

신용카드 사기 거래 탐지

Credit Card Fraud Detection

복습프로젝트 1조
강수민 김성윤 신현호



목차

- 01 주제 선정 및 목표
- 02 Data EDA
- 03 Modeling (1)
- 04 Modeling (2)
- 05 정리 및 결론



01

주제 선정 및 목표



DACON - 신용카드 사기 거래 탐지 AI 경진대회

신용카드 사기 거래 탐지 AI 경진대회

월간데이콘23 | 비지도 학습 | 이상 탐지 | 정형 | macro-f1

₩ 상금 : 100만원 + α

🕒 2022.07.01 ~ 2022.08.05 16:59

[+ Google Calendar](#)

👤 1,398명 📅 마감



- 비지도 학습(Unsupervised Learning)
- 이상 탐지(Anomaly Detection)
- 성능 지표: macro-f1



이상치 탐지



의료

암 발병률 데이터
악성 종양 데이터



제조업

제품 불량 판정
기계 결함 예측



금융

사기 거래 탐지
부도 예측

이상치

1. 정상 데이터와는 다른 매커니즘으로 생성된 데이터
2. 관측된 데이터의 범위에서 많이 벗어난 아주 작은 값이나 아주 큰 값



Goals

- 세션 복습하기
- 비지도 학습 경험하기
- 비식별화 데이터 다뤄보기
- 불균형 데이터 다뤄보기
- 협업하기



협업 Github

SungYune / dacon_FraudDetection Private

Unwatch 1 Fork 0 Star 0

<> Code Issues Pull requests Actions Projects Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

Shin-HH SHH e21a919 3 days ago 12 commits

code Delete OneClassSVM.ipynb 3 days ago

models SHH 3 days ago

submissions submissions 7 days ago

README.md Update README.md 7 days ago

README.md

신용카드 사기 거래 탐지 AI 경진대회

비식별화된 신용카드 거래 데이터로부터 사기 거래를 탐지하는 AI 솔루션 개발

Notion [our works](#)

About

2022 비타민 9기 복습프로젝트

📅 2022.07.17 ~ 2022.08.05

🕒 2022.08.07(SUN) 15:00

Members

강수민	김성윤	신현호
(@sum-k)	(@SungYune)	(@Shin-HH)

About

2022 비타민 9기 복습프로젝트

📖 Readme

☆ 0 stars

👁 1 watching

🍴 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Contributors 3

SungYune

sum-k


Shin-HH

Languages

Jupyter Notebook 100.0%



협업 Notion



BITAmin 복습프로젝트

← 1 backlink

신용카드 사기 거래 탐지 AI 경진대회

DACON | 비지도 학습 | 이상 탐지 | 정형 데이터 | macro-F1

dacon_FraudDetection
SungYune • Updated 9 minutes ago

Subgoals

- Github으로 협업하기
- 비식별화 데이터 다뤄보기
- 비타민 9기 세션 복습하기

Members

- 강수민
- 김성윤
- 신현호

Upcoming Events

- @July 24, 2022 8:00 PM Meeting 1
- @July 27, 2022 7:30 PM Meeting 2
- @Wednesday 7:30 PM Meeting 3
- @Thursday 7:30 PM Meeting 4
- @Today 7:30 PM Meeting 5

✓ 할 일

Board view

To-do 1

In Progress 3

Complete 3

비지도학습 Model 탐색
SungYune • 강수민 • 신현호
July 27, 2022

Auto-Encoder
신현호

One-class SVM

EDA
신현호
July 22, 2022

List view

진행 목록

Github	github	<input type="checkbox"/>
EDA	colab.research.google.com EDA	<input type="checkbox"/>
Local Outlier Factor	colab.research.google.com 0.4991564741 model	<input type="checkbox"/>
One-class SVM	colab.research.google.com 0.6458464983 model	<input type="checkbox"/>
PCA + EllipticEnvelope	colab.research.google.com 0.9248241501 model	<input type="checkbox"/>
Auto-Encoder + Sub-Sampling	colab.research.google.com 0.9190043503 model	<input type="checkbox"/>
Auto-Encoder	colab.research.google.com 0.910051264 model	<input type="checkbox"/>
Ensemble	ensemble	<input type="checkbox"/>
Submission	output	<input type="checkbox"/>
Presentation	presentation	<input type="checkbox"/>

