
  - 대다수의 Data set과 완전히 다른 객체
  - ex)Credit card fraud detection
- **Contextual Anomalies**
  - 조건부(context) 이상치, 특정 조건이 충족될 때 이상치로 판단됨
  - ex)온도 29도는 우리나라에선 정상이지만, 북극에서는 특이한 케이스
- **Collective Anomalies**
  - 단일 객체들은 outlier가 아니지만 모아서 보면 편차가 심한 케이스
  - ex)DDos공격. 단일 접속 패킷을 정상, 한번에 접속이 너무 많으면 서버 다운

https://towardsdatascience.com/a-comprehensive-beginners-guide-to-the-diverse-field-of-anomaly-detection-8c818d153995
https://yupsung.blogspot.com/2021/01/anomaly-detection-overview.html

What is Anomaly?

- **데이터 생성 매커니즘**
  - 일반적인 데이터와 다른 매커니즘으로 발생한 data
- **데이터 분포**
  - Data가 발생할 확률 밀도가 매우 낮은 data

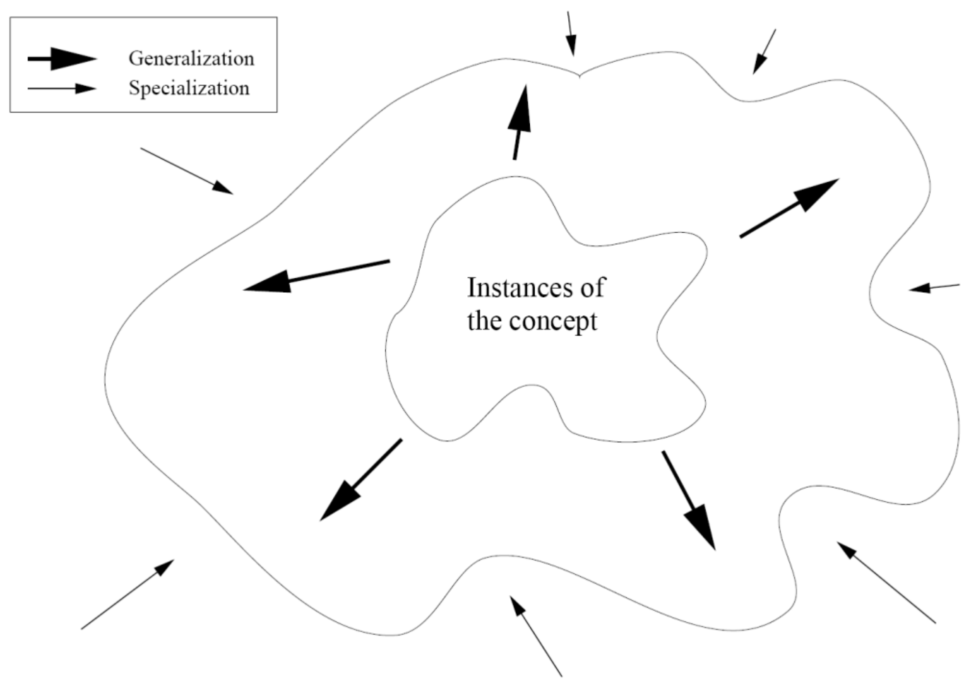
Classification vs Anomaly Detection

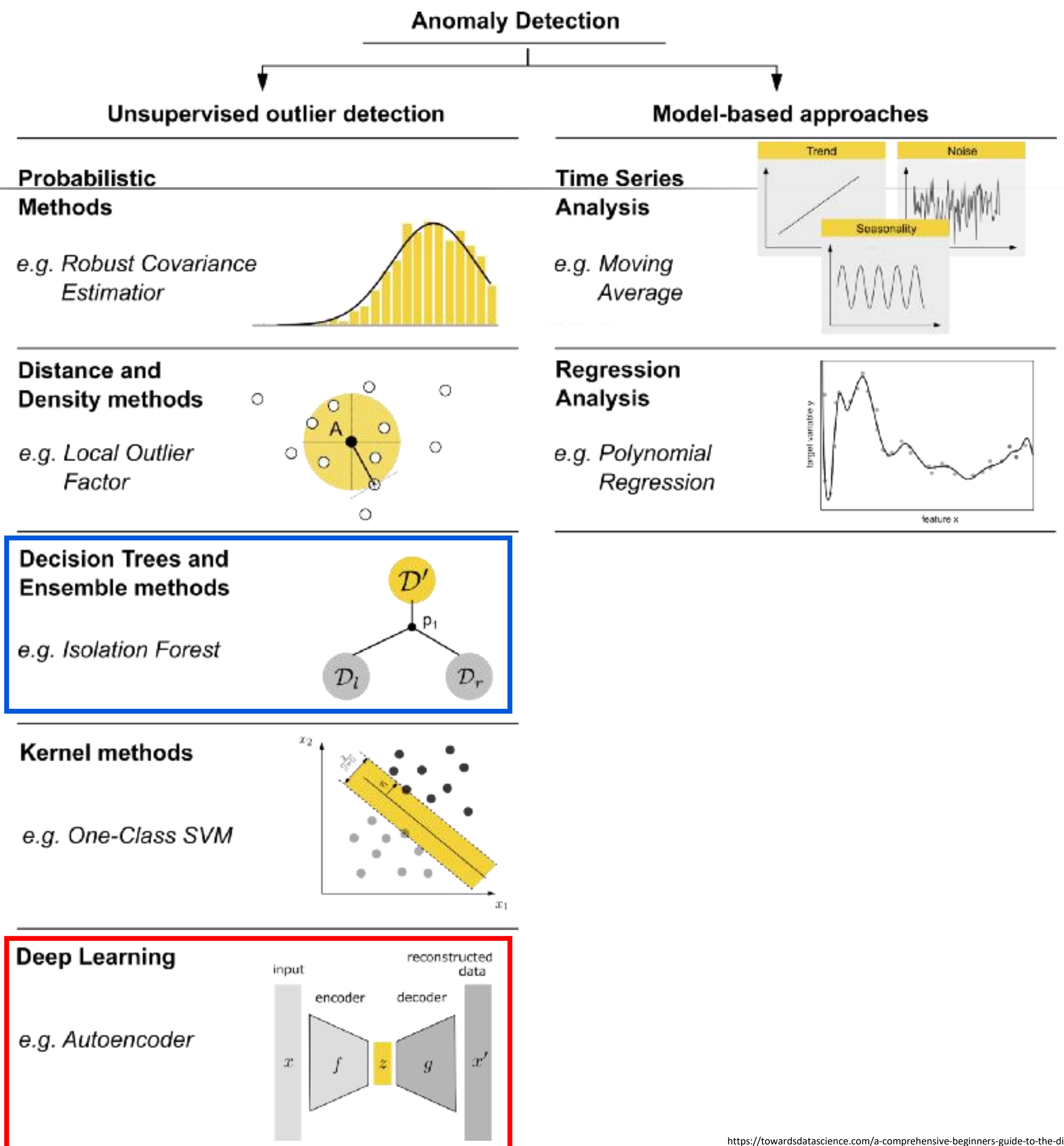


Binary classification Anomaly detection

Generalization vs Specialization

- **Generalization: 일반화**
  - abnormal data를 normal로 오분류
- **Specialization: 구체화, 특수화**
  - normal data를 abnormal로 오분류
- **Trade off 관계. 조절 필요**

