고려대학교
빅데이터 연구회

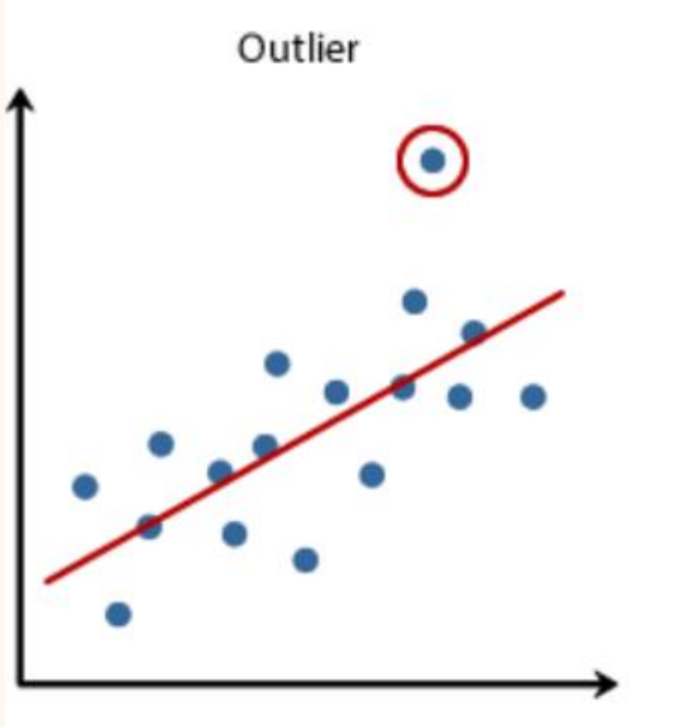
KU-BIG

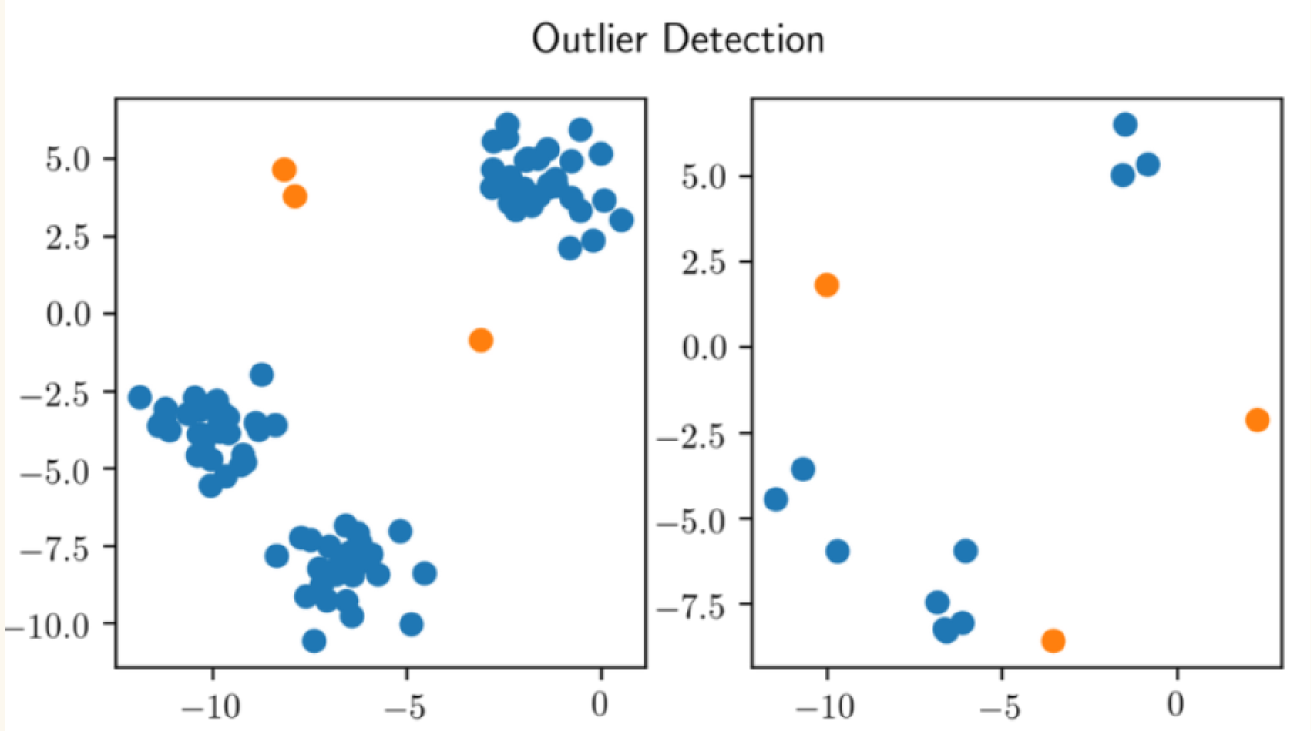
Outlier Detection

유현우 박정진 정희정 송예은 심정은 양수형

Outlier Detection

이상 감지





Outlier Detection

목 차

PART. 1 Outlier / Novelty / Anomaly

1

Outlier

Novelty

Anomaly

2

Outlier

(Anomaly)

Detection

3

Novelty

Detection

1) Outlier / Novelty / Anomaly

Outliers are also referred to as abnormalities, discordants, deviants, or **anomalies** in the data mining and statistics literature.

(Source: "Outlier Analysis" (Springer), Charu Aggarwal, 2017, http://charuaggarwal.net/outlierbook.pdf)

Outlier = Anomaly

2) Outlier(Anomaly) detection

The training data **contains outliers** which are defined as observations that are far from the others.

(Source: https://scikit-learn.org/stable/modules/outlier_detection.html)

2) Outlier(Anomaly) detection

i) Unsupervised anomaly detection

ii) Supervised anomaly detection

iii) Semi-Supervised

3) Novelty detection

The training data **is not polluted by outliers** and we are interested in detecting whether a new observation is an outlier.

Novelty

(Source: https://scikit-learn.org/stable/modules/outlier_detection.html)

PART. 2 Evaluate Measure

1

Metric

For

Out-of-Distribution Detection

2

Better metric

For

Class-imbalanced data

1) Metric for Out-of-Distribution Detection

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	a	b
Class=No	c	d

a: TP (True Positive)

b: FN (False Negative)

c: FP (False Positive)

d: TN (True Negative)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{\text{TP}+\text{FN}+\text{FP}+TN}$$

(Source: JunGeol Baek, 2019 1st semester Data mining chapter 3. pp.79.)

1) Metric for Out-of-Distribution Detection

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	a	b
Class=No	c	d

a: TP (True Positive)

b: FN (False Negative)

c: FP (False Positive)

d: TN (True Negative)

$$\text{Precision} = \frac{a}{a+c} = \frac{TP}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{a}{a+b} = \frac{TP}{\text{TP} + FN}$$

(Source: JunGeol Baek, 2019 1st semester Data mining chapter 3. pp.93.)

1) Metric for Out-of-Distribution Detection

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

a: TP (True Positive)

b: FN (False Negative)

c: FP (False Positive)

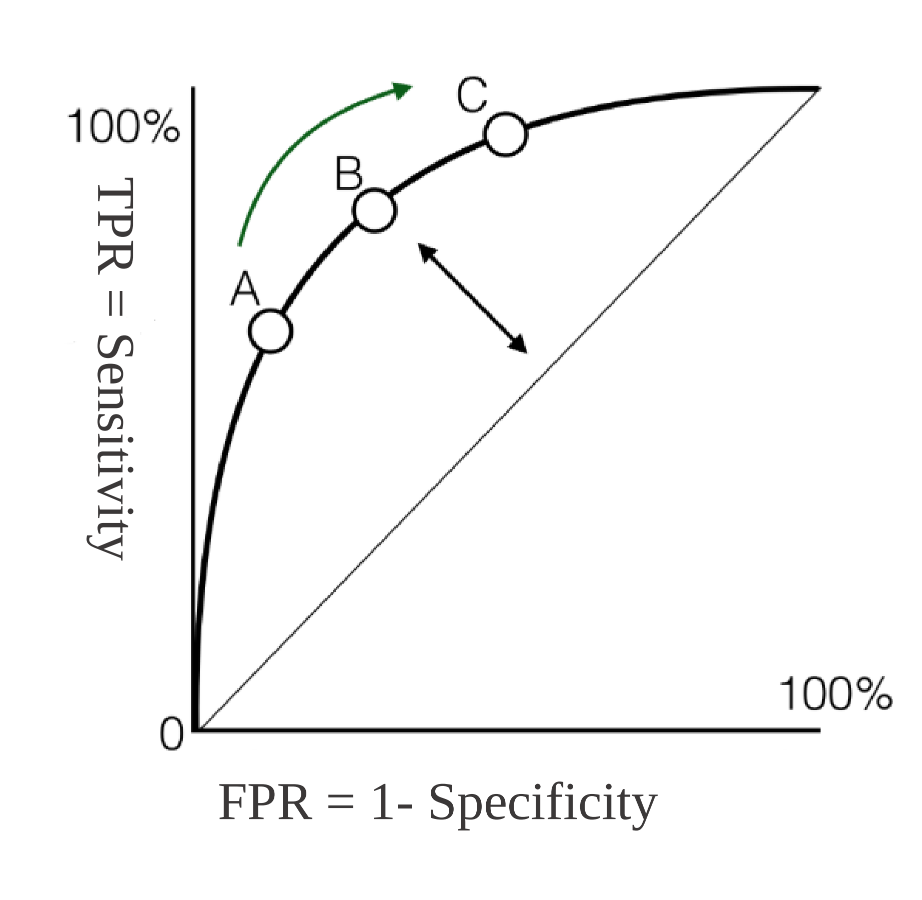
d: TN (True Negative)

$$\mathrm{TPR} = \frac{a}{a+c} = \frac{TP}{\mathrm{TP}+\mathrm{FP}}$$

$$\mathrm{FPR} = \frac{c}{c+d} = \frac{FP}{\mathrm{FP}+TN}$$

(Source: JunGeol Baek, 2019 1st semester Data mining chapter 3. pp.84.)

1) Metric for Out-of-Distribution Detection



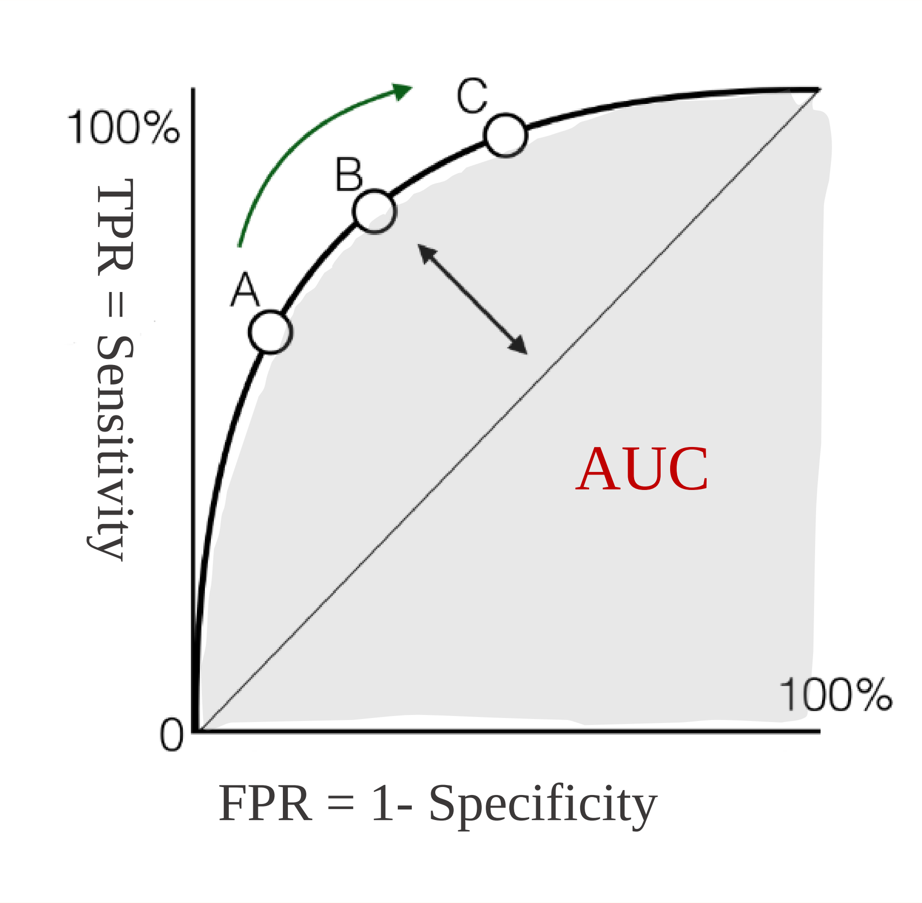
X축 : FPR = FP / (FP + TN)

Y축 : TPR = TP /(TP + FN)

Diagonal line = Random Guessing

Area under ROC curve = AUC

1) Metric for Out-of-Distribution Detection

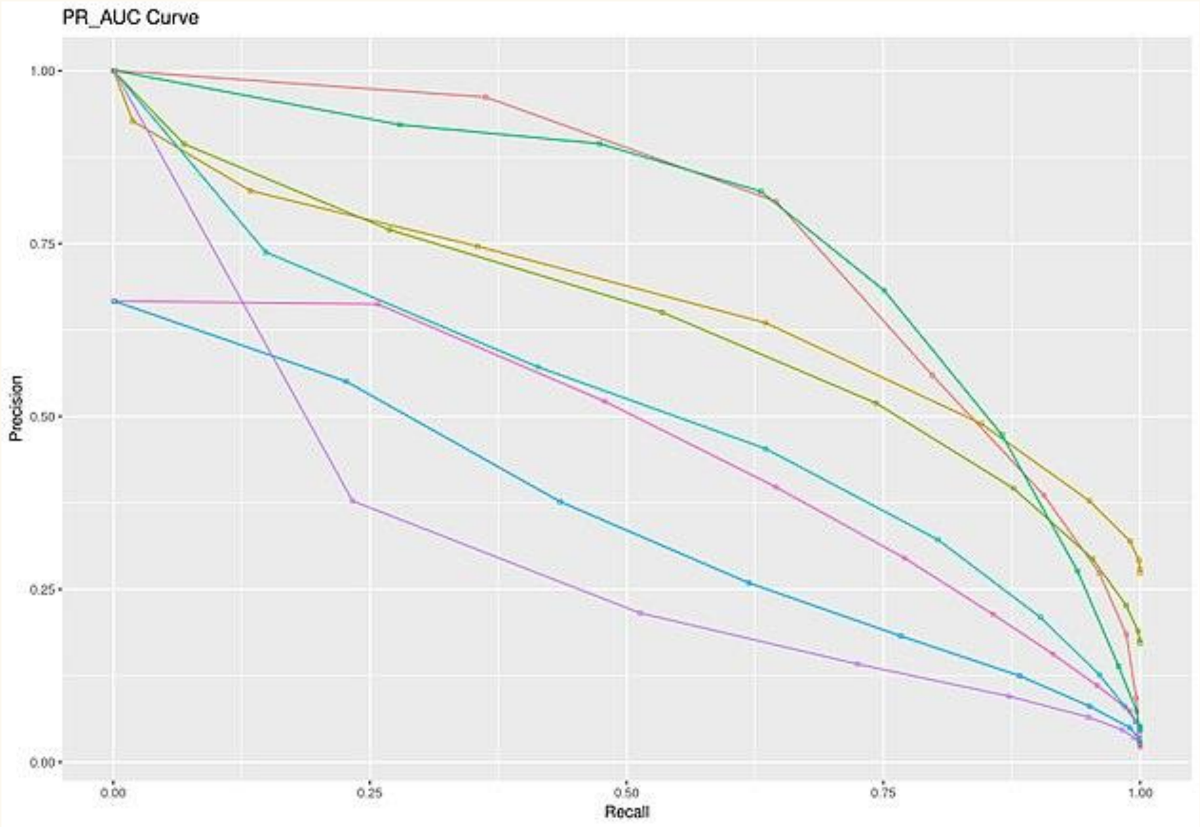


Area under ROC curve = AUC

AUC Range : [0, 1]

100% 맞는 예측 모델일 경우 AUC = 1.