+ New

17

프로젝트 일정

Calendar Table



02

Data EDA



Train Dataset

ID	신용 카드 거래 ID
V1	비식별화된 신용 카드 거래 Feature
V2	비식별화된 신용 카드 거래 Feature
V3	비식별화된 신용 카드 거래 Feature
⋮	⋮
V29	비식별화된 신용 카드 거래 Feature
V30	비식별화된 신용 카드 거래 Feature

1. 정답 Label이 존재하지 않는 Data
2. Feature들이 비식별화 되어있어 파생변수를 만들기 어려움
3. Over Sampling, Under Sampling 기법을 적용하기 어려움
4. Shape: (113842, 31)



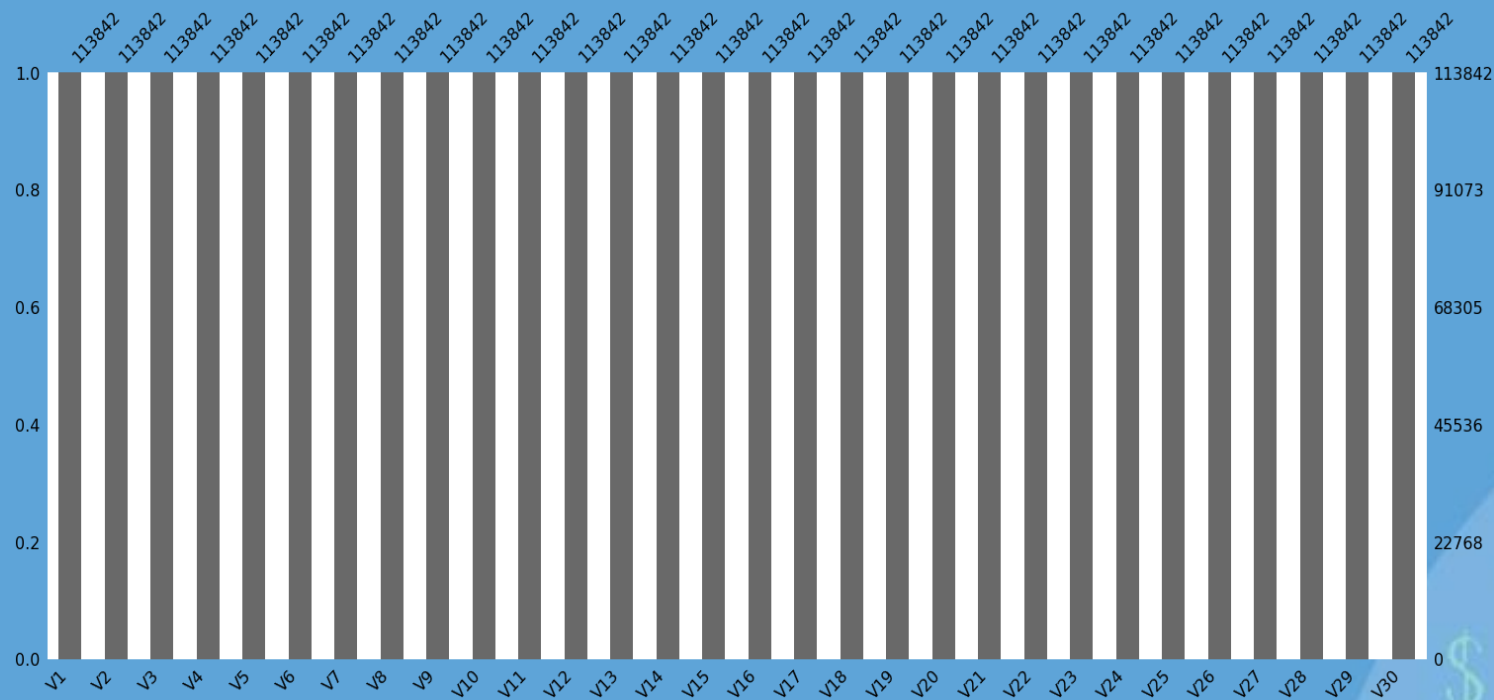
Validation Dataset

ID	신용 카드 거래 ID
V1	비식별화된 신용 카드 거래 Feature
V2	비식별화된 신용 카드 거래 Feature
⋮	⋮
V29	비식별화된 신용 카드 거래 Feature
V30	비식별화된 신용 카드 거래 Feature
Class	신용 카드 거래의 정상, 사기 여부(Target)