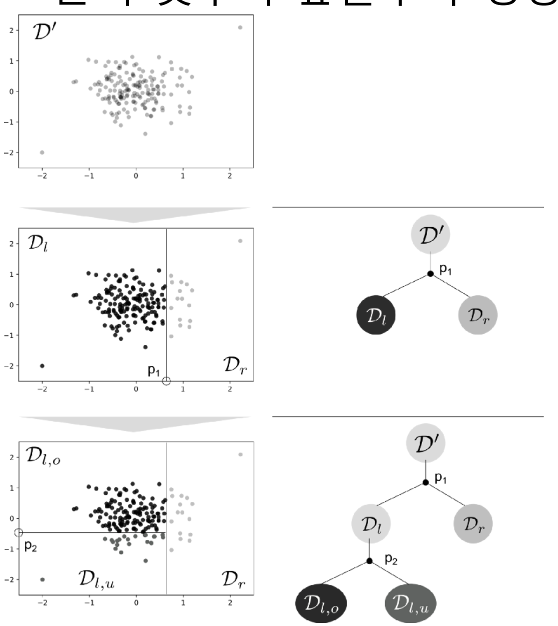


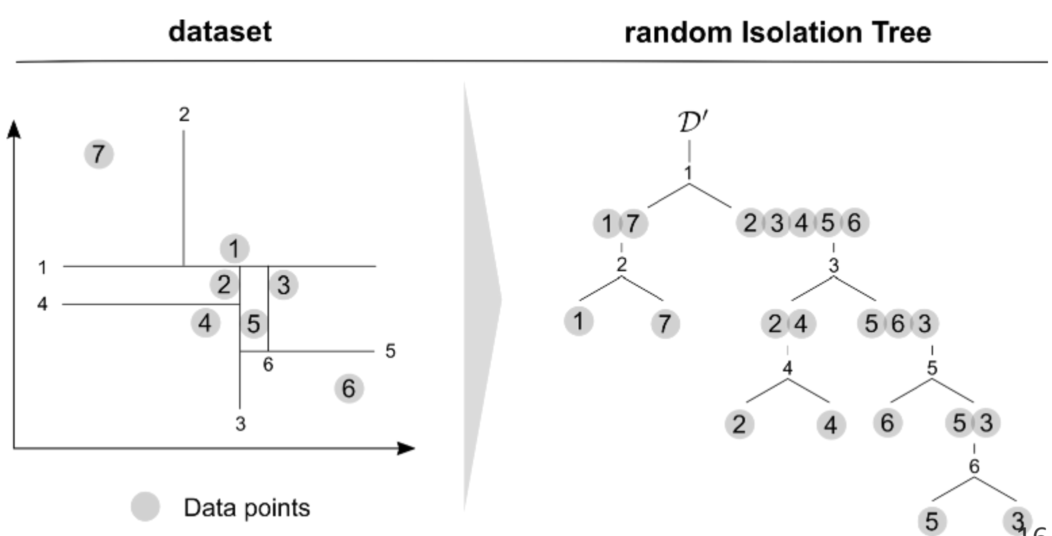


https://towardsdatascience.com/a-comprehensive-beginners-guide-to-the-diverse-field-of-anomaly-detection-8c818d153995

Isolation Forest

- 정상/비정상을 가르는 기준은 해당 데이터를 isolation(고립) 하는데 걸린 평균 분기 횟수로 예측
- 정상 데이터는 밀집 지역에 분포
- 비정상 데이터는 그로부터 떨어진 밀도가 낮은 지역에 분포
- 분기 횟수가 적을 수록 비정상
- 분기 횟수가 높을수록 정상

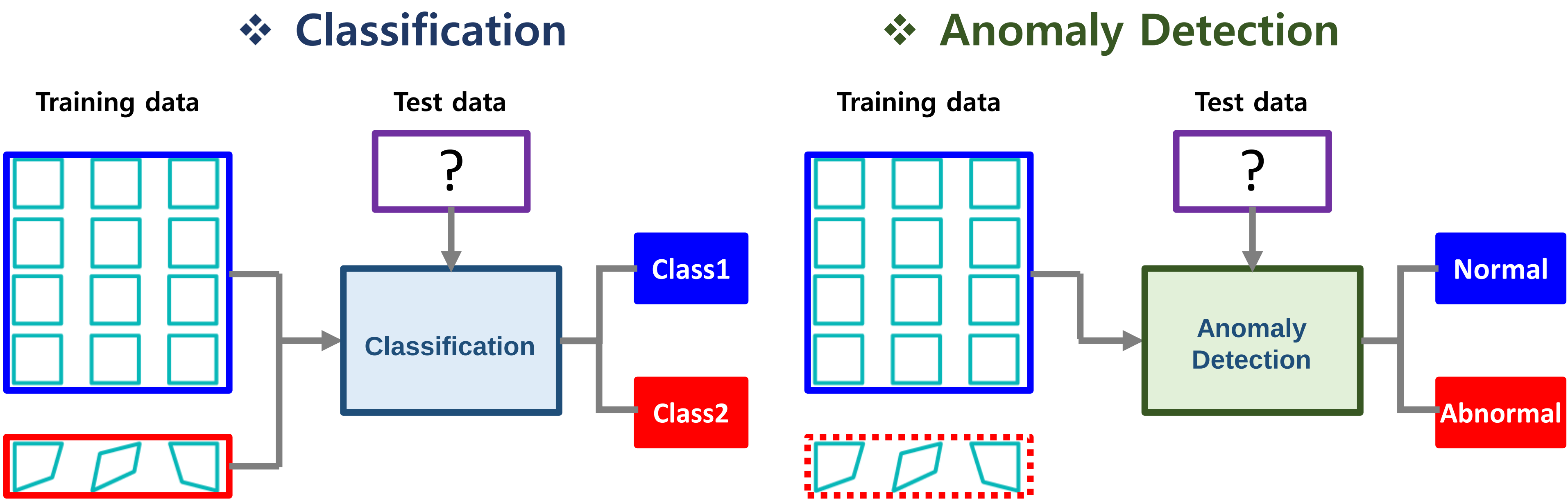




https://towardsdatascience.com/a-comprehensive-beginners-guide-to-the-diverse-field-of-anomaly-detection-8c818d153995

Classification VS Anomaly Detection

- Normal data가 Abnormal data 보다 많은 상황 전제
- Training: Only normal data만으로 모델 training
- Test: Normal+Abnomal data로 테스트



https://towardsdatascience.com/a-comprehensive-beginners-guide-to-the-diverse-field-of-anomaly-detection-8c818d153995

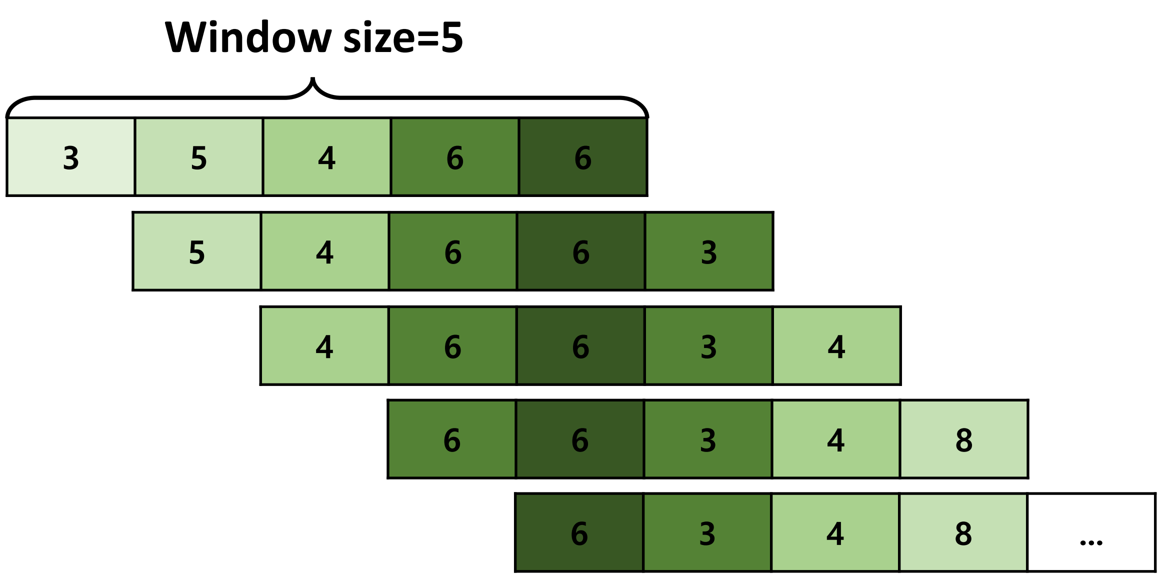
Classification VS Anomaly Detection

- **Severe data imbalance**
  - minority class samples이 조금은 있을 때 (100개 이상)
    - Classification with Oversampling
  - minority class samples이 너무 적을 때(5~10개)
    - Anomaly Detection!!