2) Better metric for class-imbalanced data



X축 : Recall : TP / (TP + FN)

Y축 : Precision : TP /(TP + FP)

In Case of imbalanced-Data

⇒ Precision이 FPR에 비해 False Positive를 더 민감하게 잡아낼 수 있다.

⇒ Imbalanced data에서 효과적인 metric!

PART. 3 EDA

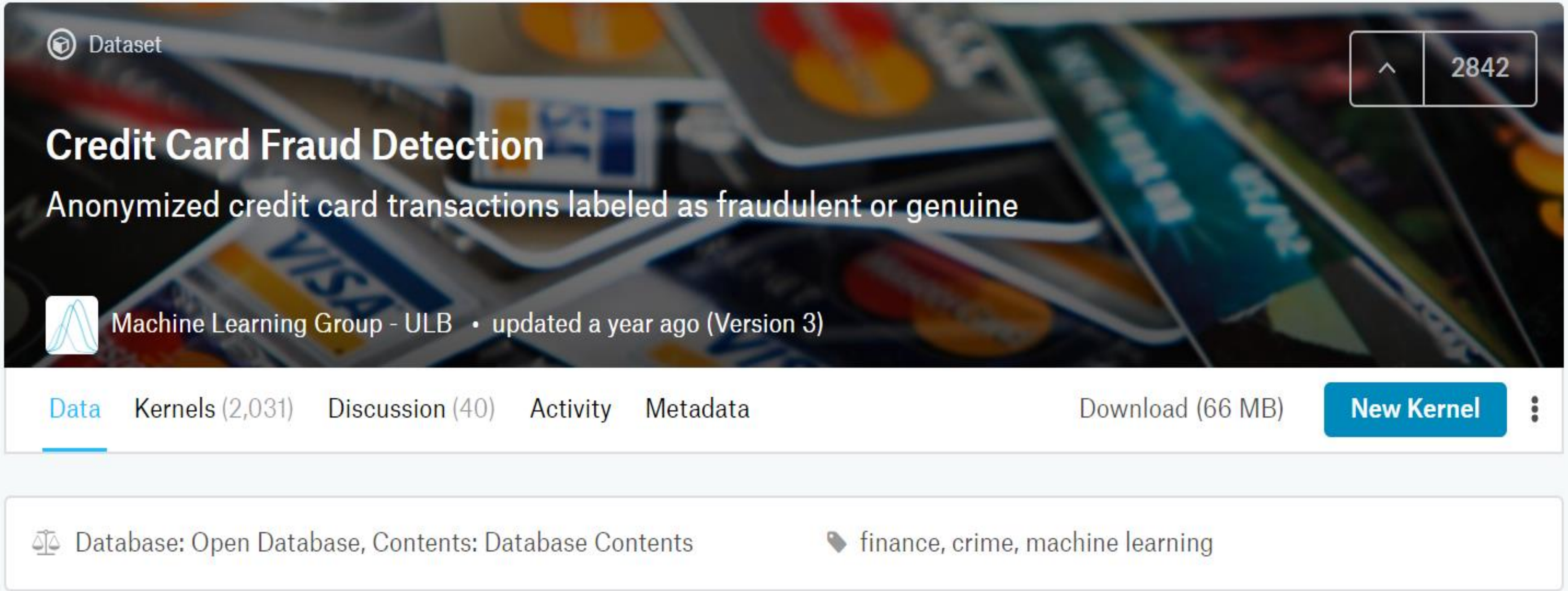
1

Description of Data

2

Non-linear Relations

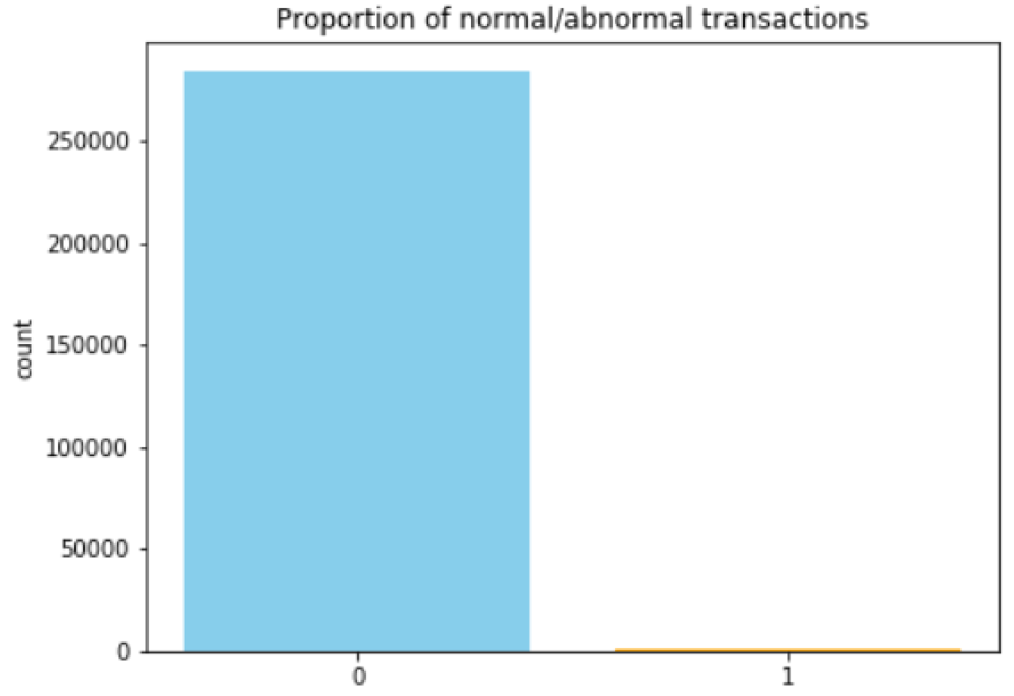
1) Description of Data



1) Description of Data

Credit Card Dataset : (From Kaggle)

- Highly unbalanced data
- 492 frauds out of 284,807 transactions



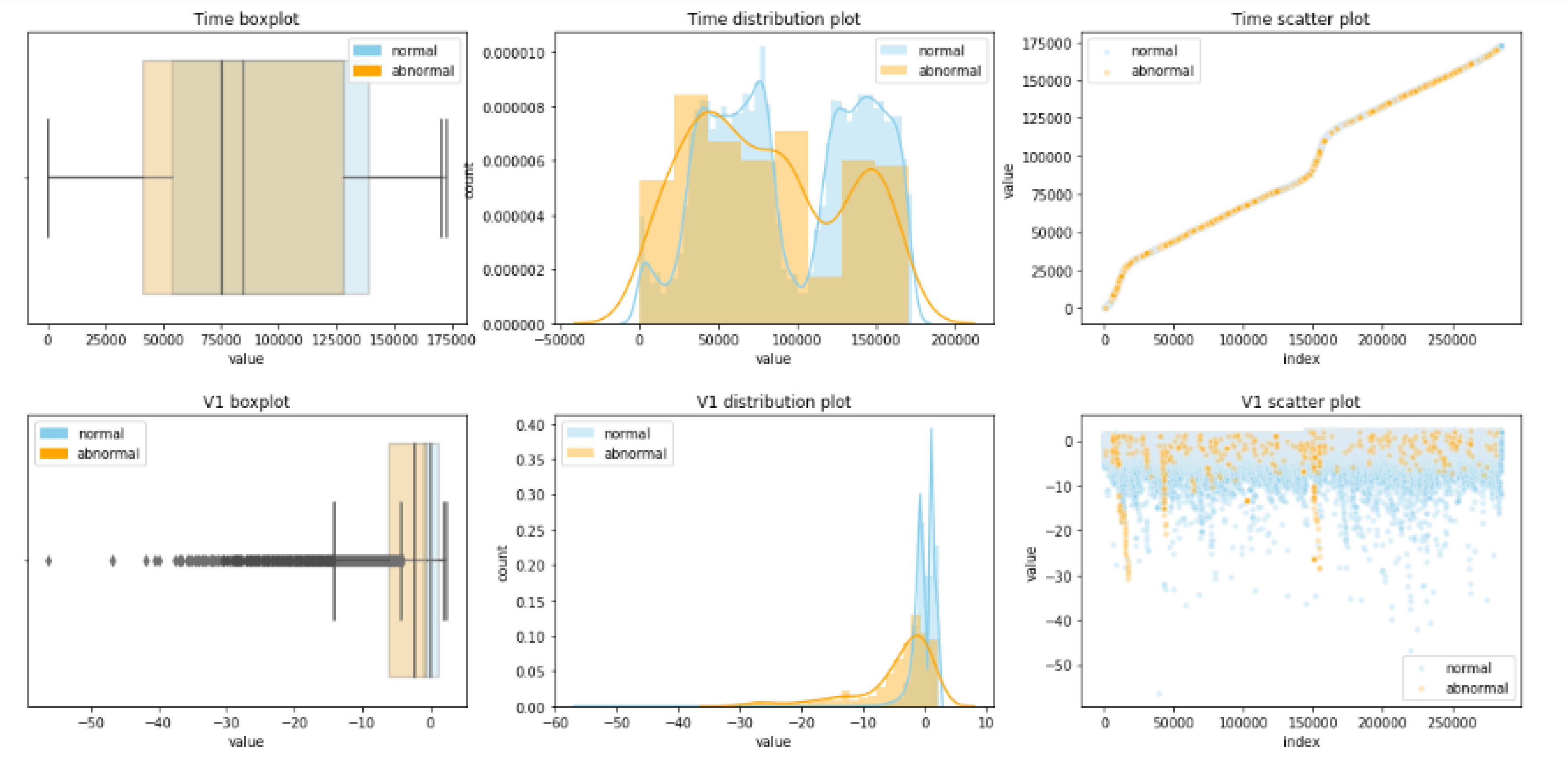

```
0    284315
1       492
Name: Class, dtype: int64
```

1) Description of Data

Input variable :

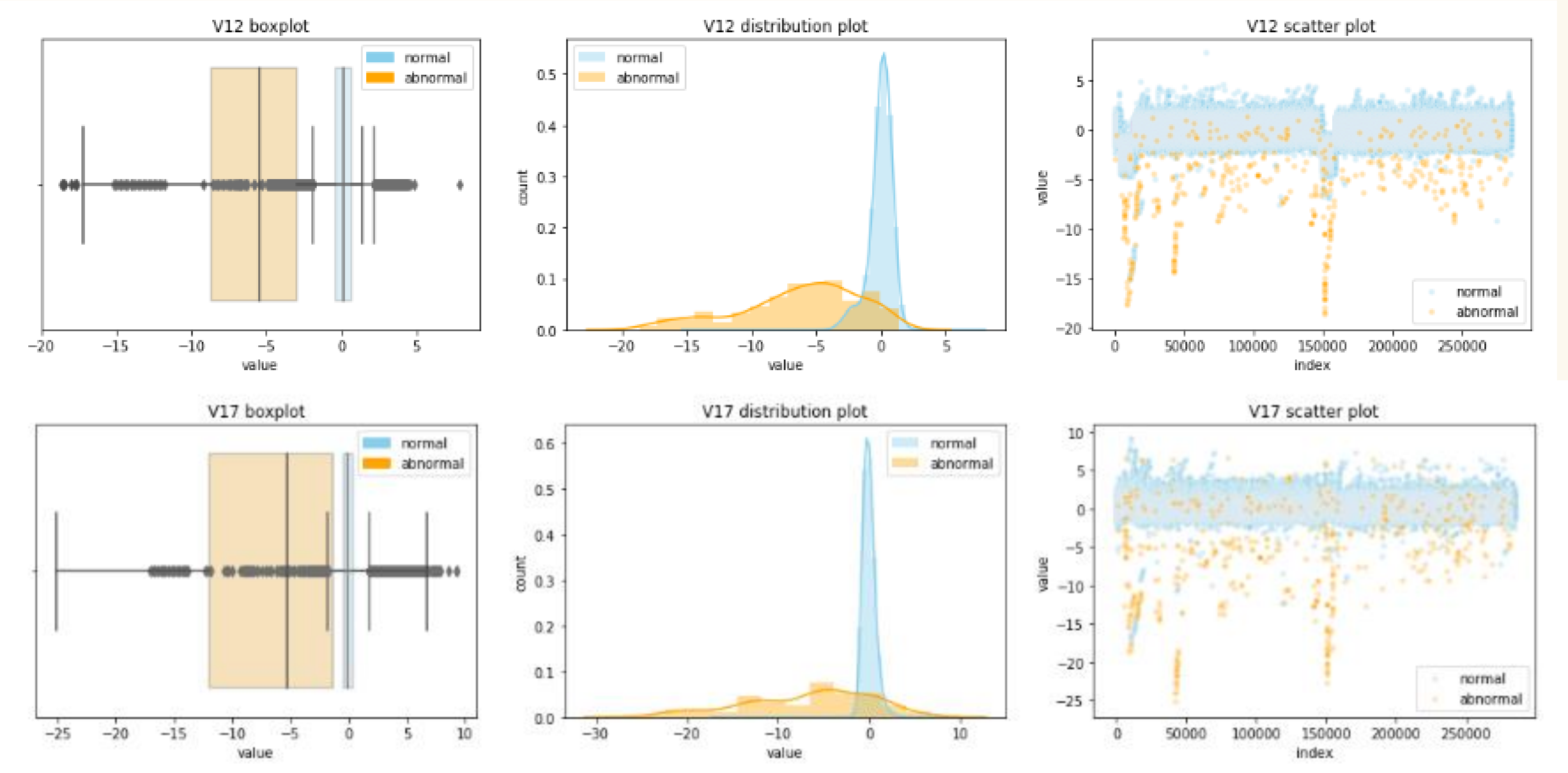
- 28 input variable V, result of PCA transformation.
- Time : seconds (about 48 hours)
- Amount : transaction amount.