1. 정답 Label이 존재하는 Data
->하이퍼 파라미터 튜닝 가능
2. 검증 데이터에 존재하는 이상치 비율과
같은 비율로 학습데이터에 이상치가
존재한다고 가정할 수 있음
3. Shape: (28462, 32)



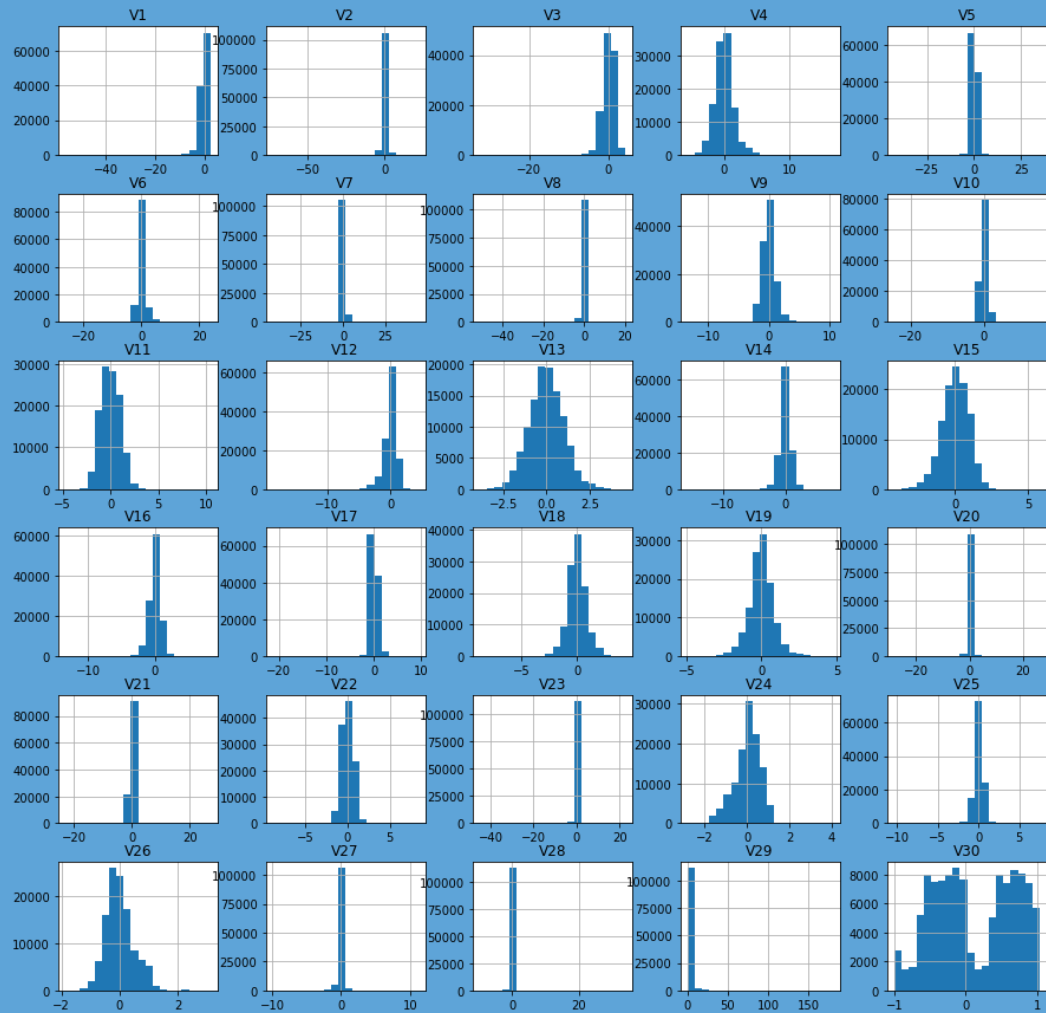
EDA - 결측치



결측치 존재하지 않음!