Data setup

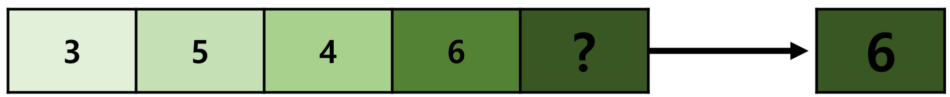
- **Sliding window**
  - Long sequence to small window sequence
  - Sliding window size: 5

Time stamp	1	2	3	4	5	6	7	8	…
Value	3	5	4	6	6	3	4	8	…

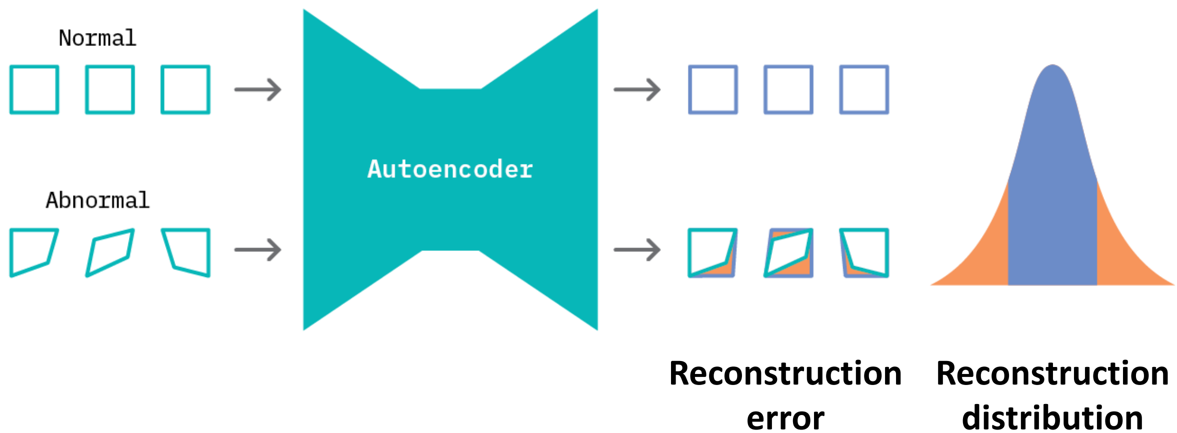


Anomaly Detection Phases

- **Phase 1) 데이터 분포 학습**
  - Seq2Seq



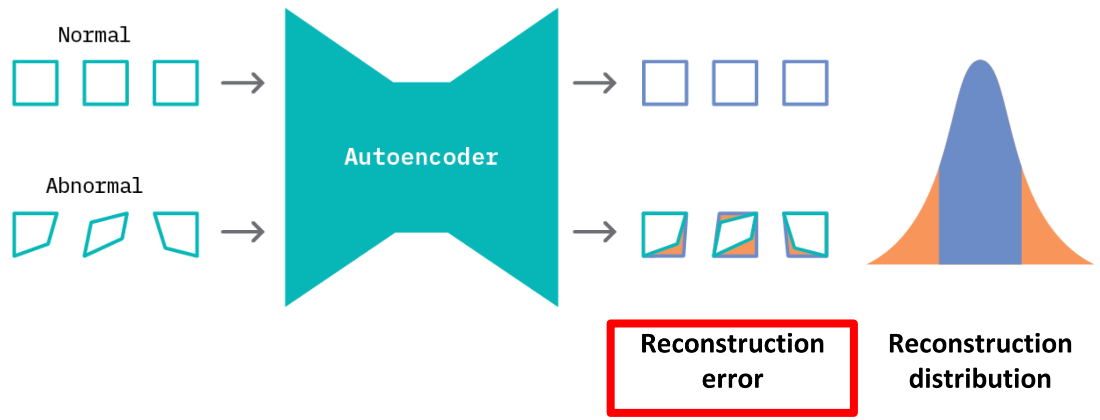
  - AutoEncoder

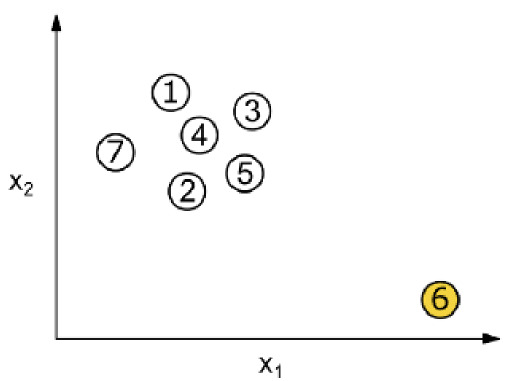


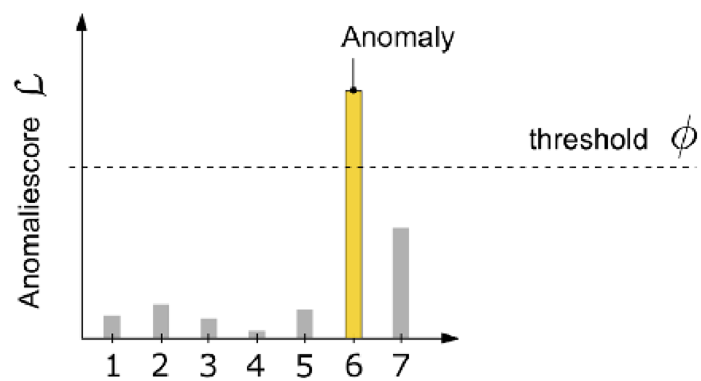
- **Phase 2) Anomaly score를 구한 후 Threshold 기준으로 판별**

Anomaly Detection Phases

- Phase 1) 데이터 분포 학습
- Phase 2) Anomaly score를 구한 후 Threshold 기준으로 판별



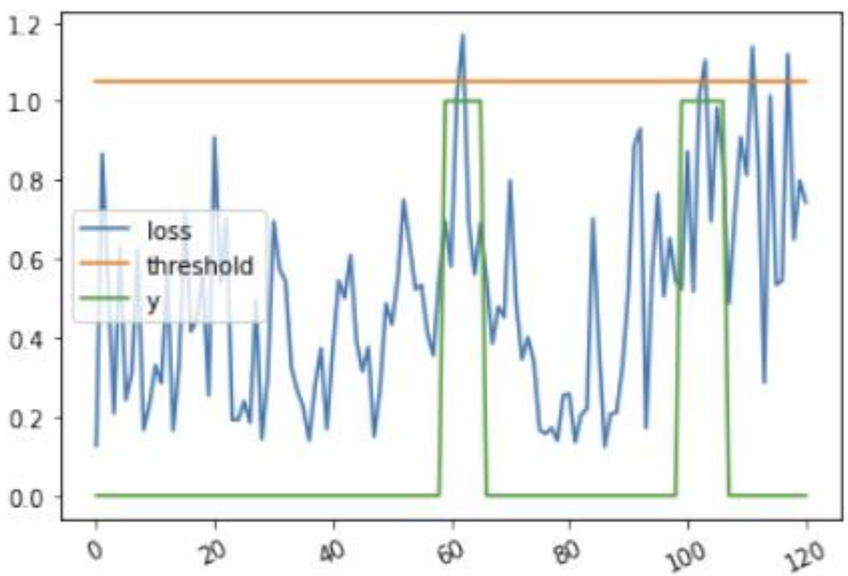


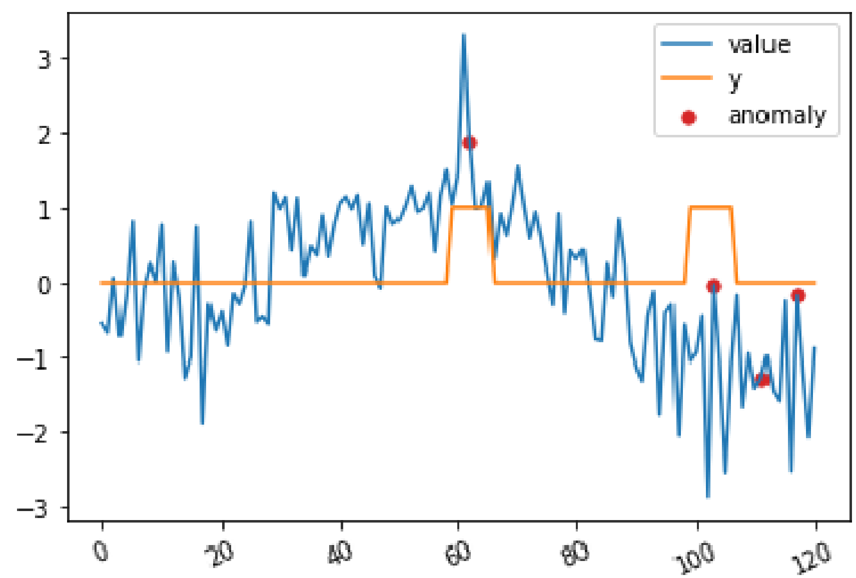


https://towardsdatascience.com/a-comprehensive-beginners-guide-to-the-diverse-field-of-anomaly-detection-8c818d153995