1) Description of Data



Plot of Time and V1

1) Description of Data



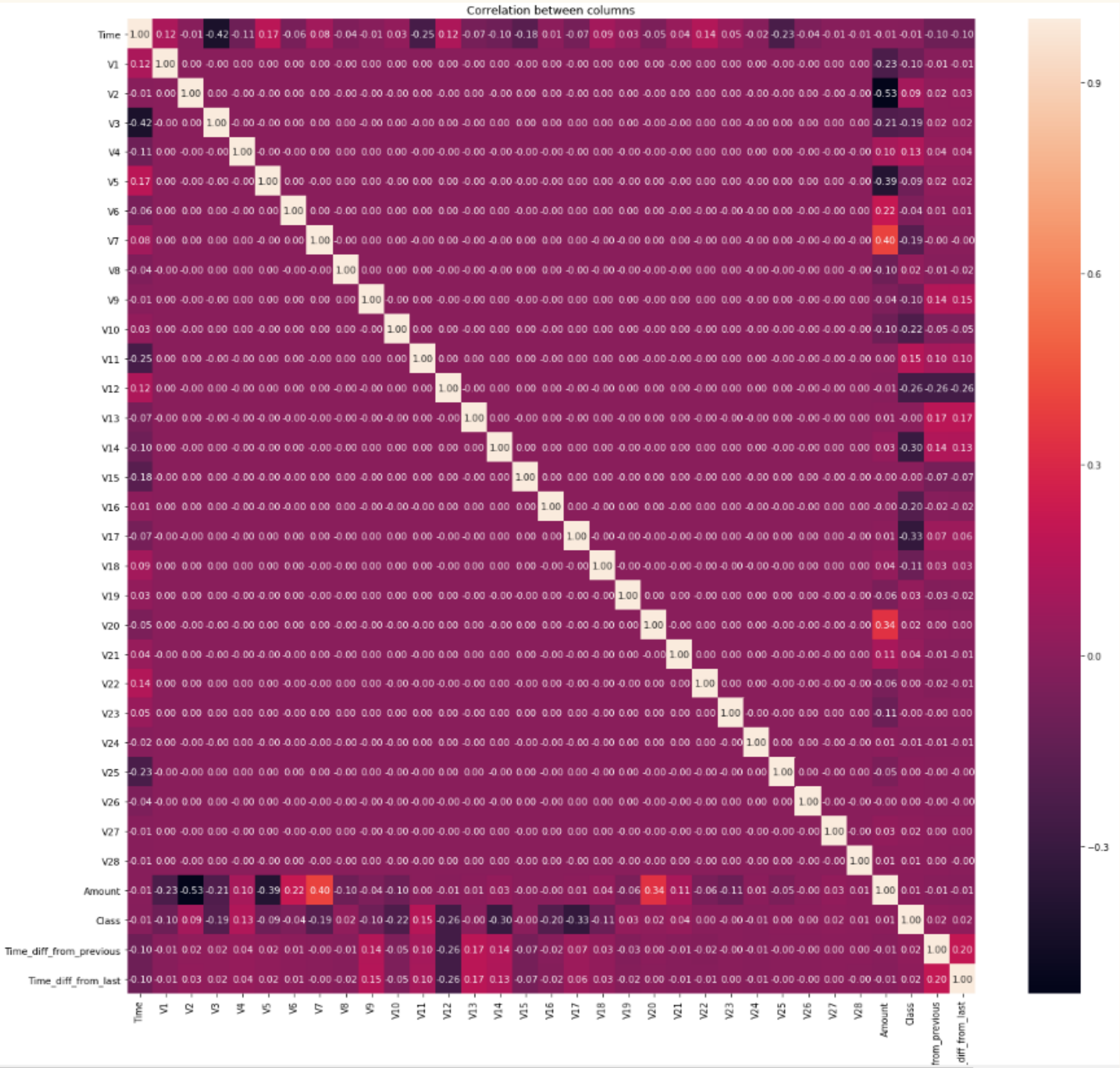
Plot of V12 and V17

2) Linear Relations

Correlation Matrix

- V's : PCA

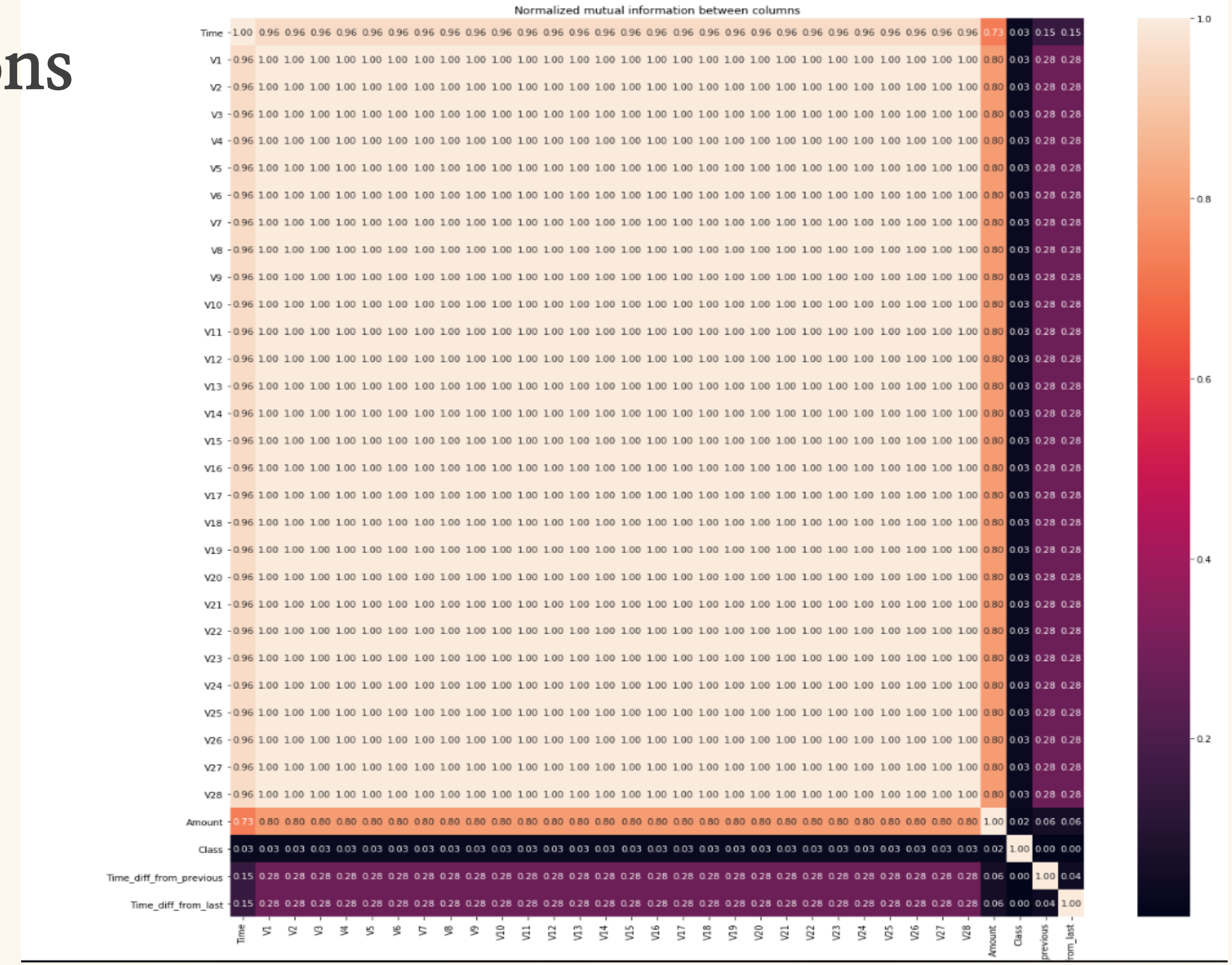
⇒ Linearly independent



2) Nonlinear Relations

Mutual Information Matrix

- All V's are nonlinearly dependent.
- All Vs and amount are nonlinearly dependent.



PART. 4 Methods to Use

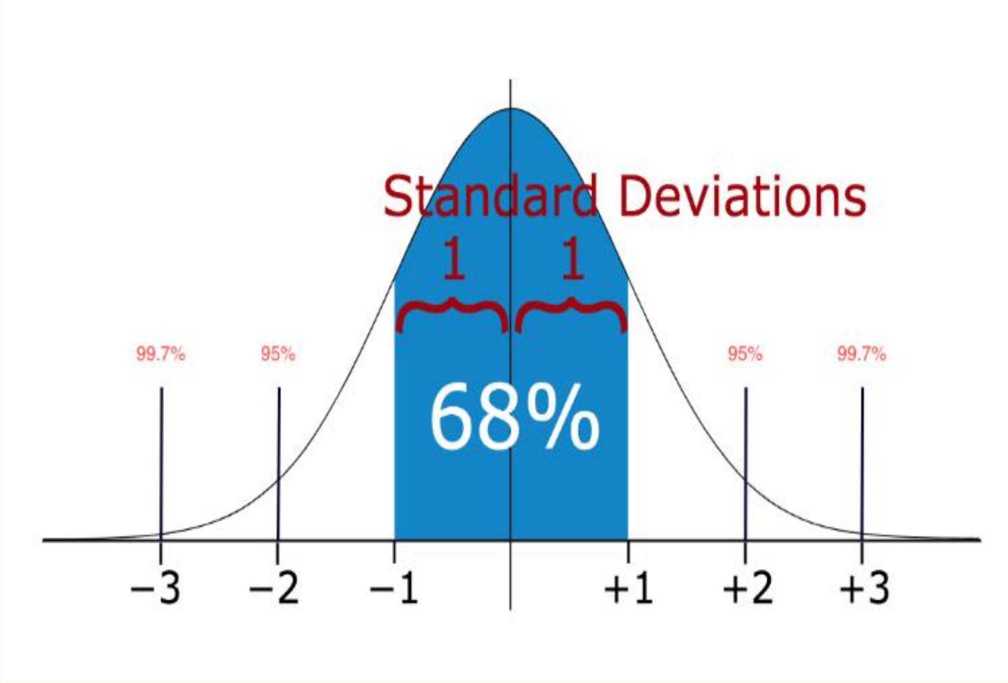
1

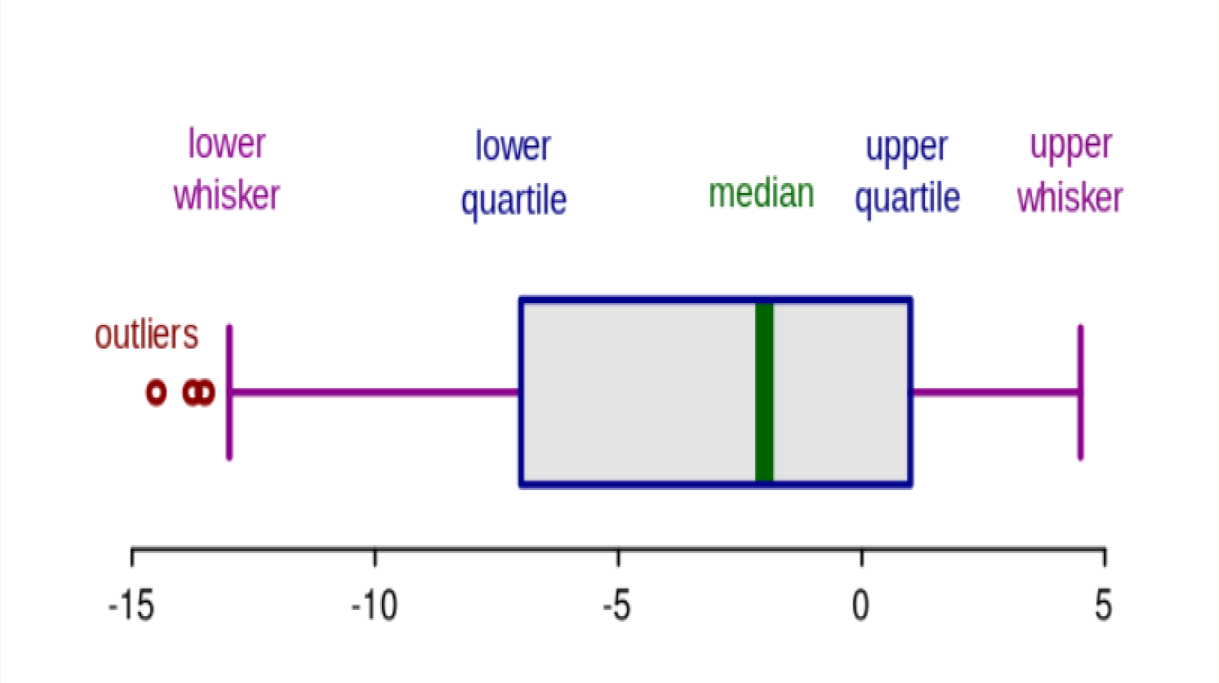
Statistical Methods

2

VAE Model

1) Statistical Methods

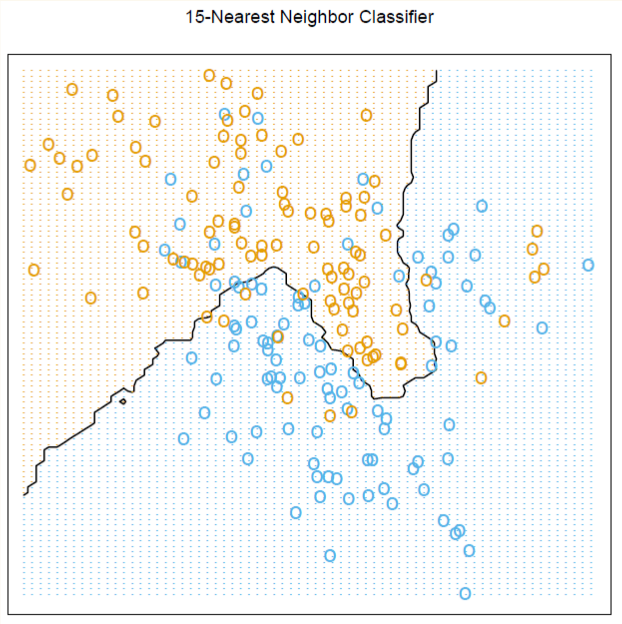




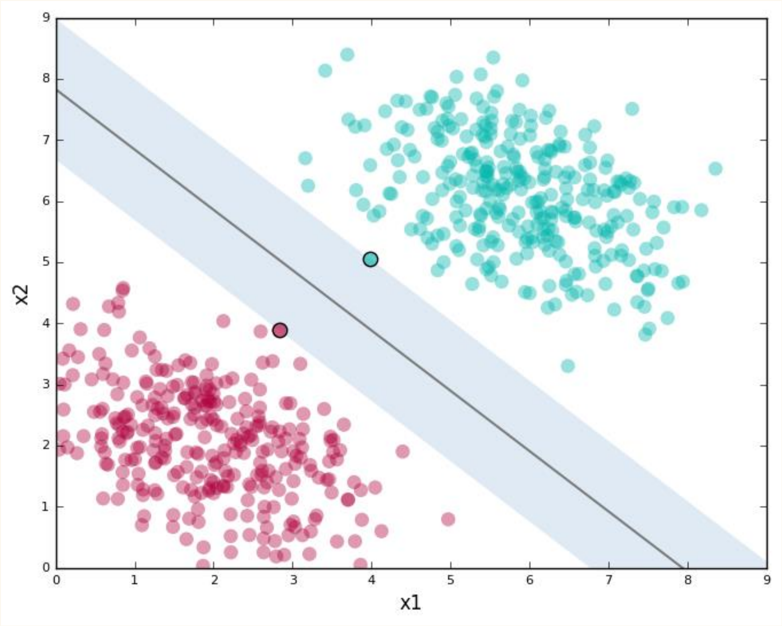
Standard Deviations

Boxplots

1) Statistical Methods



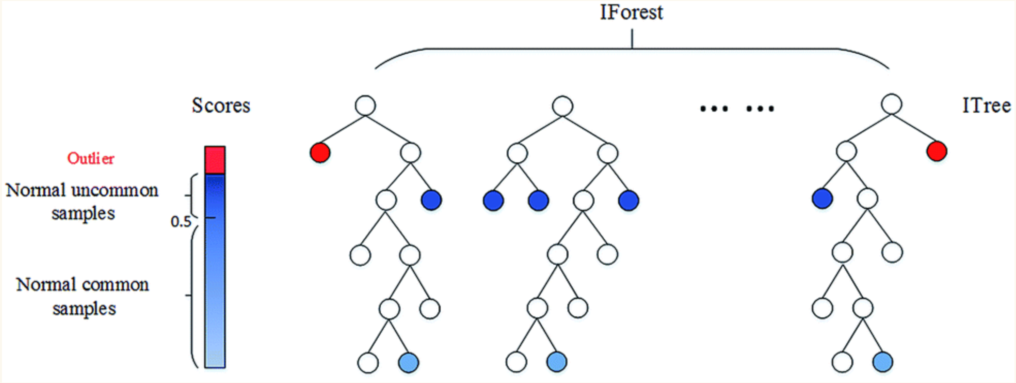
K-Nearest Neighbors



Support Vector Machine

https://en.wikipedia.org/wiki/Support-vector_machine

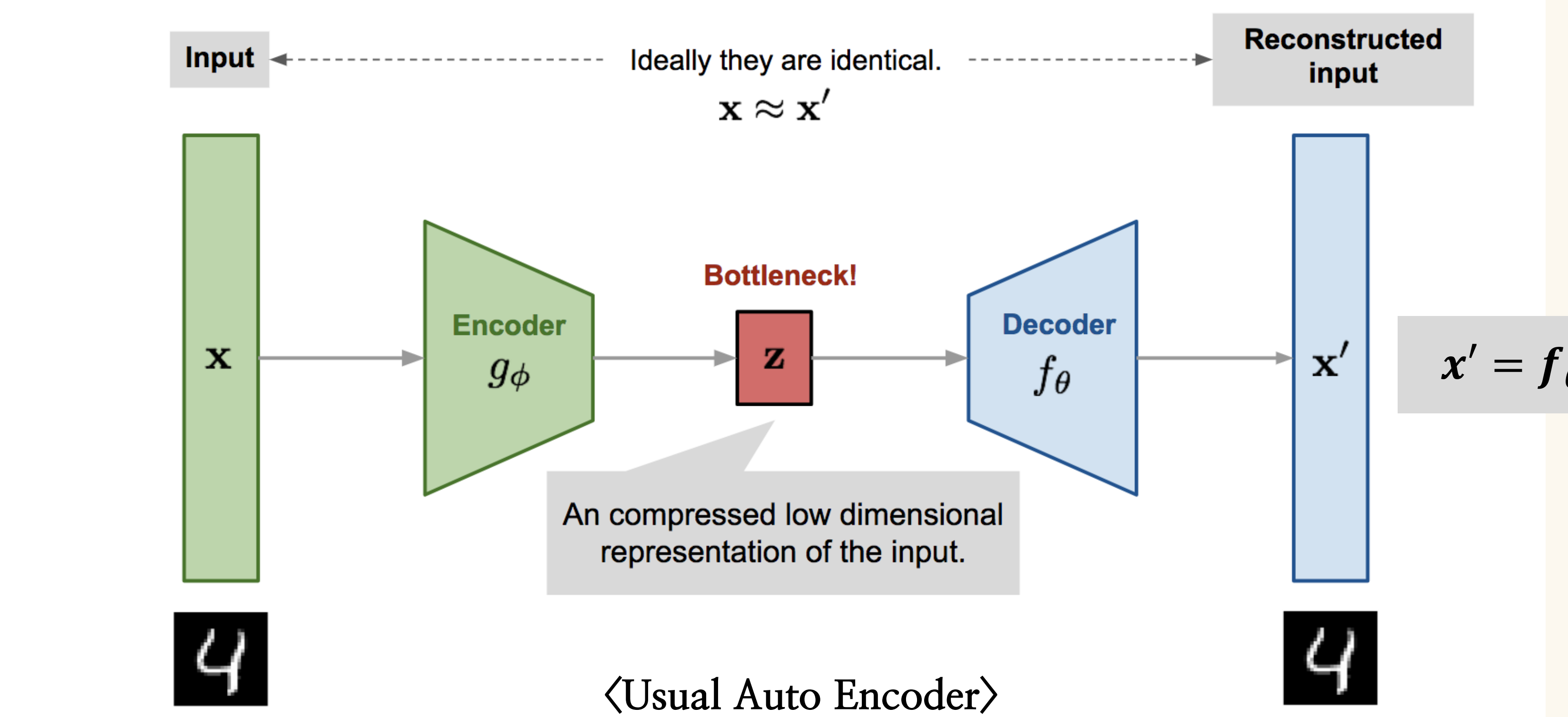
1) Statistical Methods



Isolation Forest

https://donghwa-kim.github.io/iforest.html

2) VAE Model



⟨Usual Auto Encoder⟩

$$x' = f_\theta(g_\phi(x))$$

MSE loss function : $L_{AE}(\phi,\theta) = \frac{1}{n}\sum_{i=1}^{n} (x^{(i)} - f_\theta\left(g_\phi(x^{(i)})\right))^2$