EDA - 데이터 분포



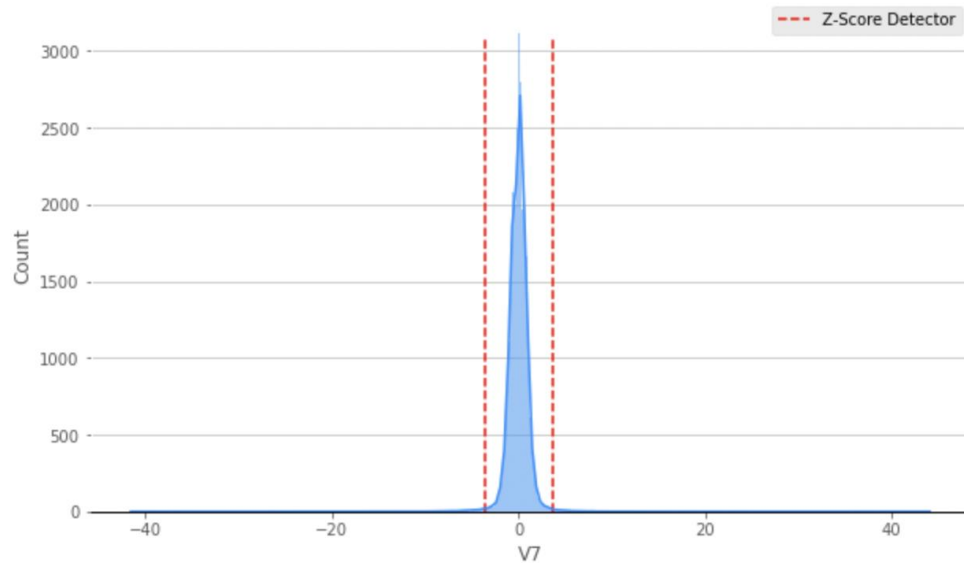
1. 대부분의 Feature가 정규화되어 있음
2. 분포의 형태가 다양함