Anomaly Detection Phases

- **Phase 1) 데이터 분포 학습**
- **Phase 2) Anomaly score를 구한 후 Threshold 기준으로 판별**



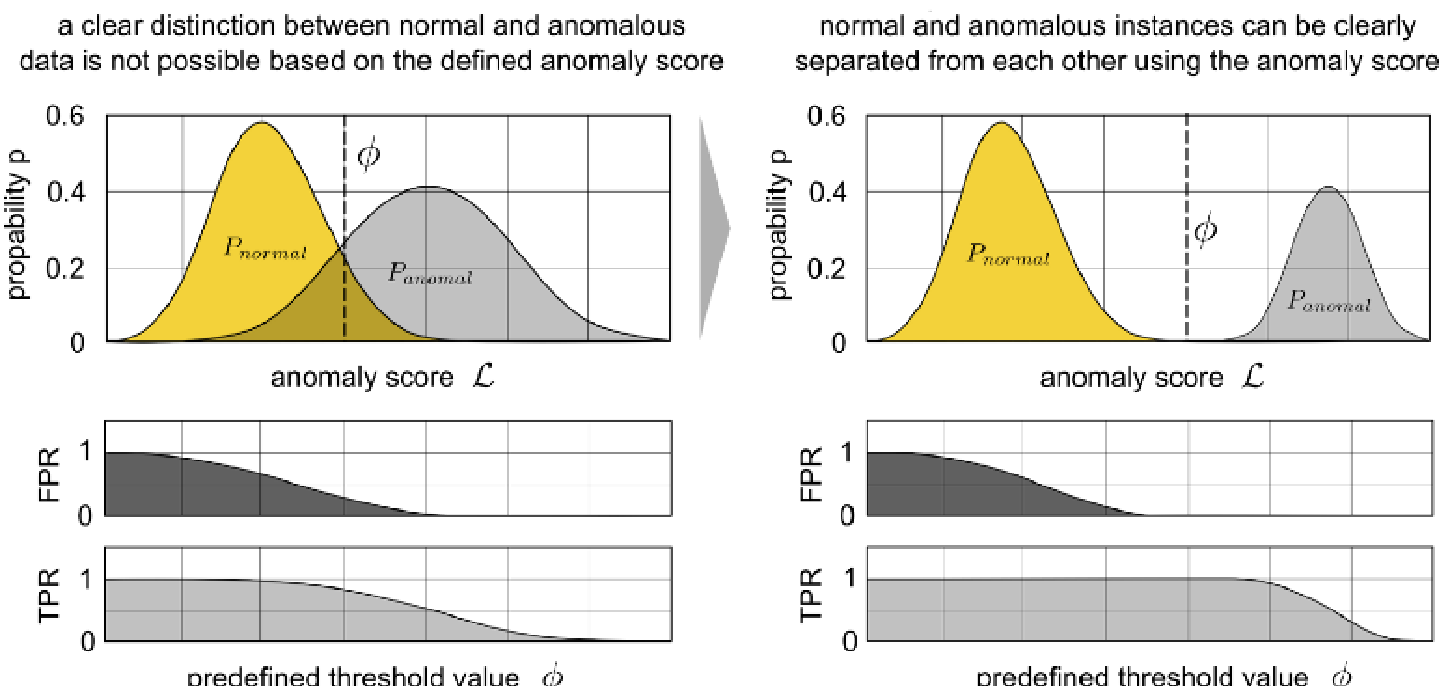


https://arxiv.org/pdf/2008.09567.pdf

Threshold

- The threshold for the anomaly score defines the sensitivity of the system
- Designation of instances as normal/anomalous based on their anomaly score and the predefined threshold value

P_{normal}	Probability distribution of normal data	TPR	True Positive Rate
P_{anomal}	Probability distribution of anomalous data	FPR	False Positive Rate



https://towardsdatascience.com/a-comprehensive-beginners-guide-to-the-diverse-field-of-anomaly-detection-8c818d153995

Challenges

- Normal과 abnormal data 사이의 경계가 모호 (Gray area)
- 연속한 데이터에서 어디를 경계로 설정할지 결정하기 어려움
- Normal과 abnormal을 명확히 구분하는 설명이 어려움
- 시간이 흘러 새로운 정상 패턴이 생겼을 때, 과거 데이터로는 비정상이라고 탐지할 확률이 높음

감사합니다

USAD

Unsupervised Anomaly Detection on Multivariate Time Series

KDD 2020

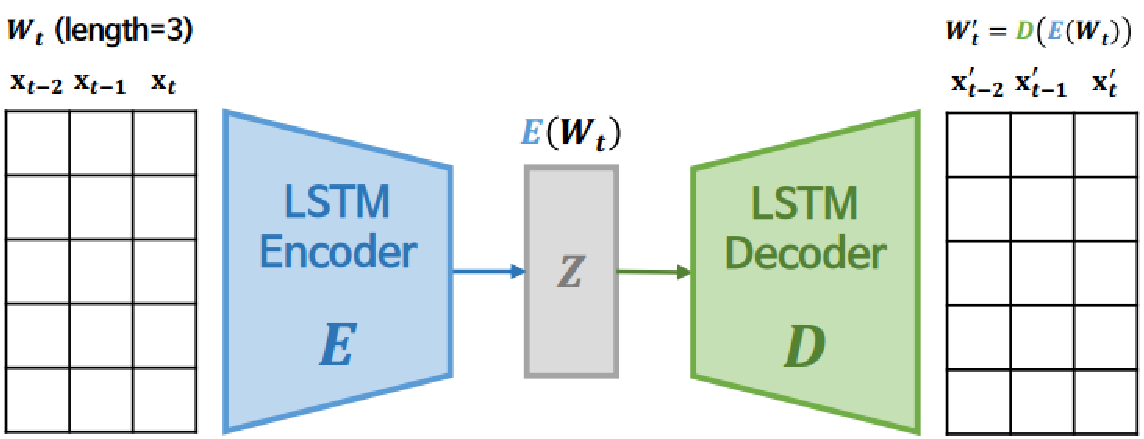
영상: https://youtu.be/gCleQ9JxibI

논문: https://dl.acm.org/doi/pdf/10.1145/3394486.3403392

코드: https://github.com/manigalati/usad/blob/master/USAD.ipynb

AE-based Multivariate Time Series AD

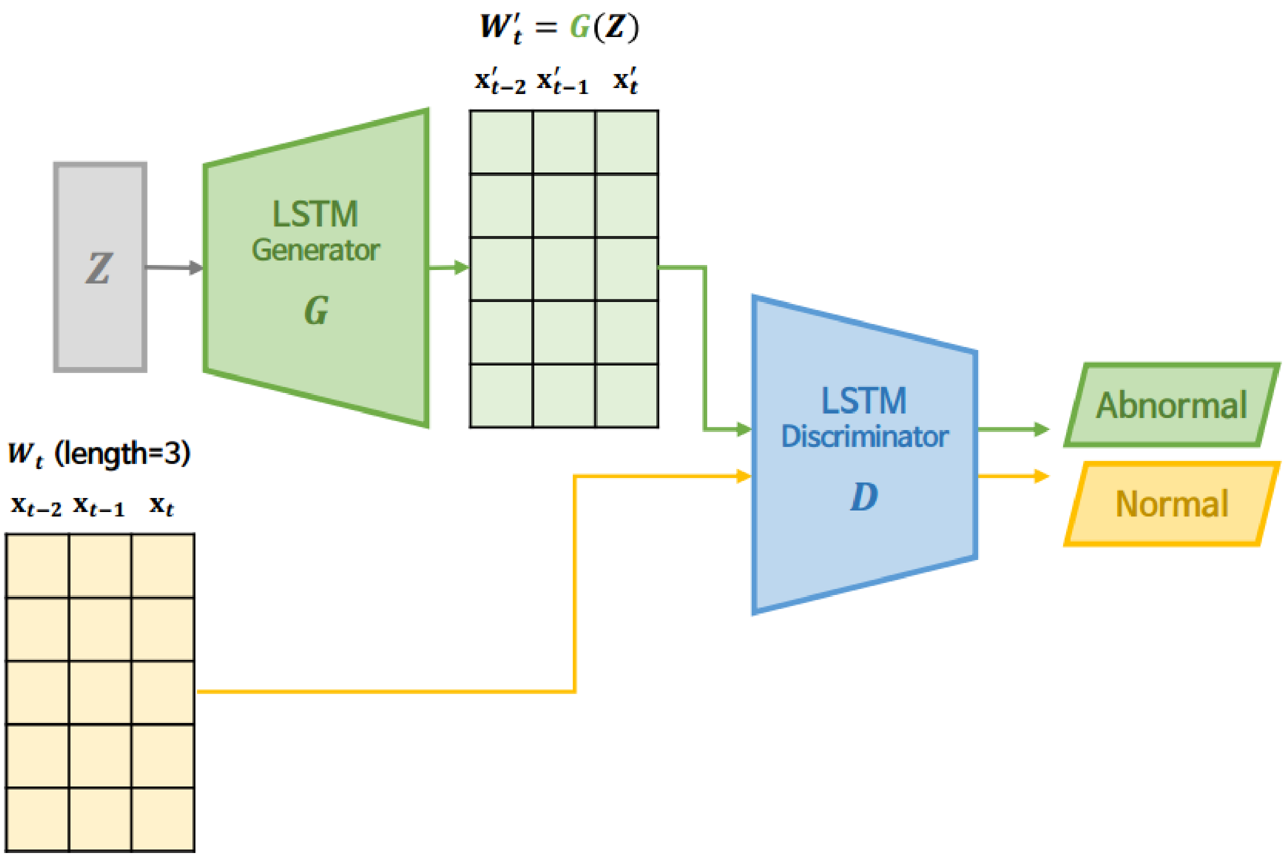
- **LSTM-AE (Unsupervised)**
  - Training: 정상 데이터의 reconstruction error를 기반으로 LSTM-AE를 학습하여 정상 데이터의 분포를 학습
  - Anomaly detection: 학습이 완료된 LSTM-AE를 기반으로 도출한 새로운 입력의 reconstruction error가 threshold를 초과하면 이상치로 탐지함