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

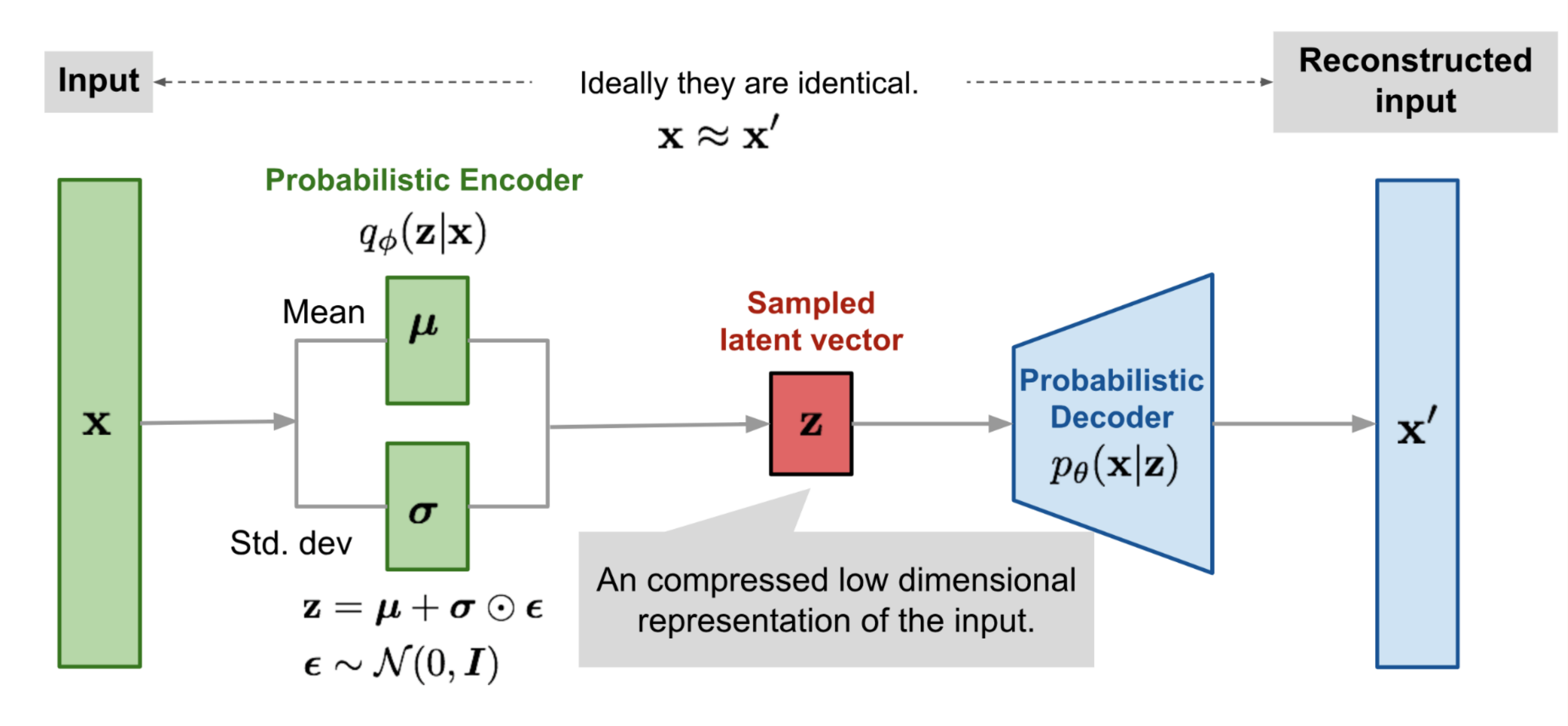
2) VAE Model

Autoencoder 제약의 효과?

단순히 입력을 바로 출력으로 복사하지 못하도록 방지

데이터를 효율적으로 표현하는 방법을 학습하도록 제어

2) VAE Model



<Decoder>

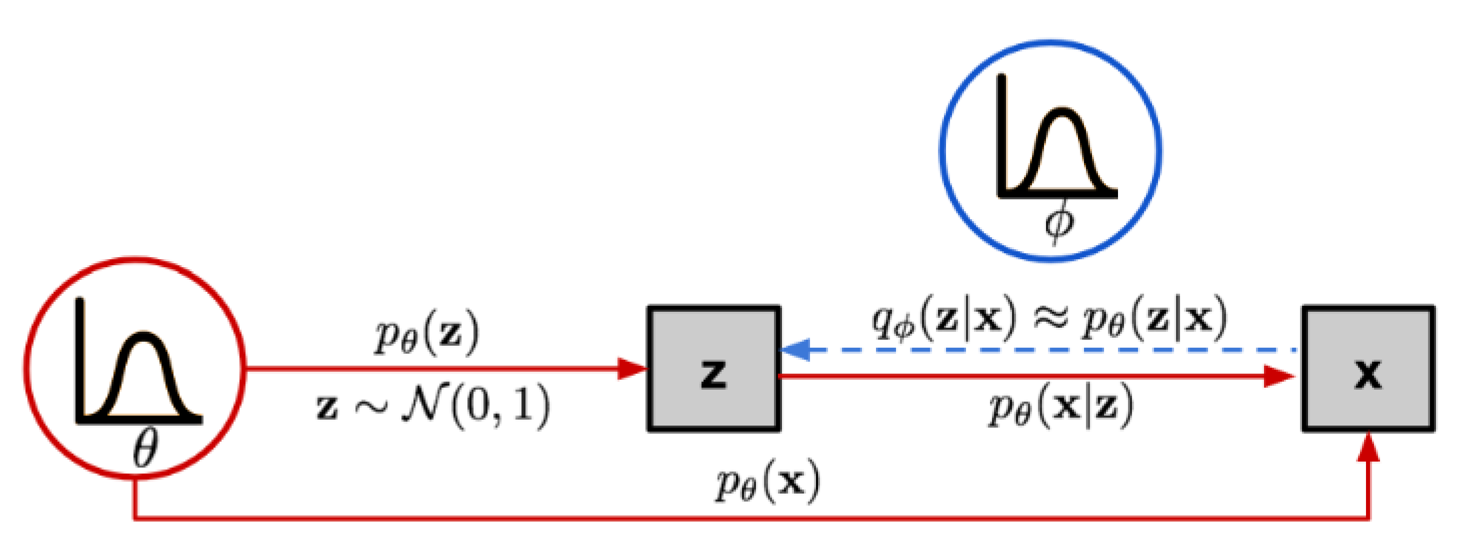
θ^* : true parameter

1. $p_{\theta^*}(\mathbf{z})$에서 $\mathbf{z}^{(\mathrm{i})}$ 추출.
2. $p_{\theta^*}(\mathbf{x}|\mathbf{z}^{(\mathrm{i})})$에서 $\mathbf{x}^{(\mathrm{i})}$생성

Optimal parameter : $\theta^* = \arg\max_\theta \sum_{i=1}^{n} \log \boxed{p_\theta\left(\mathbf{x}^{(i)}\right)}$ → Intractable !!

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

2) VAE Model



z : latent variable

$p_\theta(\mathbf{z})$: prior

$p_\theta(\mathbf{x}|\mathbf{z})$: Likelihood

$p_\theta(\mathbf{z}|\mathbf{x})$: posterior

알려져 있는 분포 활
용

$p_\theta(\mathbf{z})$를 추론하고자 $p_\theta(\mathbf{z}|\mathbf{x})$를 이용하고, $p_\theta(\mathbf{z}|\mathbf{x})$를 추론하고자 $q_\phi(\mathbf{z}|\mathbf{x})$를 활용한다.

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

2) VAE Model

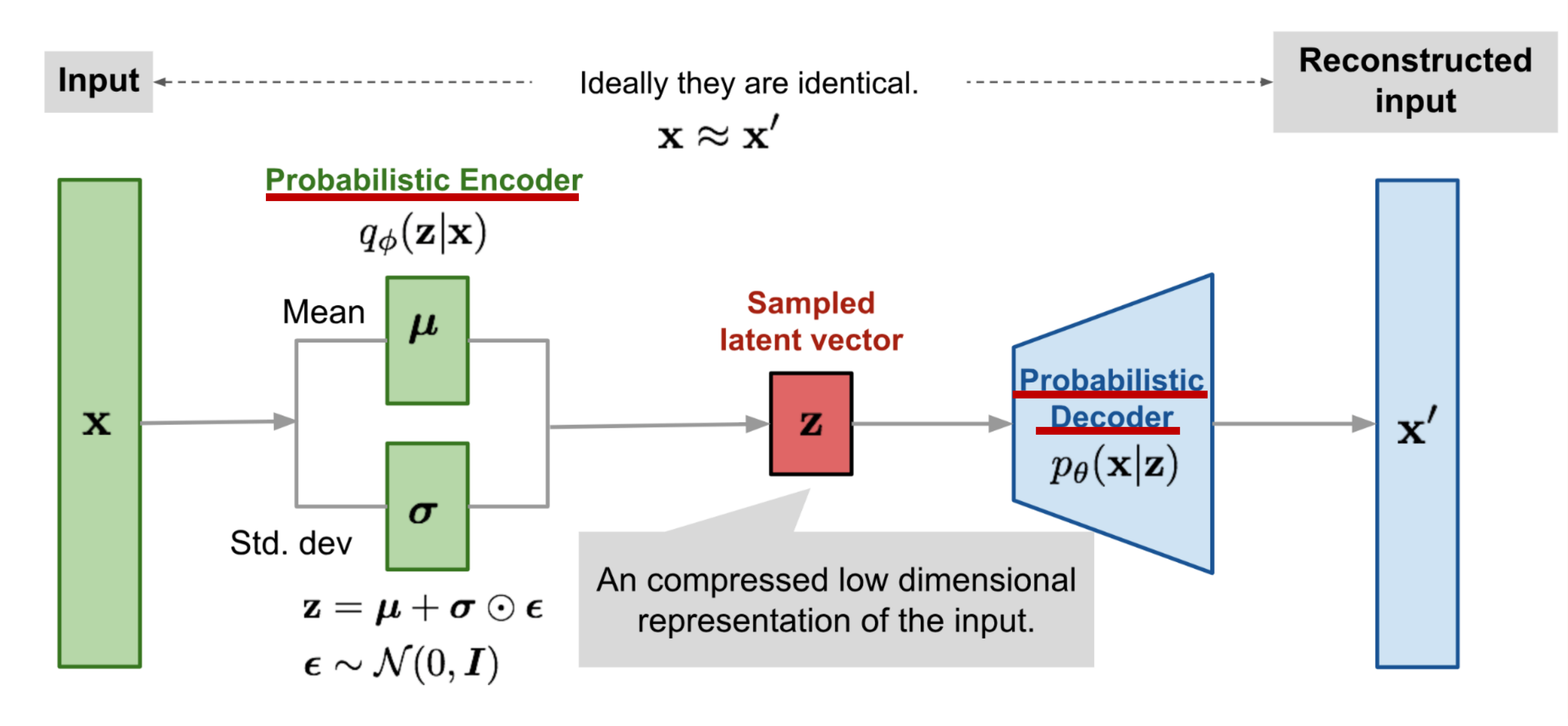
$p_\theta(\mathbf{z})$를 추론하고자 $p_\theta(\mathbf{z}|\mathbf{x})$를 이용하고, $p_\theta(\mathbf{z}|\mathbf{x})$를 추론하고자 $q_\phi(\mathbf{z}|\mathbf{x})$를 활용한다.

Variational Inference

1. $p_\theta(\mathbf{x})$를 계산하기 힘든 경우
2. $p_\theta(\mathbf{z}), p_\theta(\mathbf{x}|\mathbf{z})$ 를 더 복잡하게 모델링하고 싶은 경우

https://ratsgo.github.io/generative%20model/2017/12/19/vi/

2) VAE Model



Loss function :

$$
\begin{aligned}
L_{\mathrm{VAE}}(\theta, \phi) &= -\log p_\theta(\mathbf{x}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x})) \\
&= -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} p_\theta(\mathbf{x}|\mathbf{z}) + D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) \\
\theta^*, \phi^* &= \arg\min_{\theta, \phi} L_{\mathrm{VAE}}
\end{aligned}
$$

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

PART. 5 Results – Statistcal Methods

1

K-Nearest Neighbors

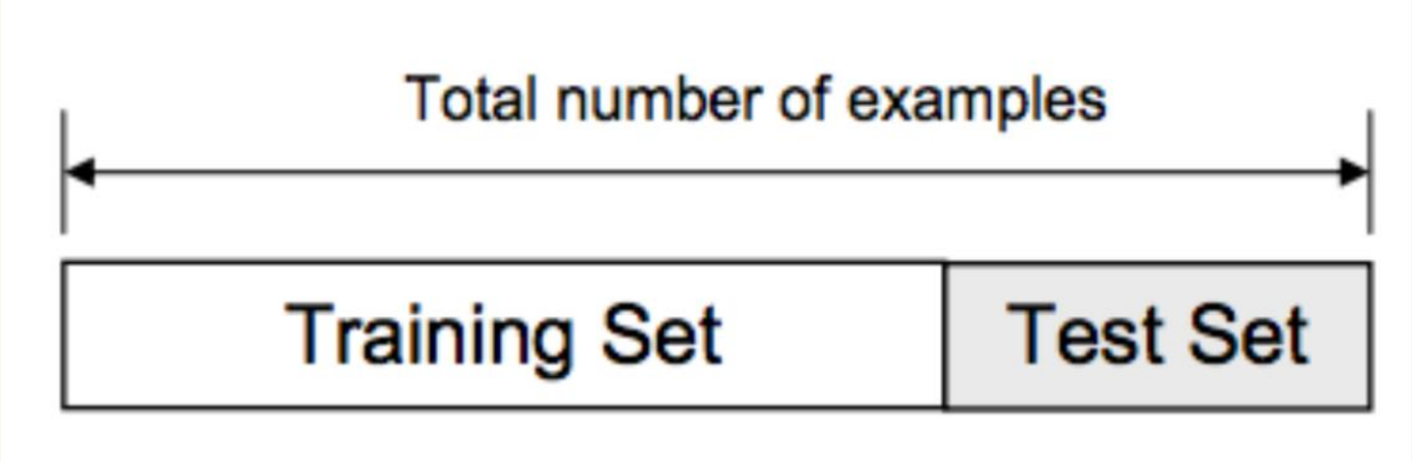
2

Support Vector Machine

3

Isolation Forest

1) K-Nearest Neighbors



Training : Test = 7 : 3

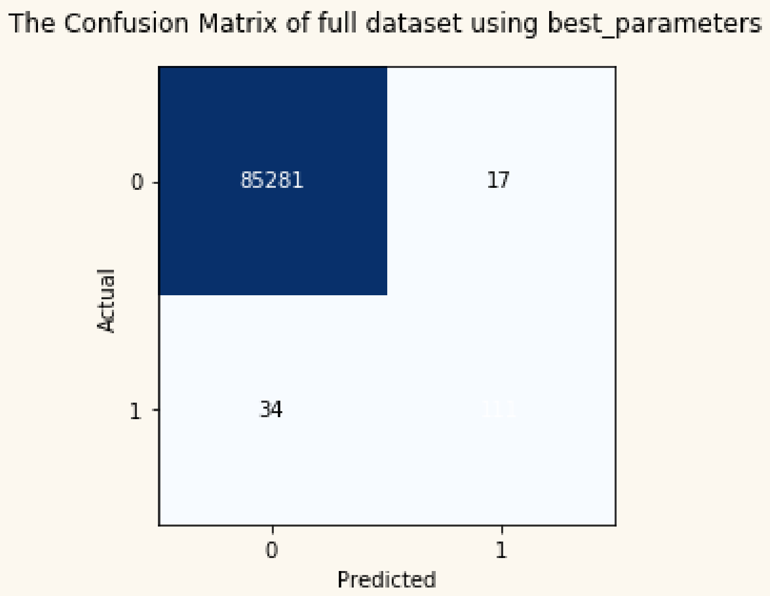
The number of testing data is about 85,000