$$Anomaly\ Score = \|W_t - D(E(W_t))\|_2$$

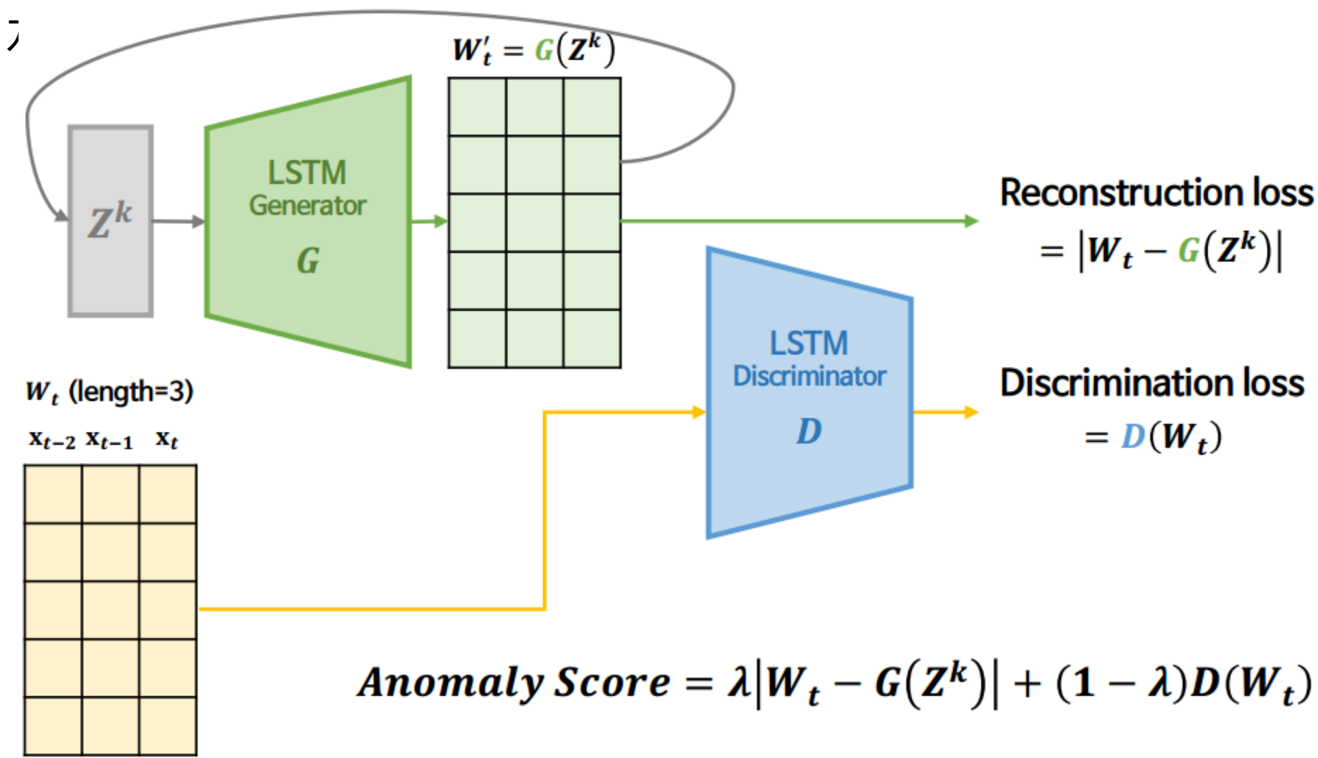
GAN-based Multivariate Time Series AD

- **MAD-GAN (Unsupervised)**
  - Training: 정상 데이터만으로 LSTM 구조의 generator와 discriminator를 학습하여 정상 데이터의 분포를 학습



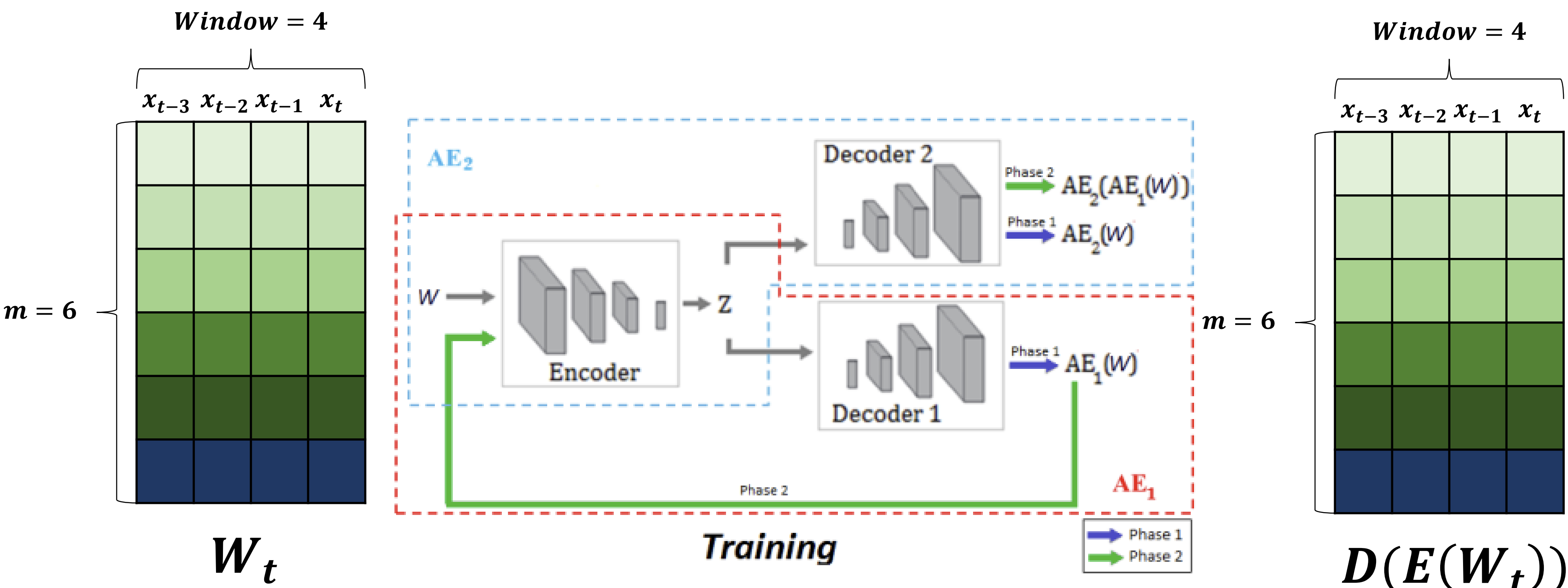
GAN-based Multivariate Time Series AD

- **MAD-GAN (Unsupervised)**
  - Anomaly detection: 새로운 입력의 optimal latent space를 기반으로 생성한 reconstructed sample과 입력의 reconstruction loss와 입력에 대한 discrimination loss의 가중 합이 특정 threshold를 초과하면 이상치로 탐