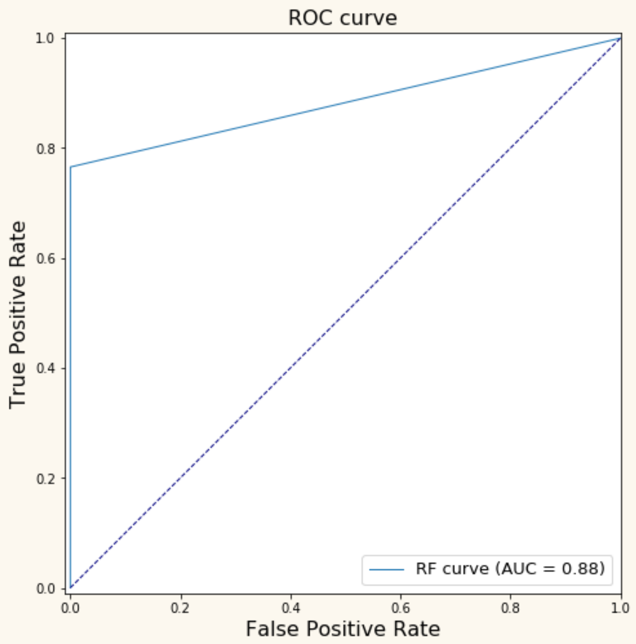
1) K-Nearest Neighbors

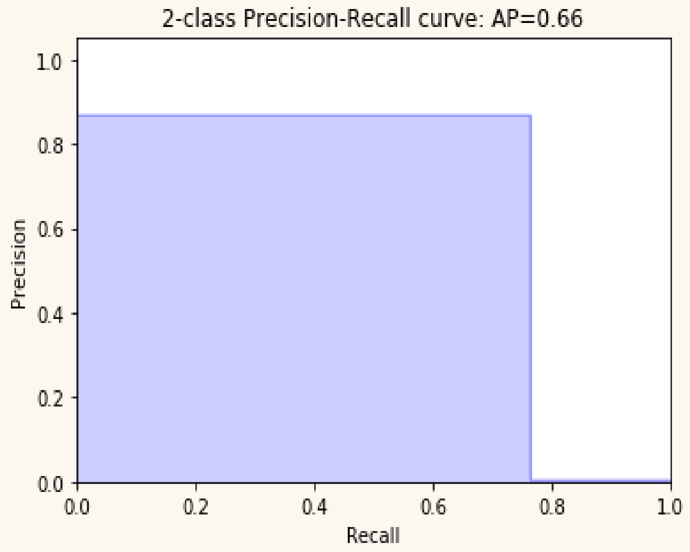
```
K-Nearest Neighbours
Confusion Matrix
tn = 85281 fp = 17
fn = 34 tp = 111
Scores
Accuracy --> 0.999403110845827
Precison --> 0.8671875
Recall --> 0.7655172413793103
F1 --> 0.8131868131868132
*******************************************************************
Area under the curve : 0.882659
Average precision-recall score RF: 0.6642449088614026
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     85298
           1       0.87      0.77      0.81       145

    accuracy                           1.00     85443
   macro avg       0.93      0.88      0.91     85443
weighted avg       1.00      1.00      1.00     85443

*******************************************************************
```

Most of the data is normal -> Accuracy is not important, though it is 99%

1) K-Nearest Neighbors







Testing Sets	P = Normal	P = Outlier
Actual = Normal	85281	17
Actual = Outlier	34	111

2) Support Vector Machine

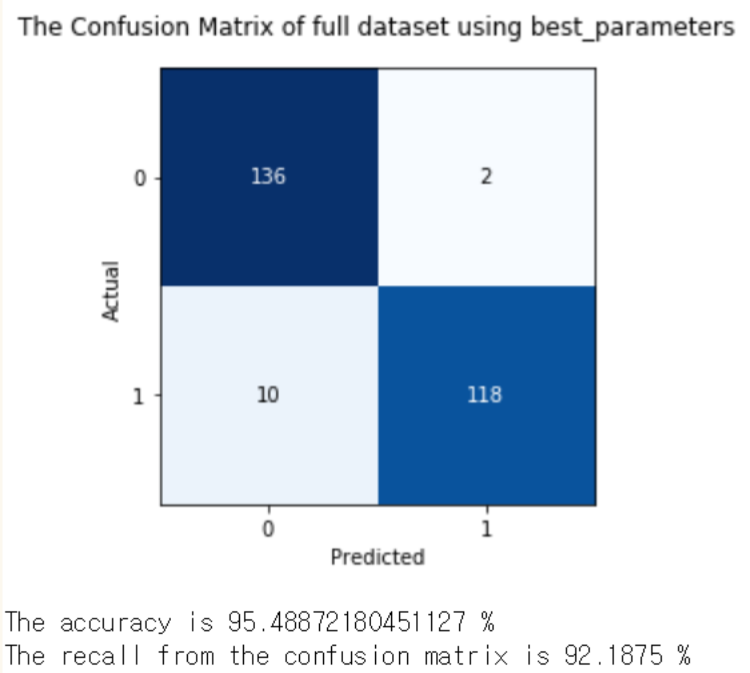
Time Consuming with Full data

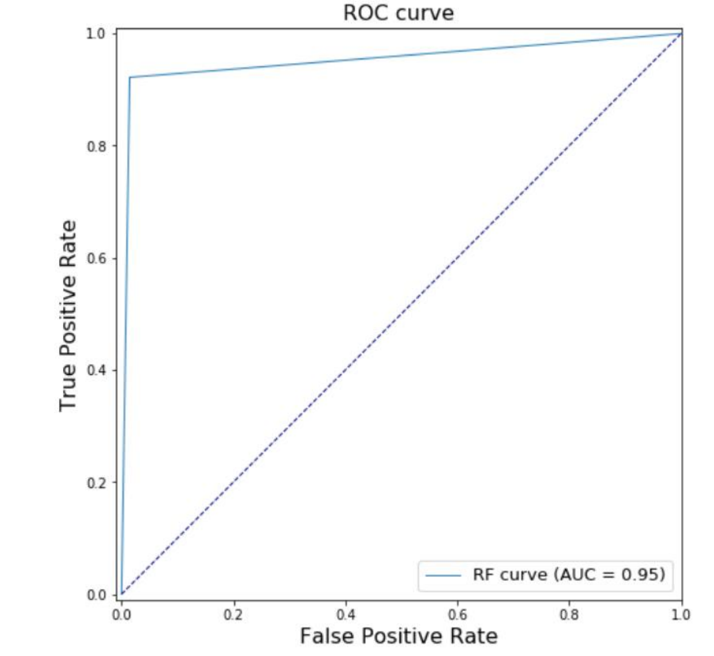
Totally Imbalanced

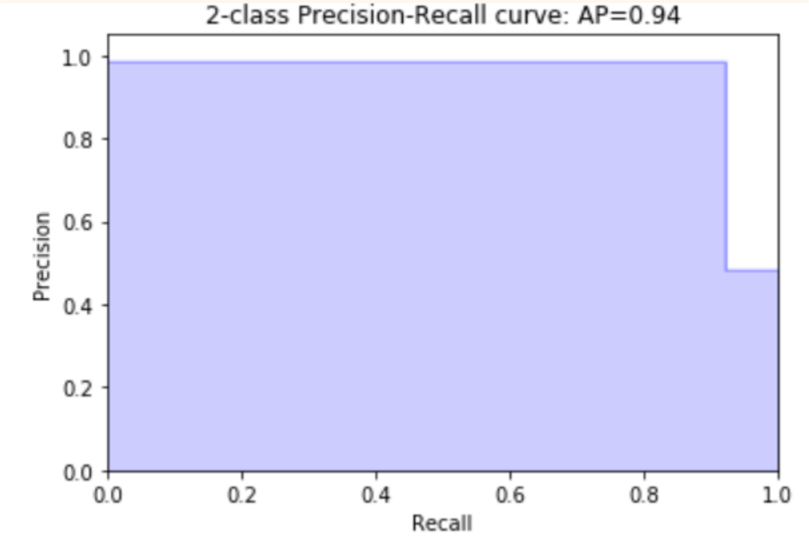
→

Small, Balanced Sample

2) Support Vector Machine







Testing Sets	P = Normal	P = Outlier
Actual = Normal	136	2
Actual = Outlier	10	118

3) Isolation Forest

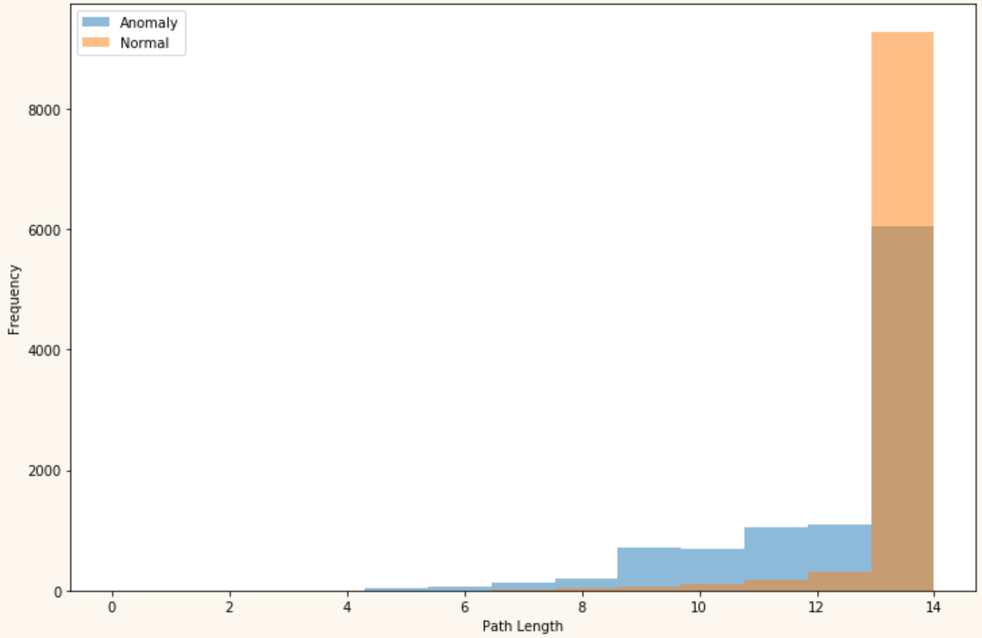
```
In [6]: df=pd.read_csv("../data/creditcard.csv")
        y_true=df['Class']
        df_data=df.drop('Class',1)
```

```
In [7]: # create the forest
        sampleSize=10000
        ifor=iForest(df_data.sample(100000),10,sampleSize) ##Forest of 10 trees
```

Repeated Sampling of obs=10,000 for each tree

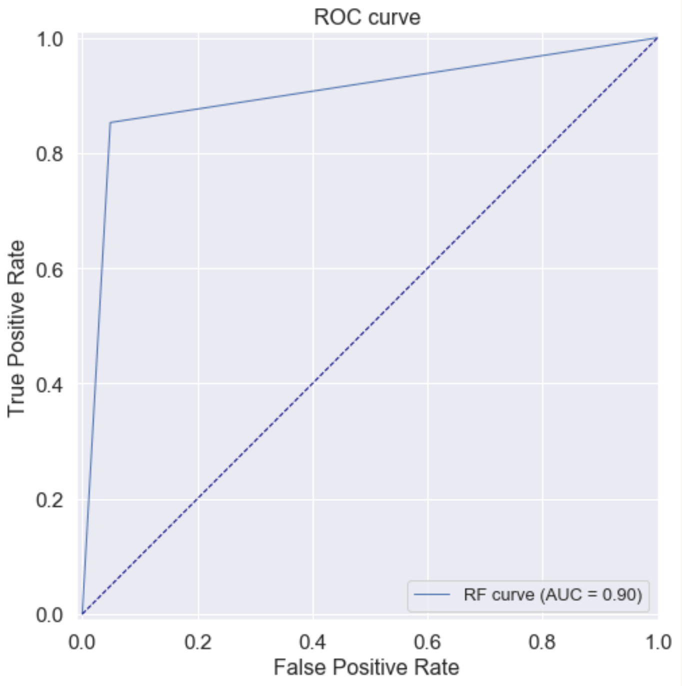
Forest of 10 trees

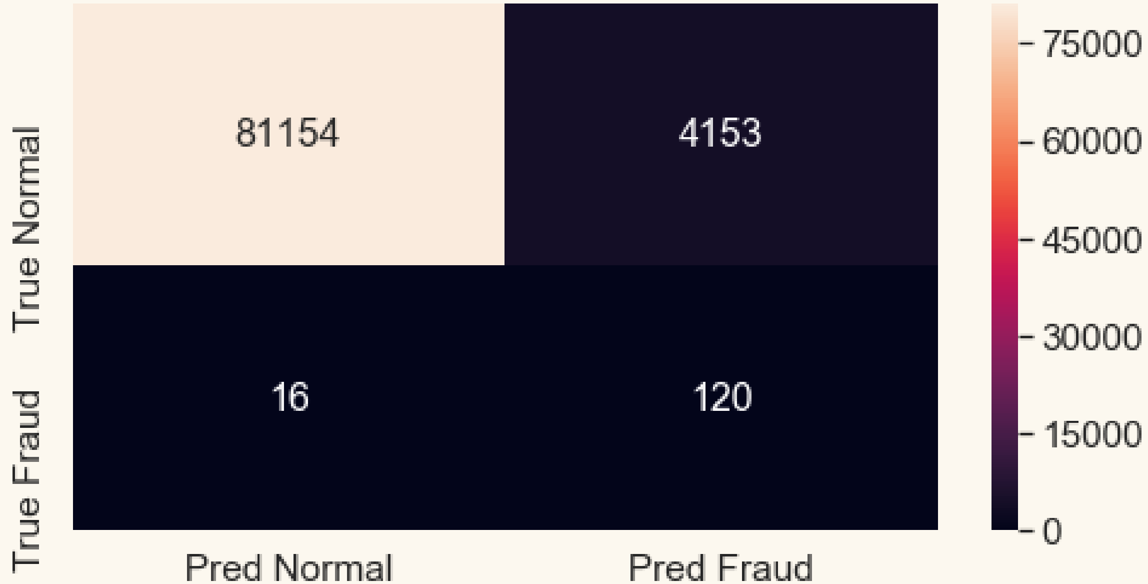
3) Isolation Forest



Paths of the Abnormal Data are shorter!