USAD: Training

- **Phase 1: AutoEncoder Training**
  - ***Encoder***가 입력 $\boldsymbol{W_t}$를 latent space Z로 압축
  - ***Decoder***가 latent space Z 를 입력 $\boldsymbol{W_t}$와 동일하게 복원

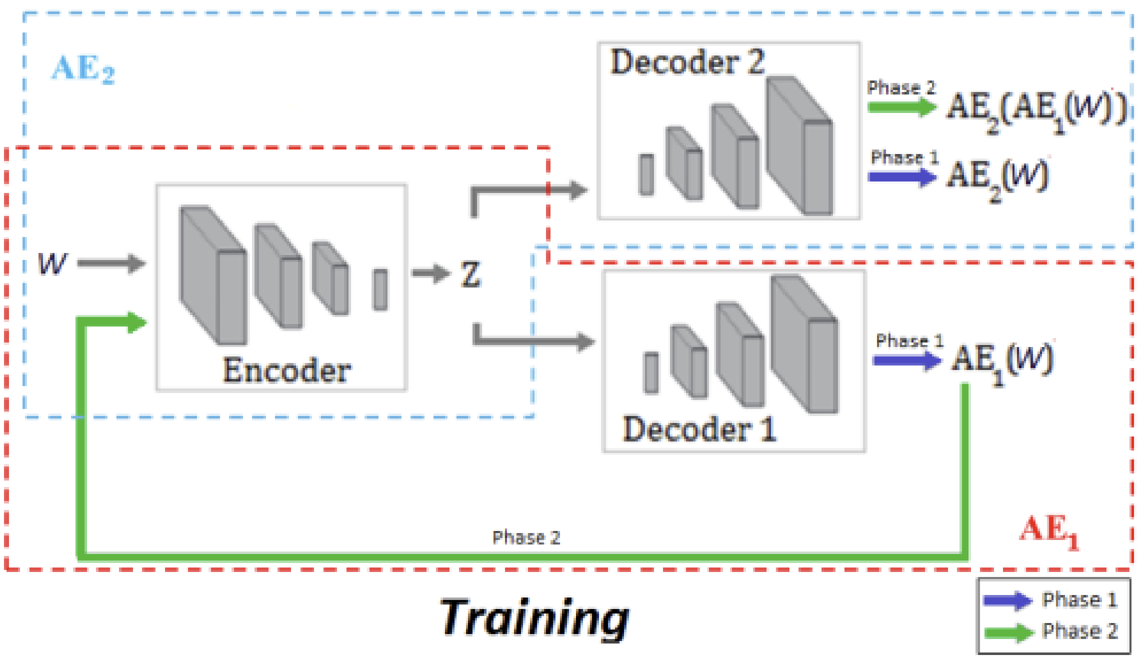


$$Loss_1 = \|W_t - D_1(E(W_t))\|_2$$

$$Loss_2 = \|W_t - D_2(E(W_t))\|_2$$

USAD: Training

- **Phase 2: Adversarial Training**
  - $\boldsymbol{AE_1}$: 입력($\boldsymbol{W_t}$)을 잘 복원하면서 $\boldsymbol{AE_2}$를 잘 속이도록 적대적 학습
  - $\boldsymbol{AE_2}$: 입력($\boldsymbol{W_t}$)을 잘 복원하면서 $\boldsymbol{AE_1}$이 복원한 입력을 잘 구별하도록 학습

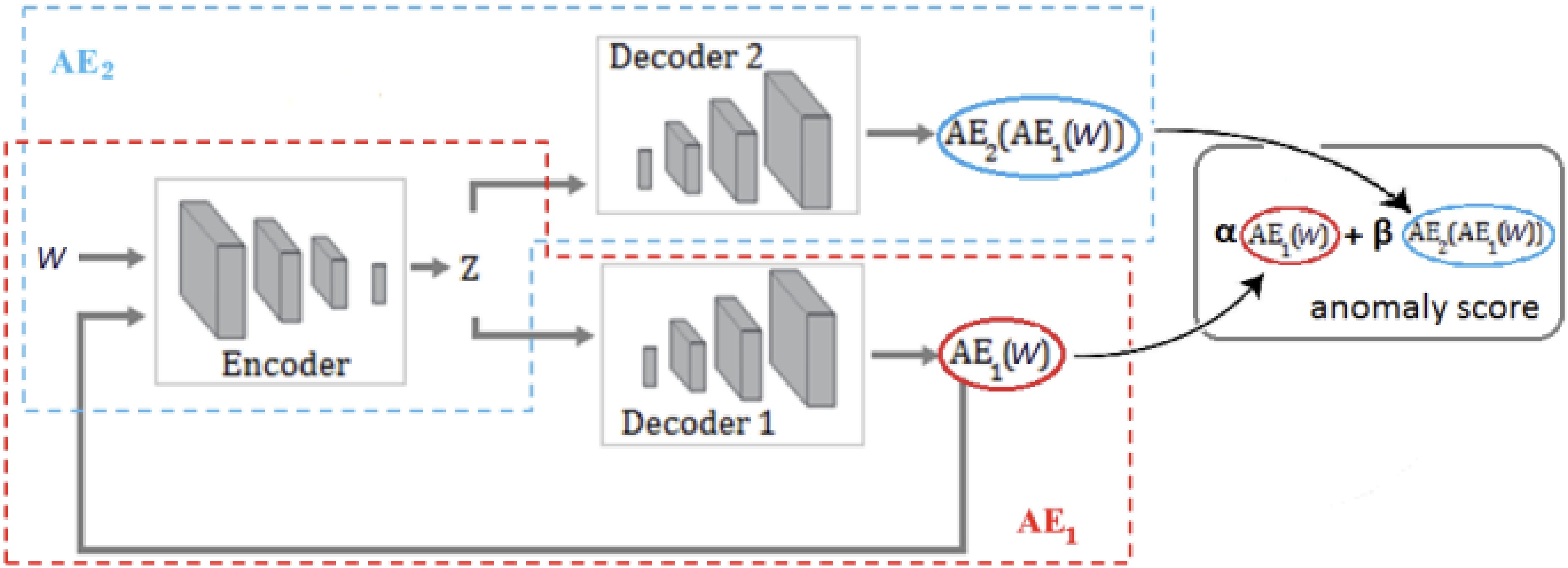


$$\min_{AE_1} \|W_t - AE_2(AE_1(W_t))\|_2$$

$$\max_{AE_2} \|W_t - AE_2(AE_1(W_t))\|_2$$

USAD: Detection

- **Anomaly Score**



Detection

USAD: Detection

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

- **Anomaly Score**
  - Anomaly Score의 α, β는 비례 계수(합이 1)로
  - 비중에 따라 TP/FP 간에 trade-off
    - a>b: TP/FP 감소 -> low detection sensitivity scenario
    - a<b: TP/FP 증가 -> high detection sensitivity scenario

$$\text{Anomaly Score} = \alpha \left\| \widehat{W}_t - AE_1(\widehat{W}_t) \right\|_2 + \beta \left\| \widehat{W}_t - AE_2\left(AE_1(\widehat{W}_t)\right) \right\|_2$$

α	β	FP	TP	F1
0.0	1.0	604	35,616	0.7875
0.1	0.9	580	35,529	0.7853
0.2	0.8	571	35,285	0.7833
0.5	0.5	548	34,590	0.7741
0.7	0.3	506	34,548	0.7738
0.9	0.1	299	34,028	0.7684

USAD: Encoder

A.4.1 Encoder.

- Linear : input size -> input size / 2
- Relu
- Linear : input size /2 -> input size / 4
- Relu
- Linear : input size /4 -> latent space size
- Relu

```
class Encoder(nn.Module):
    def __init__(self, in_size, latent_size):
        super().__init__()
        self.linear1 = nn.Linear(in_size, int(in_size/2))
        self.linear2 = nn.Linear(int(in_size/2), int(in_size/4))
        self.linear3 = nn.Linear(int(in_size/4), latent_size)
        self.relu = nn.ReLU(True)

    def forward(self, w):
        out = self.linear1(w)
        out = self.relu(out)
        out = self.linear2(out)
        out = self.relu(out)
        out = self.linear3(out)
        z = self.relu(out)
        return z
```

A.4.2 Decoder. Both decoders have the same architecture.

- Linear : latent space size -> input size / 4
- Relu
- Linear : input size /4 -> input size / 2
- Relu
- Linear : input size /4 -> input size
- Sigmoid

```
class Decoder(nn.Module):
    def __init__(self, latent_size, out_size):
        super().__init__()
        self.linear1 = nn.Linear(latent_size, int(out_size/4))
        self.linear2 = nn.Linear(int(out_size/4), int(out_size/2))
        self.linear3 = nn.Linear(int(out_size/2), out_size)
        self.relu = nn.ReLU(True)
        self.sigmoid = nn.Sigmoid()

    def forward(self, z):
        out = self.linear1(z)
        out = self.relu(out)
        out = self.linear2(out)
        out = self.relu(out)
        out = self.linear3(out)
        w = self.sigmoid(out)
        return w
```

USAD: Training

Algorithm 1 USAD training algorithm

Input: Normal windows Dataset $\mathcal{W} = \{W_1, ..., W_T\}$, Number epochs N
Output: Trained AE_1, AE_2
  $E, D_1, D_2 \leftarrow$ initialize weights
  $n \leftarrow 1$
  repeat
    for $t = 1$ to T **do**
      $Z_t \leftarrow E(W_t)$
      $W_t^{1'} \leftarrow D_1(Z_t)$
      $W_t^{2'} \leftarrow D_2(Z_t)$
      $W_t^{2''} \leftarrow D_2(E(W_t^{1'}))$
      $\mathcal{L}_{AE_1} \leftarrow \frac{1}{n}\left\|W_t - W_t^{1'}\right\|_2 + \left(1 - \frac{1}{n}\right)\left\|W_t - W_t^{2''}\right\|_2$
      $\mathcal{L}_{AE_2} \leftarrow \frac{1}{n}\left\|W_t - W_t^{2'}\right\|_2 - \left(1 - \frac{1}{n}\right)\left\|W_t - W_t^{2''}\right\|_2$
      $E, D_1, D_2 \leftarrow$ update weights using $\mathcal{L}_{AE_1}$ and $\mathcal{L}_{AE_2}$
    end for
    $n \leftarrow n + 1$
  until $n = N$

```
def training_step(self, batch, n):
    z = self.encoder(batch)
    w1 = self.decoder1(z)
    w2 = self.decoder2(z)
    w3 = self.decoder2(self.encoder(w1))
    loss1 = 1/n*torch.mean((batch-w1)**2)+(1-1/n)*torch.mean((batch-w3)**2)
    loss2 = 1/n*torch.mean((batch-w2)**2)-(1-1/n)*torch.mean((batch-w3)**2)
    return loss1,loss2
```

USAD: Detection

Algorithm 2 USAD Detection algorithm

Input: Test windows Dataset $\widehat{\mathcal{W}}$: $(\widehat{W}_1, ..., \widehat{W}_{T^*})$, threshold λ , parameters α and β

Output: Labels $\mathbf{y}$: $\{y_1, ..., y_{T^*}\}$

for $t = 1$ to T^* **do**

$\widehat{W}_t^{1'} \leftarrow D_1(E(\widehat{W}_t))$

$\widehat{W}_t^{2''} \leftarrow D_2(E(\widehat{W}_t^{1'}))$

$\mathscr{A} \leftarrow \alpha \left\| \widehat{W}_t - \widehat{W}_t^{1'} \right\|_2 + \beta \left\| \widehat{W}_t - \widehat{W}_t^{2''} \right\|_2$

if $\mathscr{A} \geq \lambda$ **then**

$y_t \leftarrow 1$

else

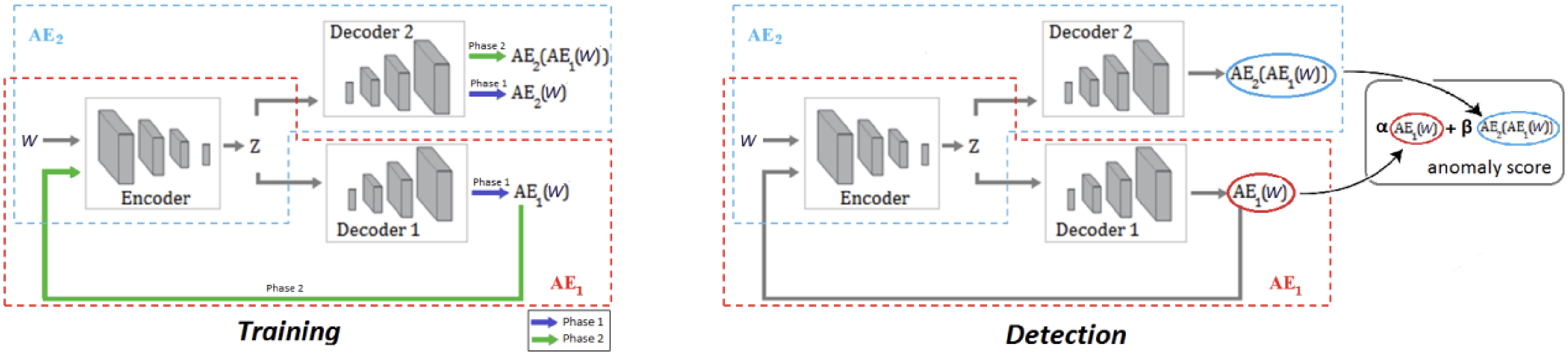
$y_t \leftarrow 0$

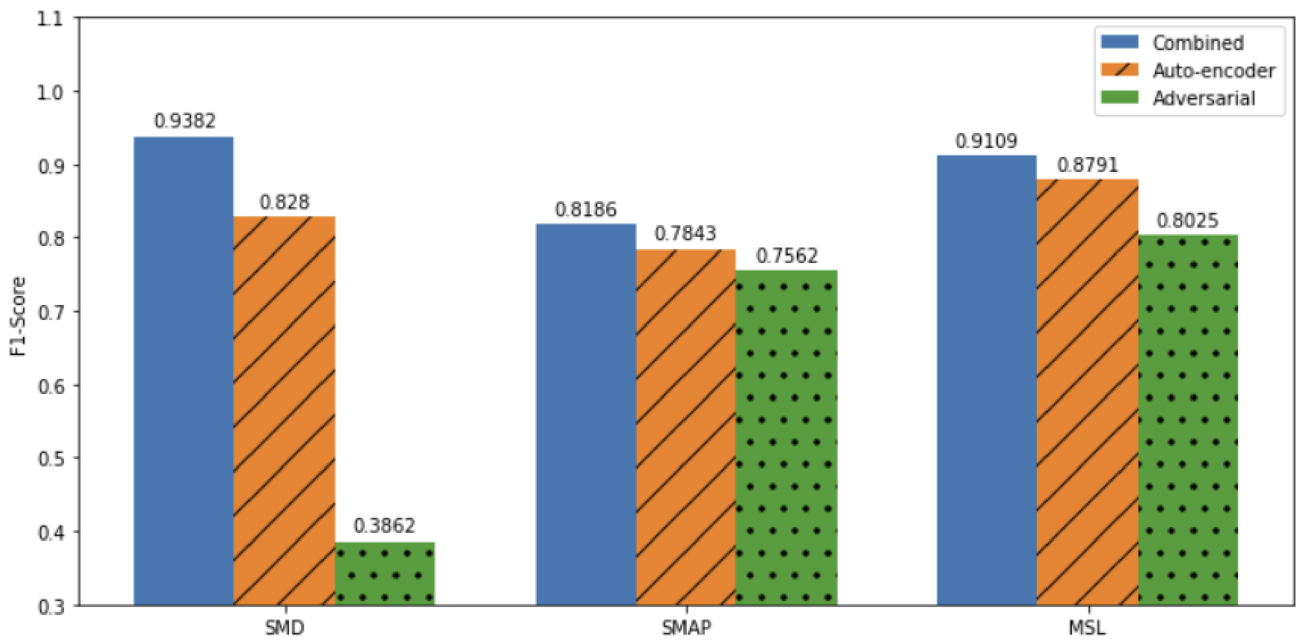
end if

end for

```
def testing(model, test_loader, alpha=.5, beta=.5):
    results=[]
    for [batch] in test_loader:
        batch=to_device(batch,device)
        w1=model.decoder1(model.encoder(batch))
        w2=model.decoder2(model.encoder(w1))
        results.append(alpha*torch.mean((batch-w1)**2,axis=1)+beta*torch.mean((batch-w2)**2,axis=1))
    return results
```

USAD: w/, wo adversarial training





Transformer/GRU-based prediction

Unsupervised Anomaly Detection on Multivariate Time Series

KDD 2020

GRU-based code: https://dacon.io/competitions/official/235757/codeshare/3086?page=2&dtype=recent