3) Isolation Forest





Testing Sets	P = Normal	P = Outlier
Actual = Normal	81154	4153
Actual = Outlier	16	120

PART. 6 Results – VAE Models

1

Hdim = 10

Zdim = 2

2

Hdim = 15, 7

Zdim = 2

3

Hdim = 17, 9

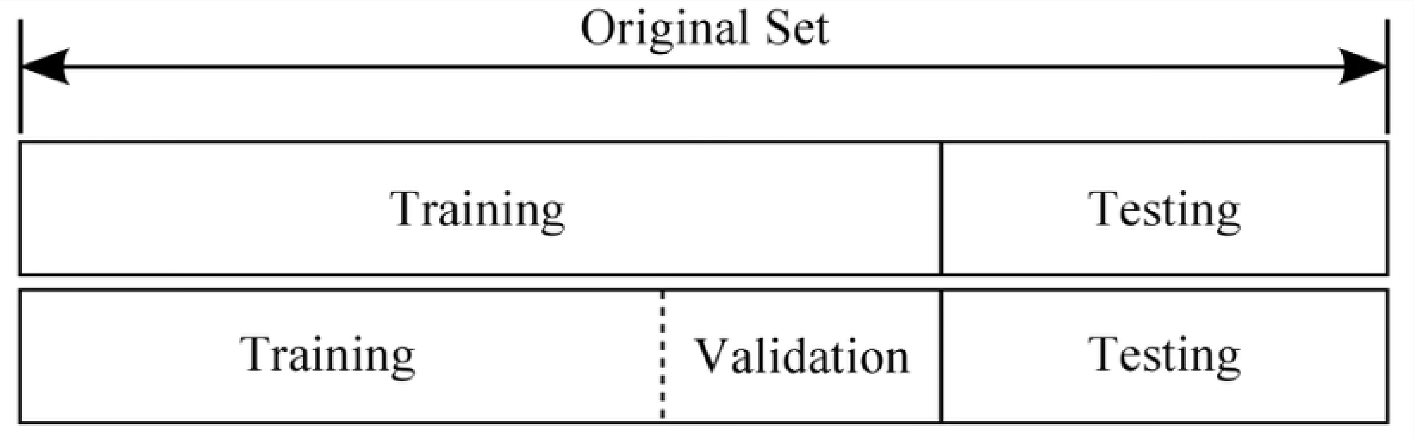
Zdim = 5

4

Hdim = 20, 15

Zdim = 10

1) Hdim = 10, Zdim = 2

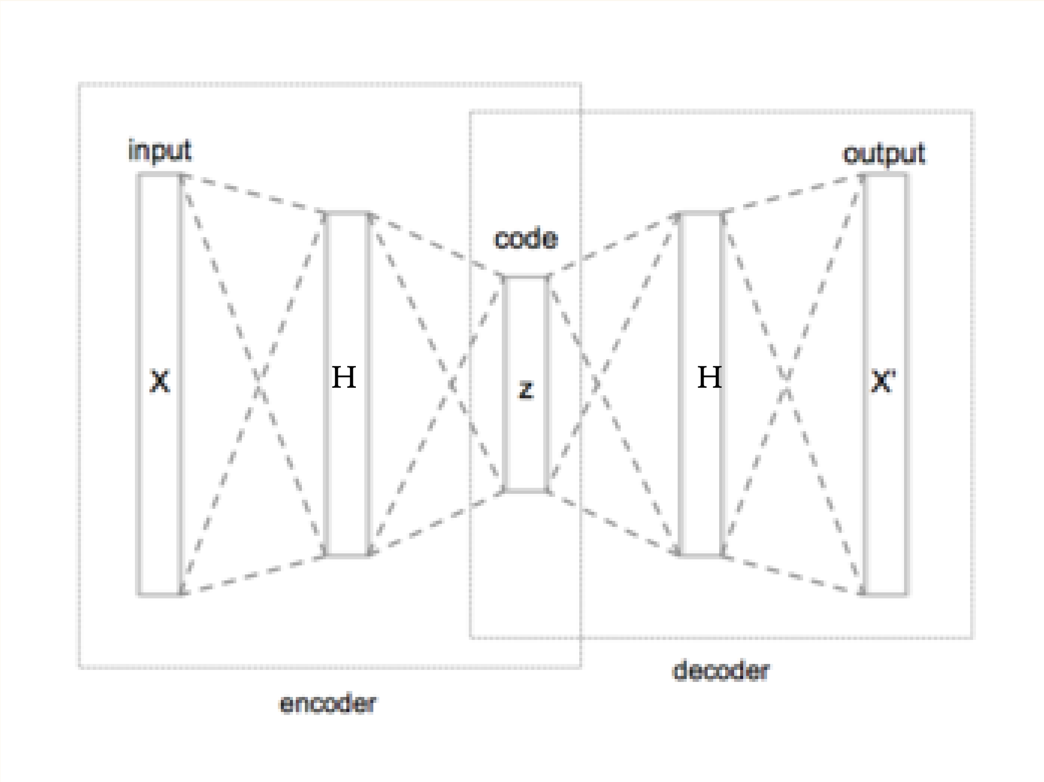


X_Training = 170,589

X_Validation = 56,863

X_Test = 56,863 + 492 = 57,355

1) Hdim = 10, Zdim = 2

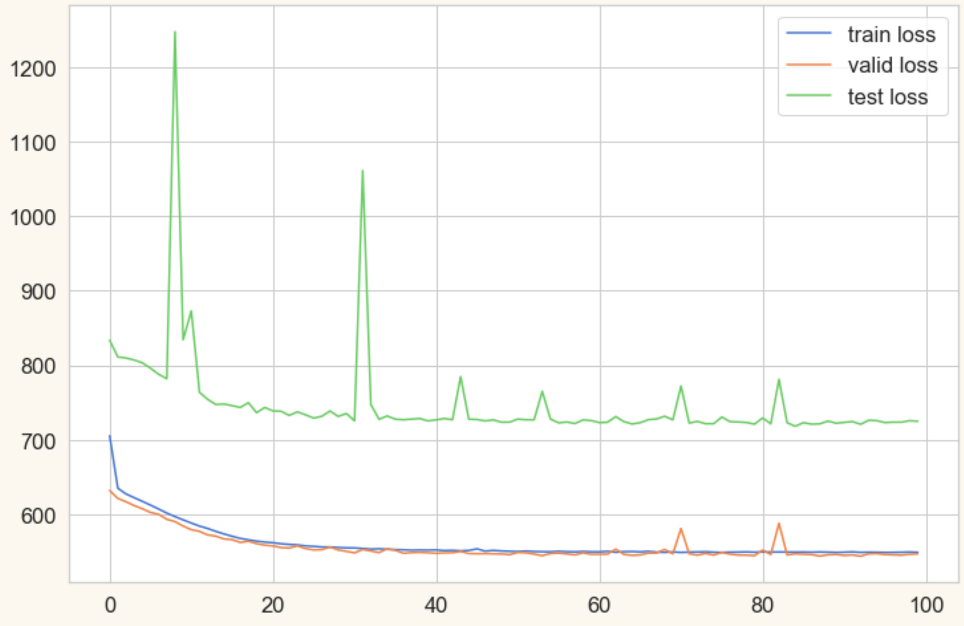


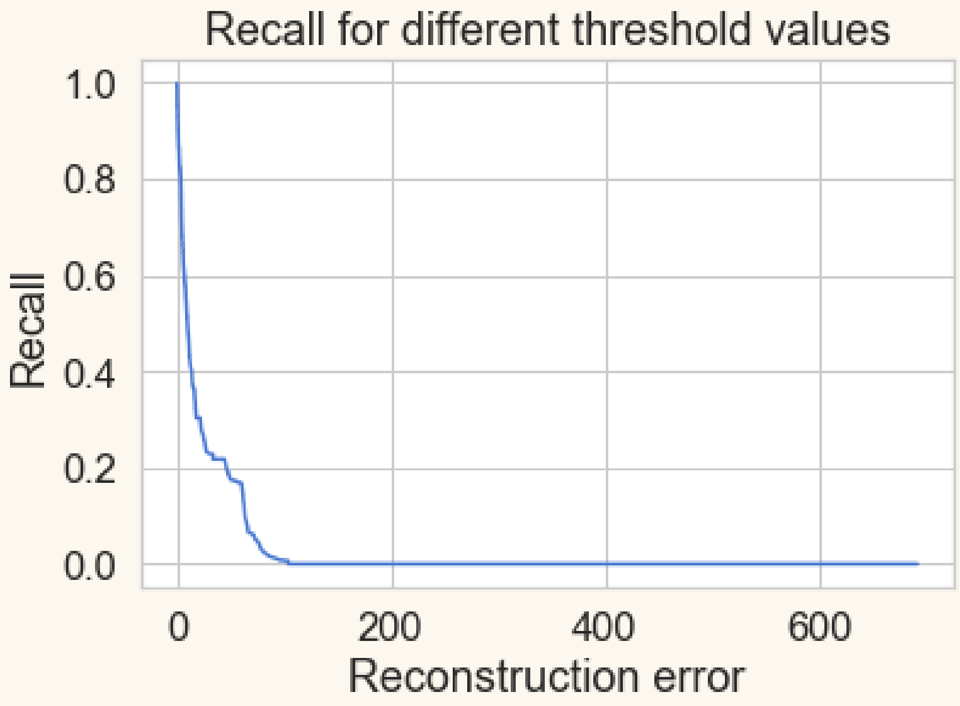
Input Dim = 29

H Dim = 10

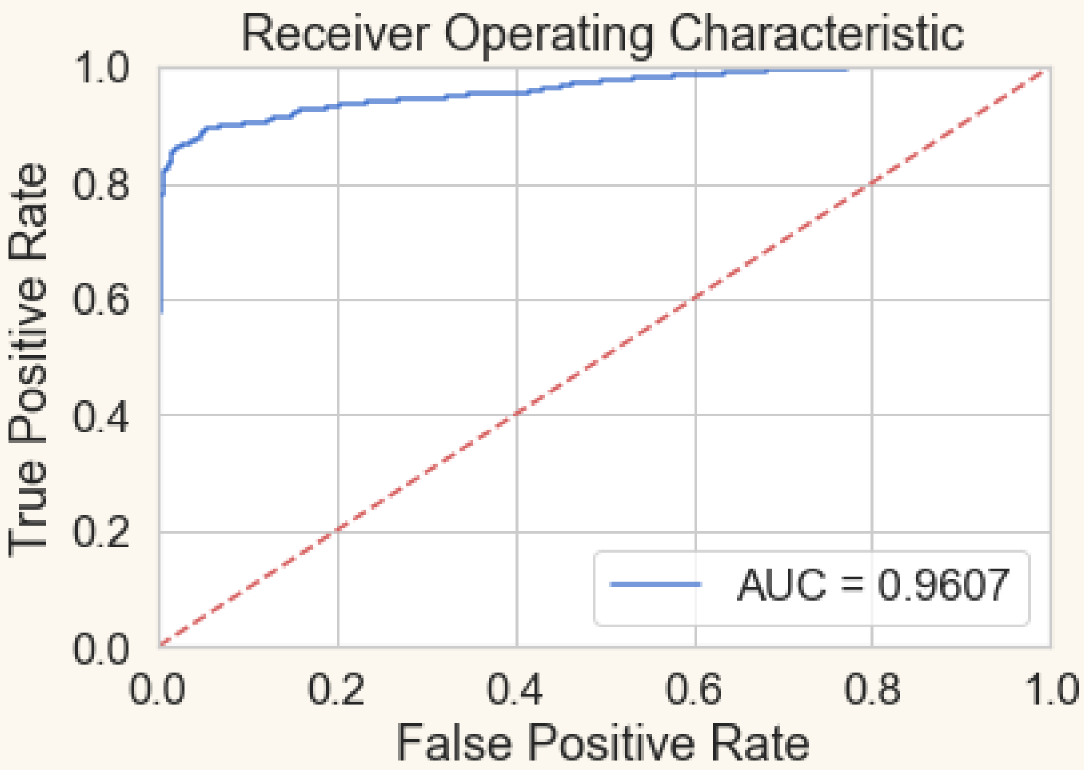
Z Dim = 2

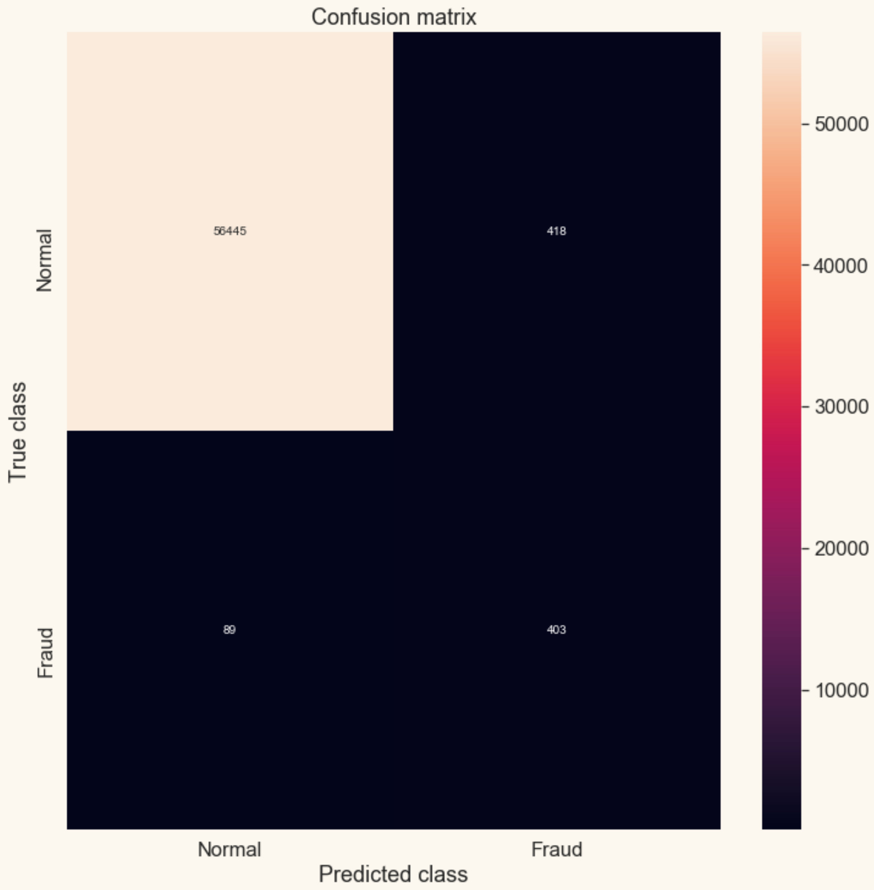
1) Hdim = 10, Zdim = 2



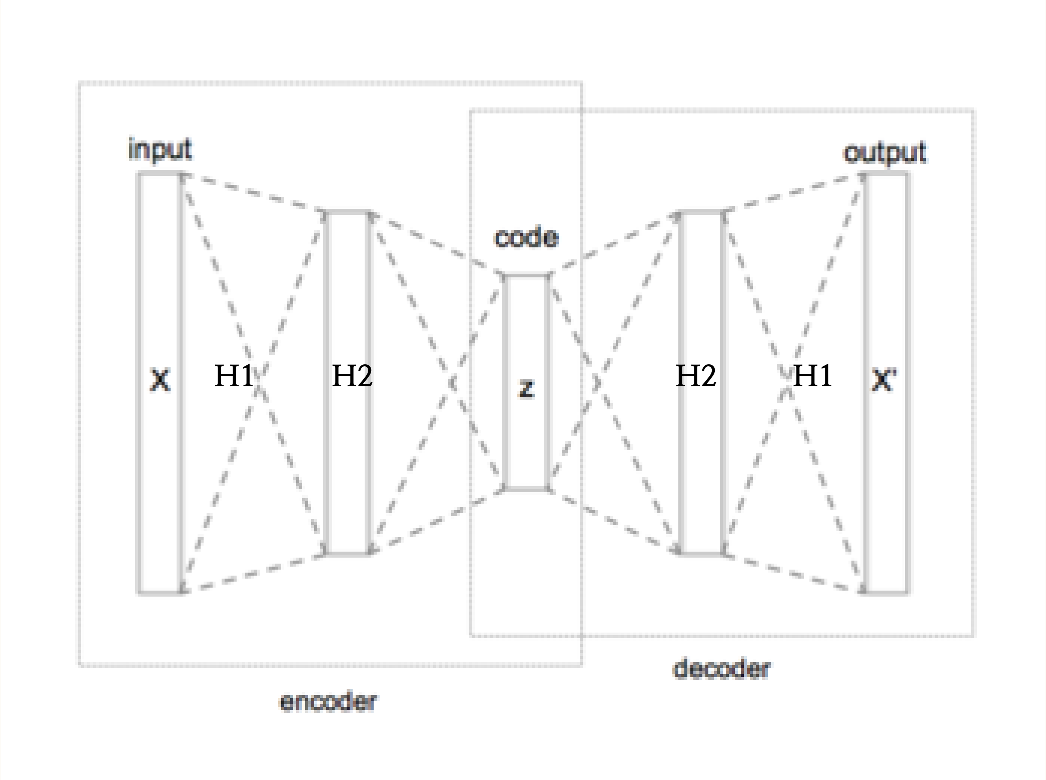


1) Hdim = 10, Zdim = 2





2) H1 dim = 15, H2 dim = 7, Zdim = 2



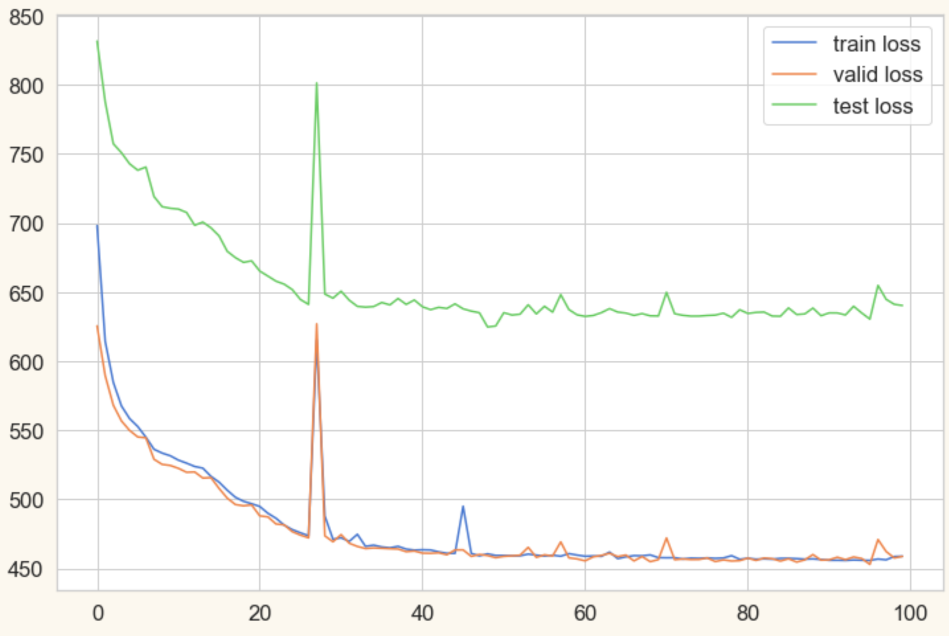
Input Dim = 29

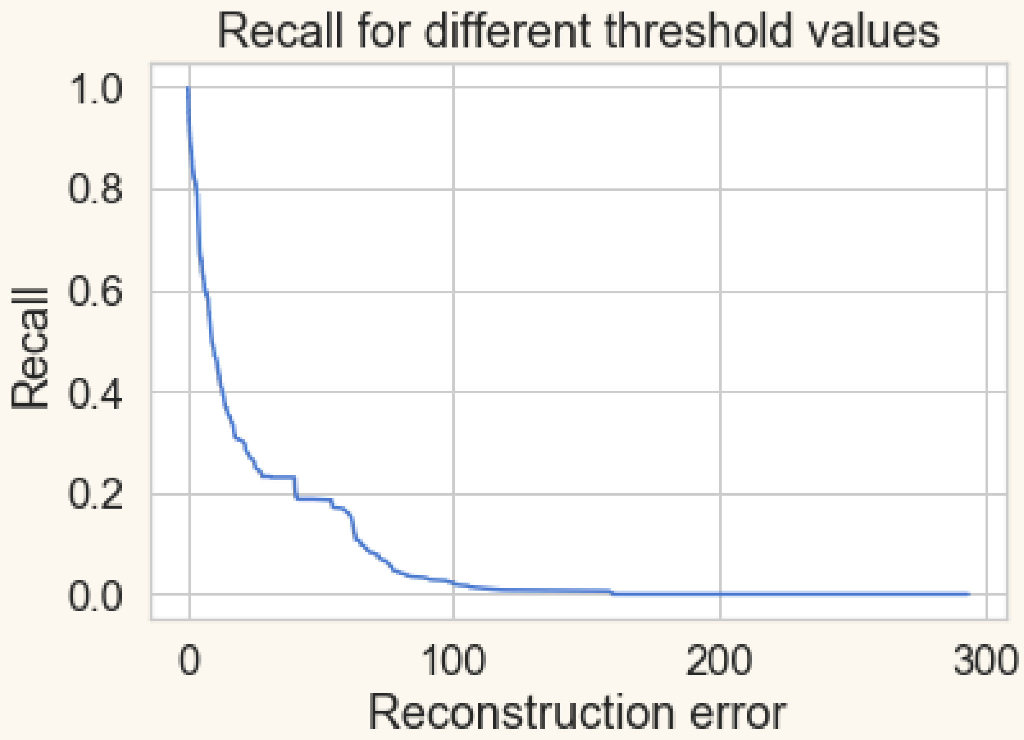
H1 Dim = 15

H2 Dim = 7

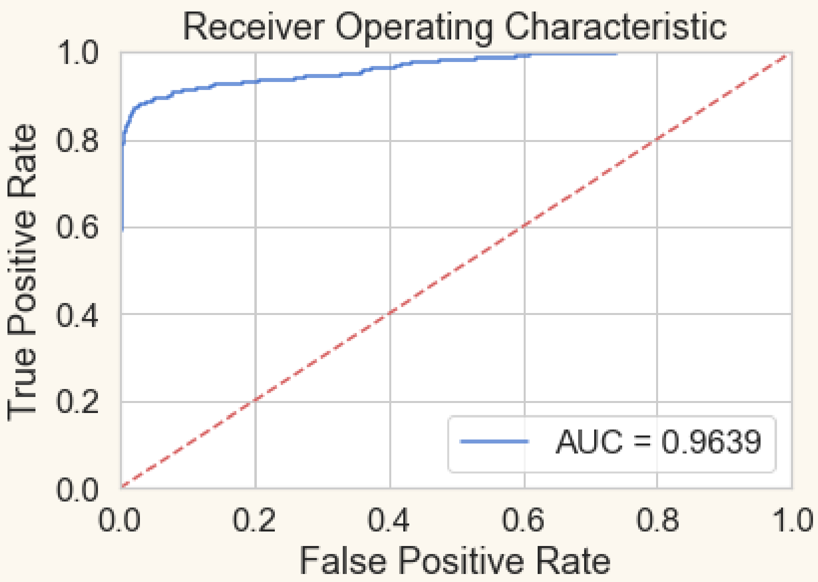
Z Dim = 2

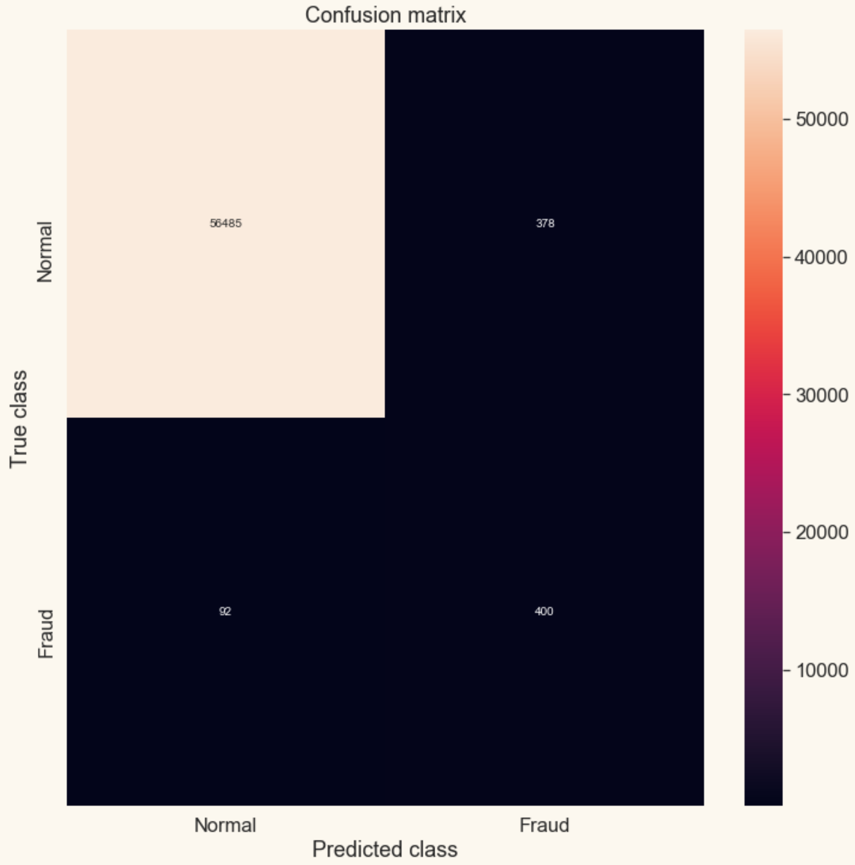
2) H1 dim = 15, H2 dim = 7, Zdim = 2



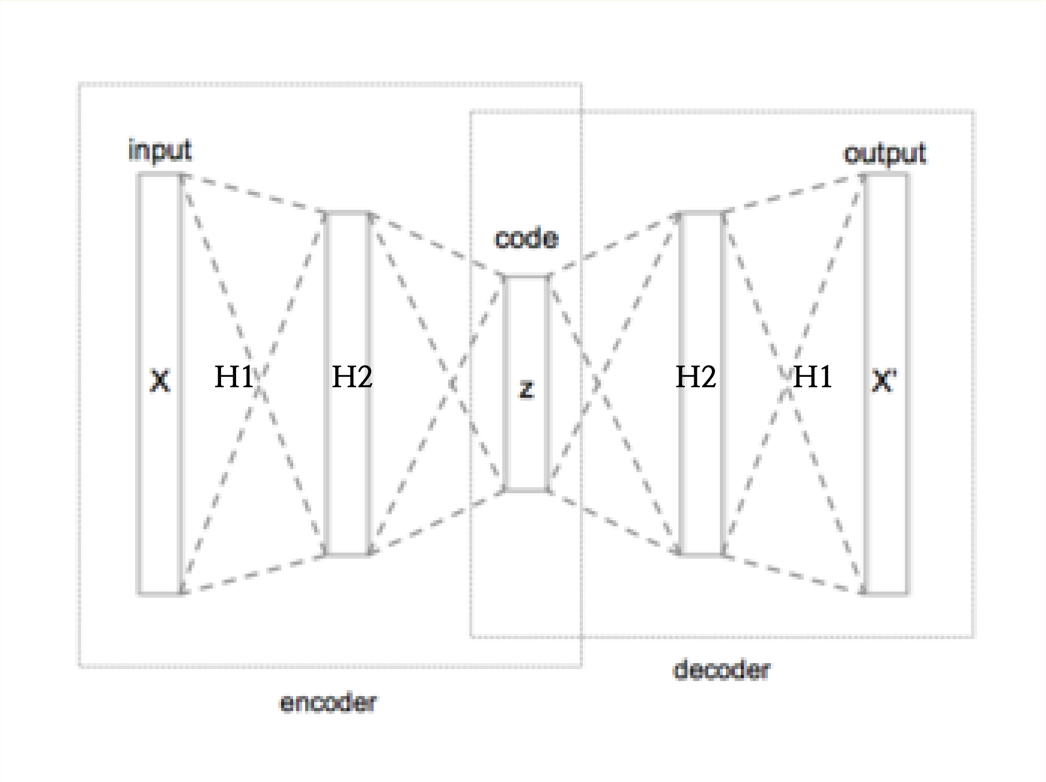


2) H1 dim = 15, H2 dim = 7, Zdim = 2





3) H1 dim = 17, H2 dim = 9, Zdim = 5



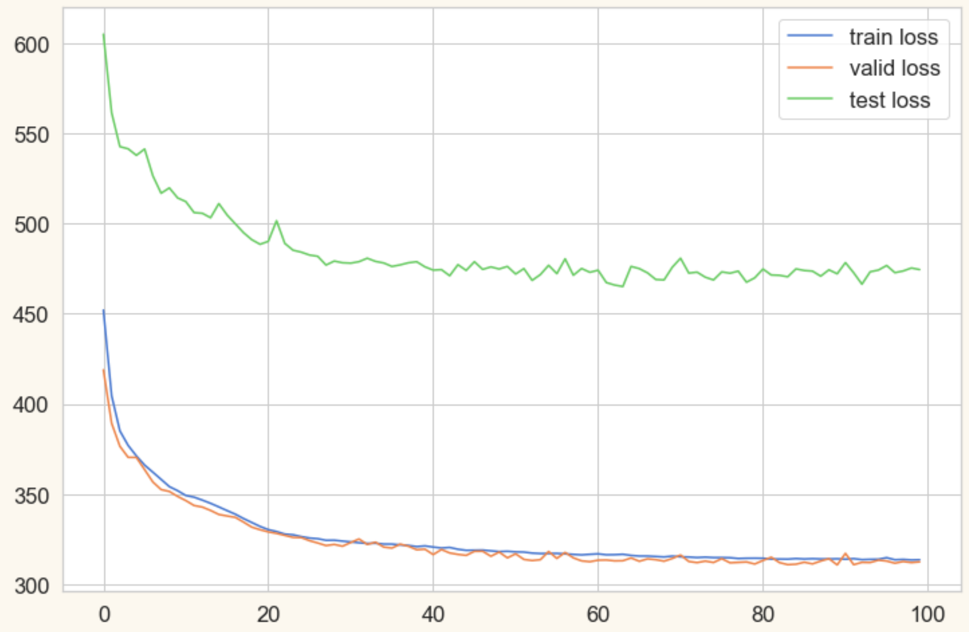
Input Dim = 29

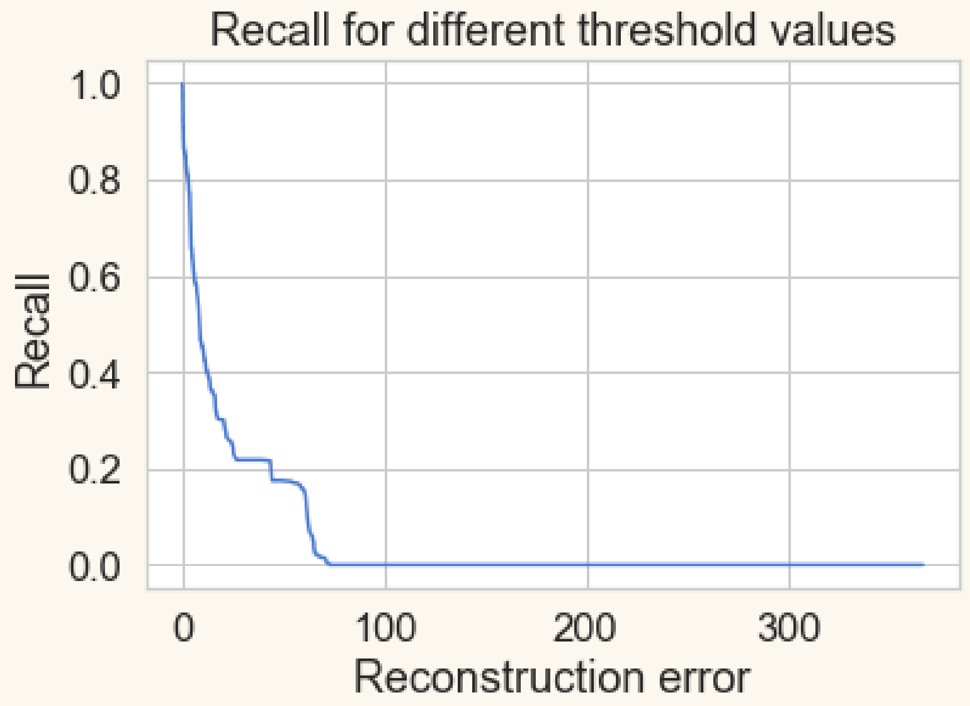
H1 Dim = 17

H2 Dim = 9

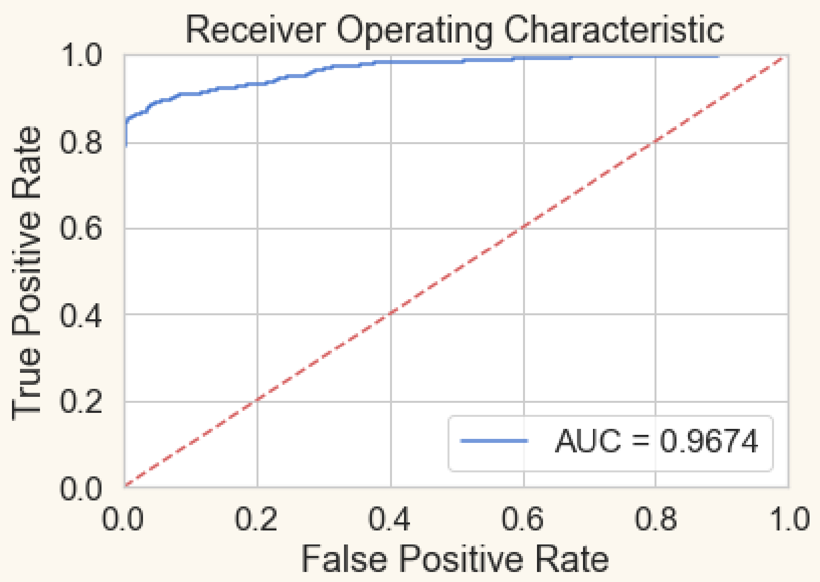
Z Dim = 5

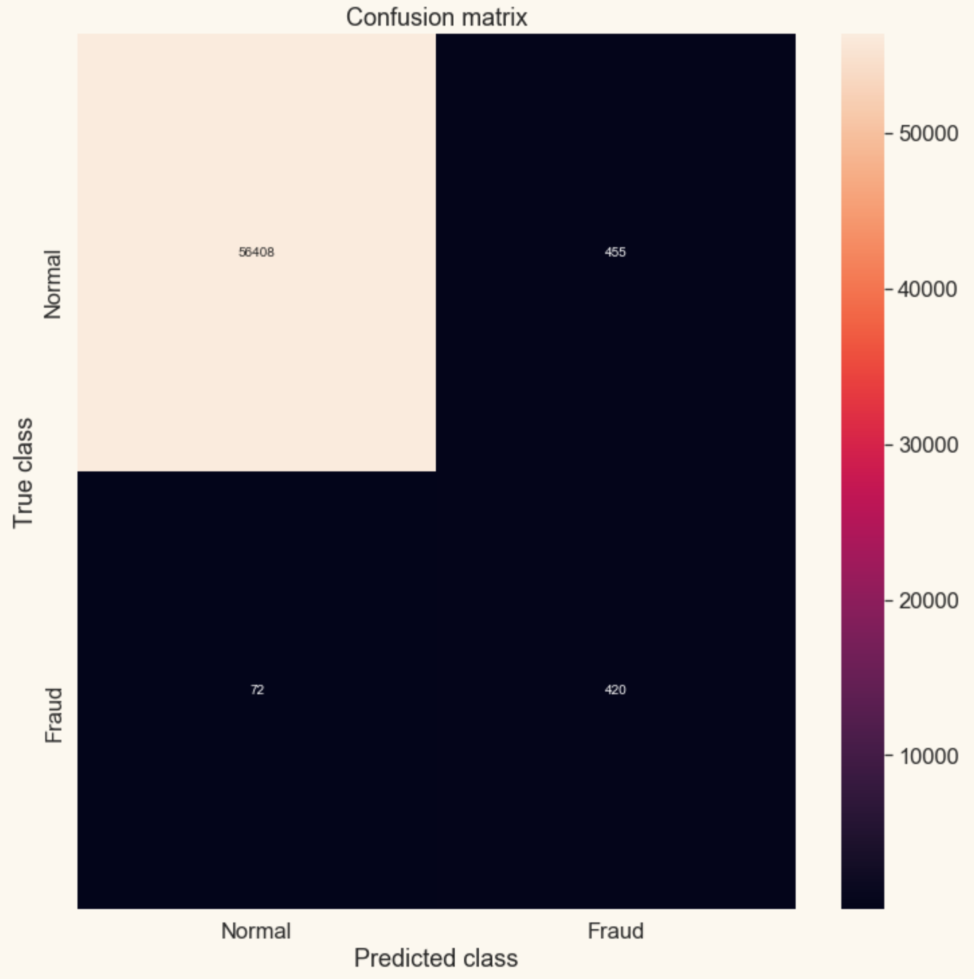
3) H1 dim = 17, H2 dim = 9, Zdim = 5