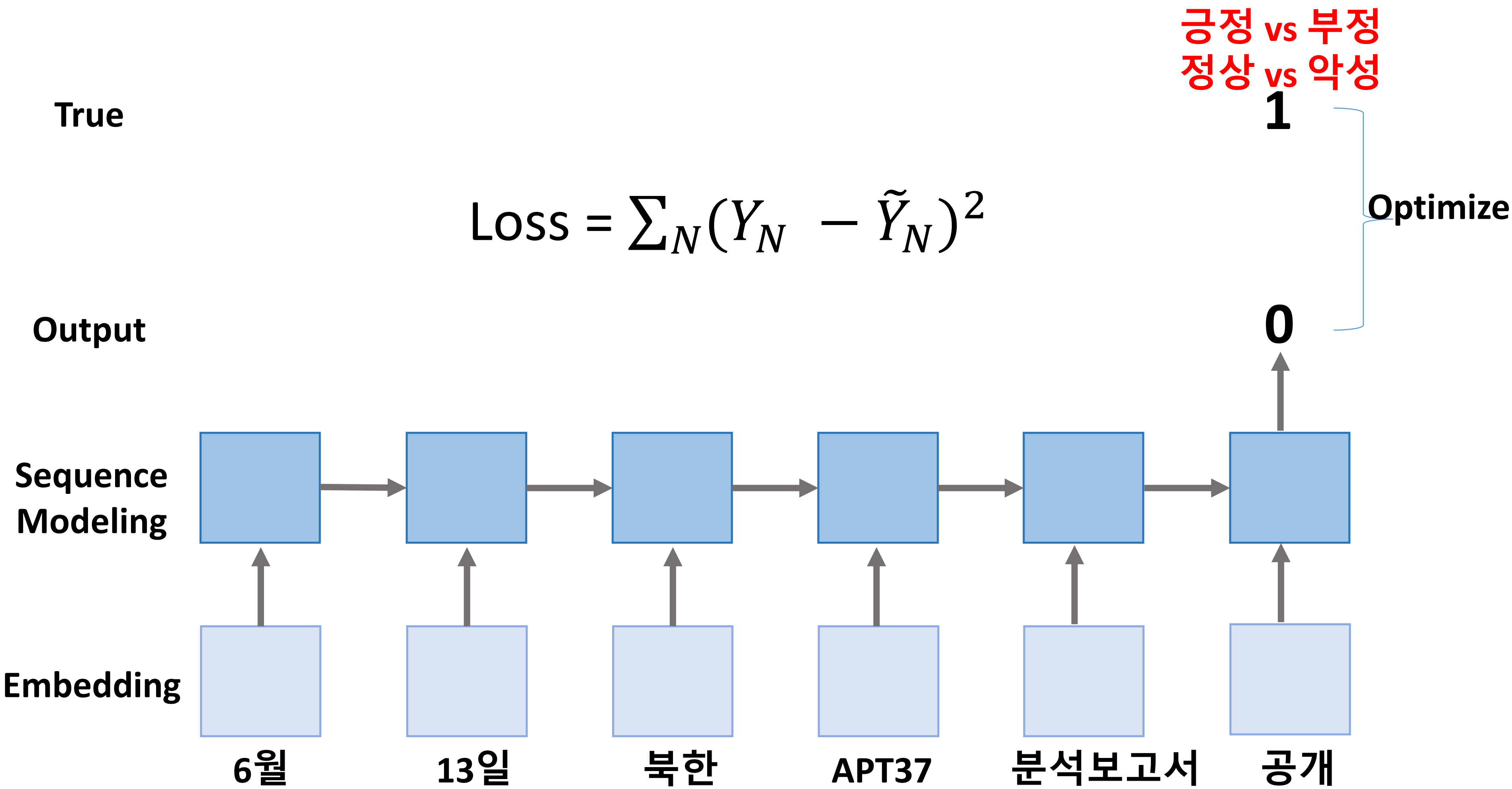
Transformer-based code:
https://dacon.io/competitions/official/235757/codeshare/3244?page=1&dtype=recent

Data setup

- **Sliding window**
  - Long sequence to small window sequence
  - Sliding window size: 5

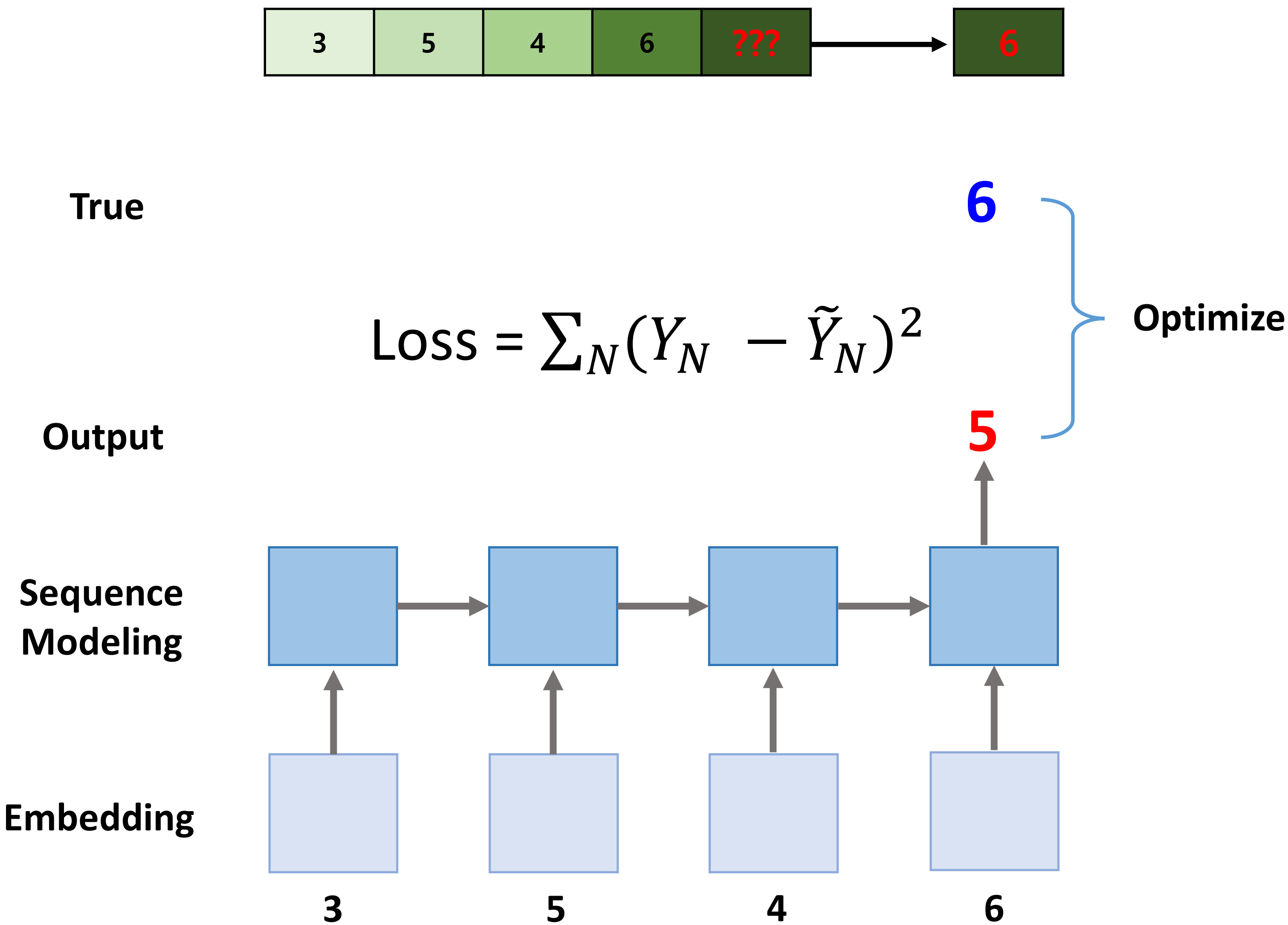
Time stamp	1	2	3	4	5	6	7	8	...
Value	3	5	4	6	6	3	4	8	...

일반적인 NLP 문제(N to 1)



$$\text{Loss} = \sum_N (Y_N - \tilde{Y}_N)^2$$

일반적인 NLP 문제(N to 1)



$$\text{Loss} = \sum_N (Y_N - \tilde{Y}_N)^2$$

감사합니다