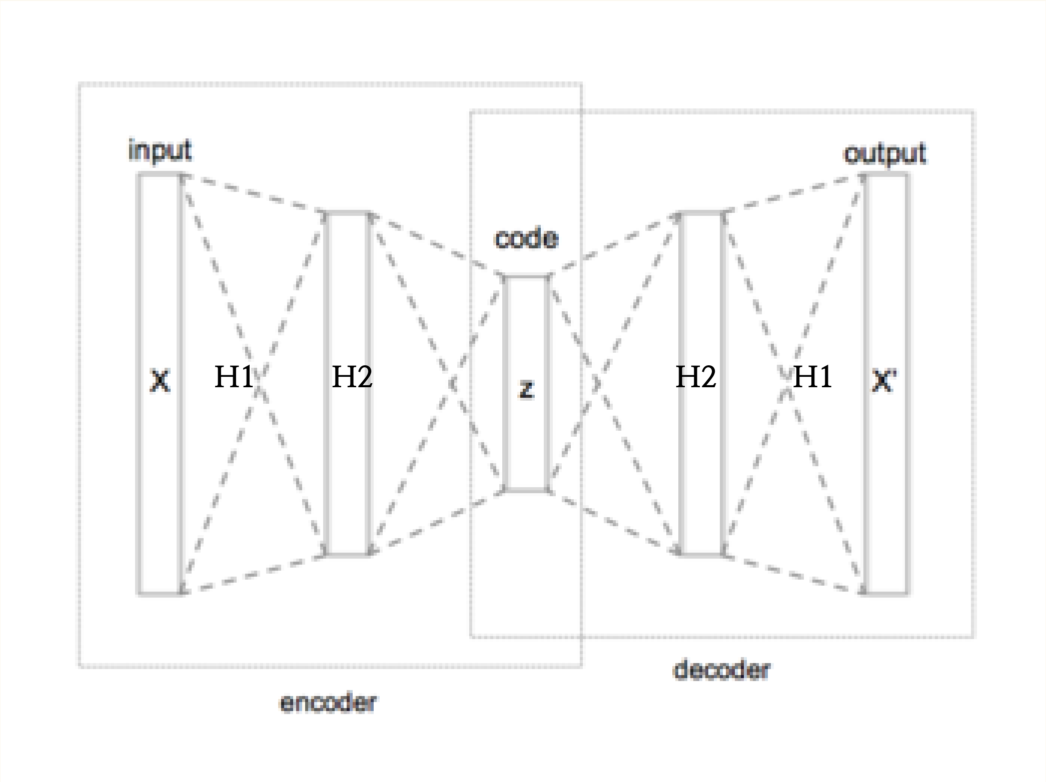


3) H1 dim = 17, H2 dim = 9, Zdim = 5





4) H1 dim = 20, H2 dim = 15, Zdim = 10



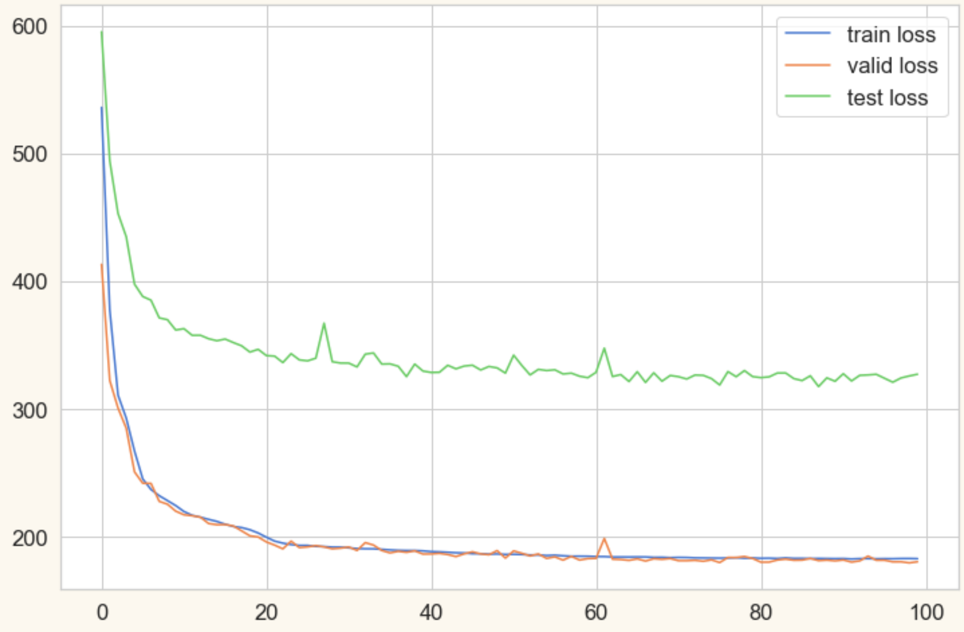
Input Dim = 29

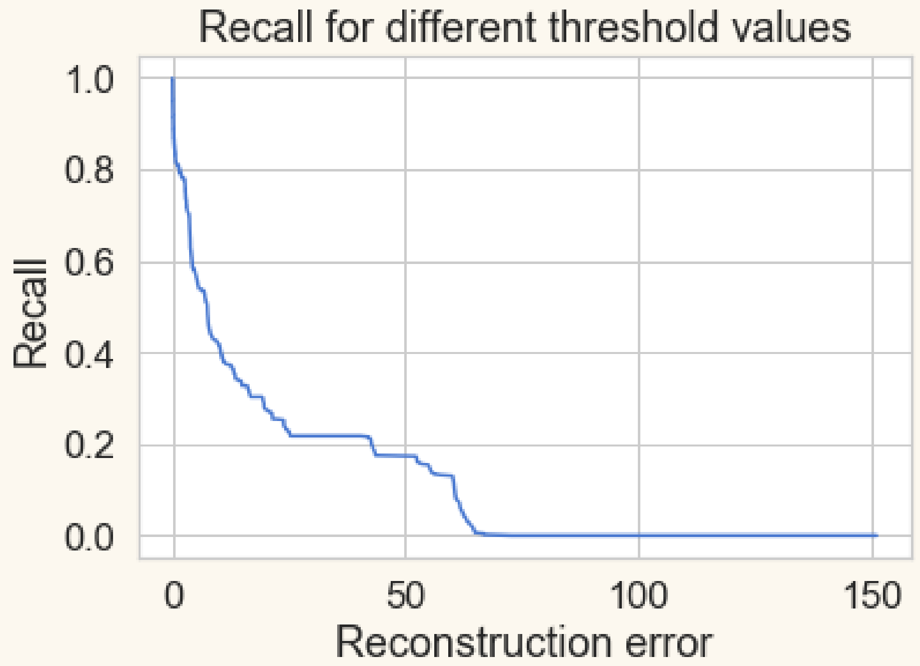
H1 Dim = 20

H2 Dim = 15

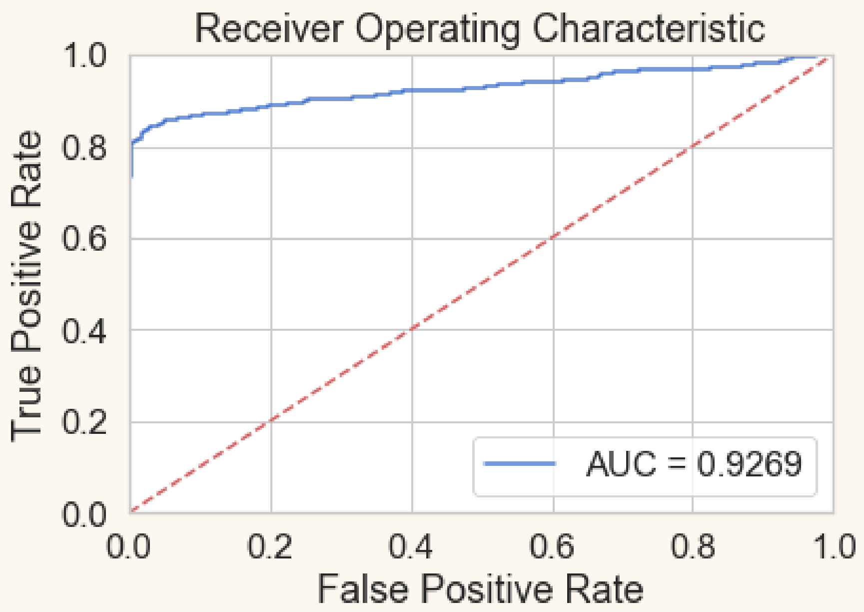
Z Dim = 10

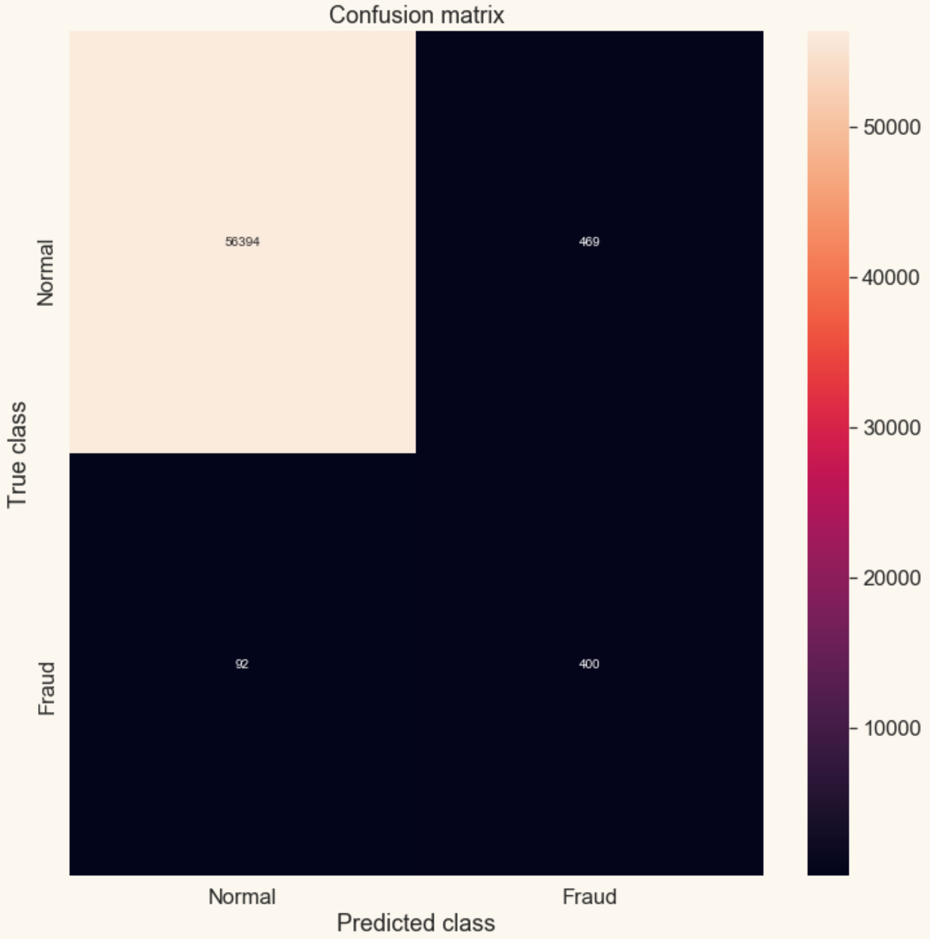
4) H1 dim = 20, H2 dim = 15, Zdim = 10





4) H1 dim = 20, H2 dim = 15, Zdim = 10





Conclusion

1. VAE with 3rd Model shows the best performance.
2. Generally, VAE models show better performance.
3. Danger of Overfitting Exists in VAE model.
4. Statistical Models are not always the worst.


Do you have any question?
Thank you for your attention.