EDA - 데이터 분포

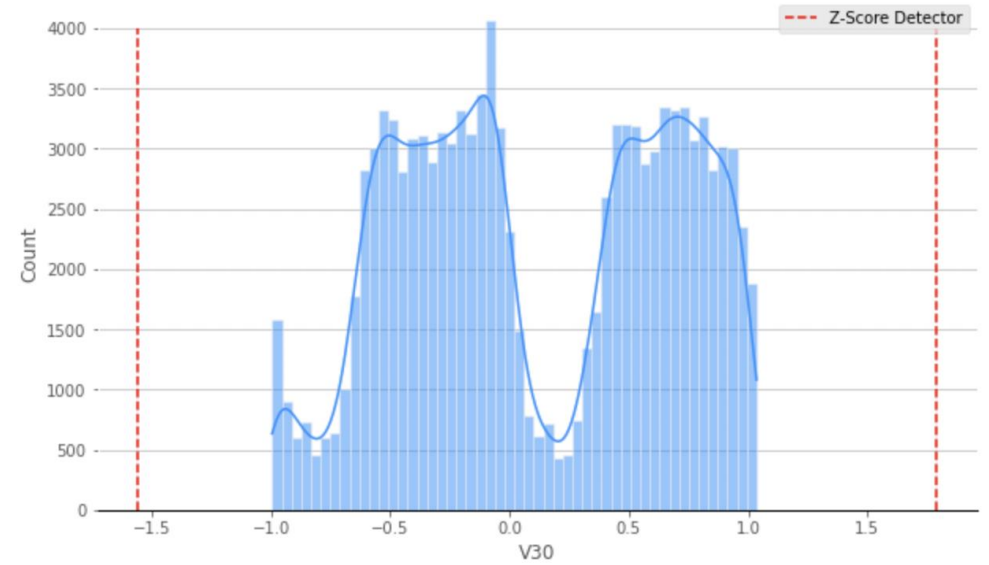
Feature: V7

평균: 0.00577
표준편차: 1.20411
중앙값: 0.03904
Z-Score기반 이상치 비율: 1.25086%

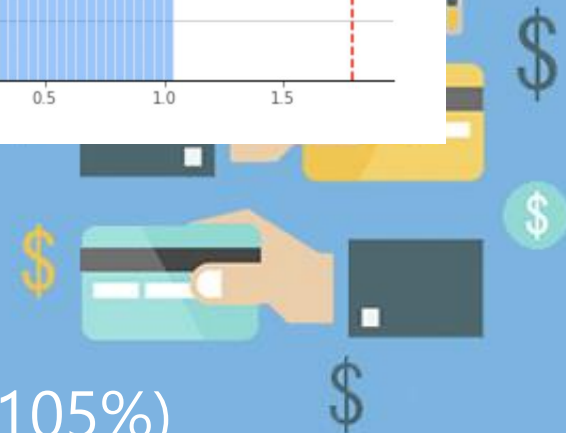


Feature: V30

평균: 0.11623
표준편차: 0.55816
중앙값: -0.00259
Z-Score기반 이상치 비율: 0.0%



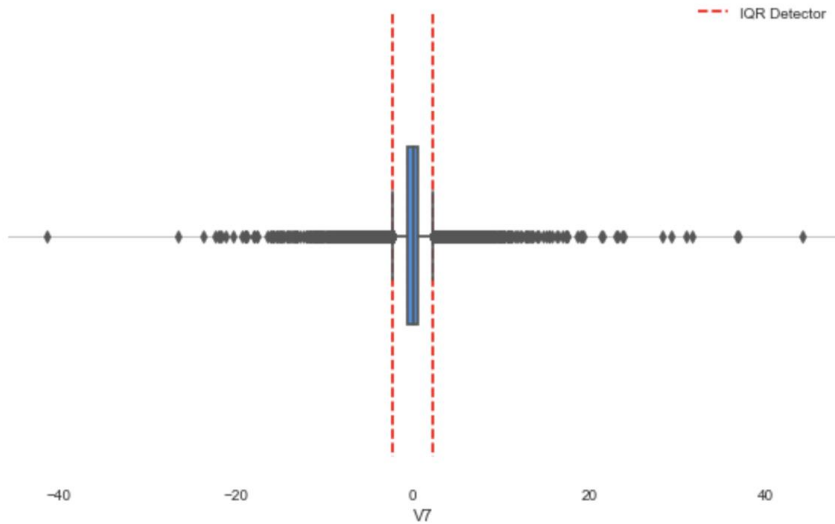
1. 분포가 많이 차이나는 것을 볼 수 있음
2. Z-Score 기반으로 이상치 탐지를 했을 때 너무 많은 데이터를 이상치로 간주(검증 데이터의 이상치 비율은 0.105%)



EDA - 데이터 분포

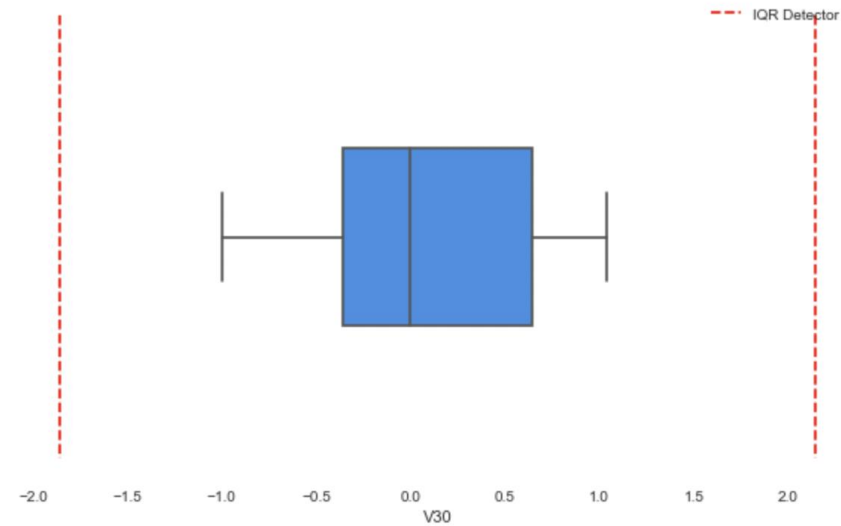
Feature: V7

평균: 0.00577
표준편차: 1.20411
중앙값: 0.03904
IQR기반 이상치 비율: 3.11221%



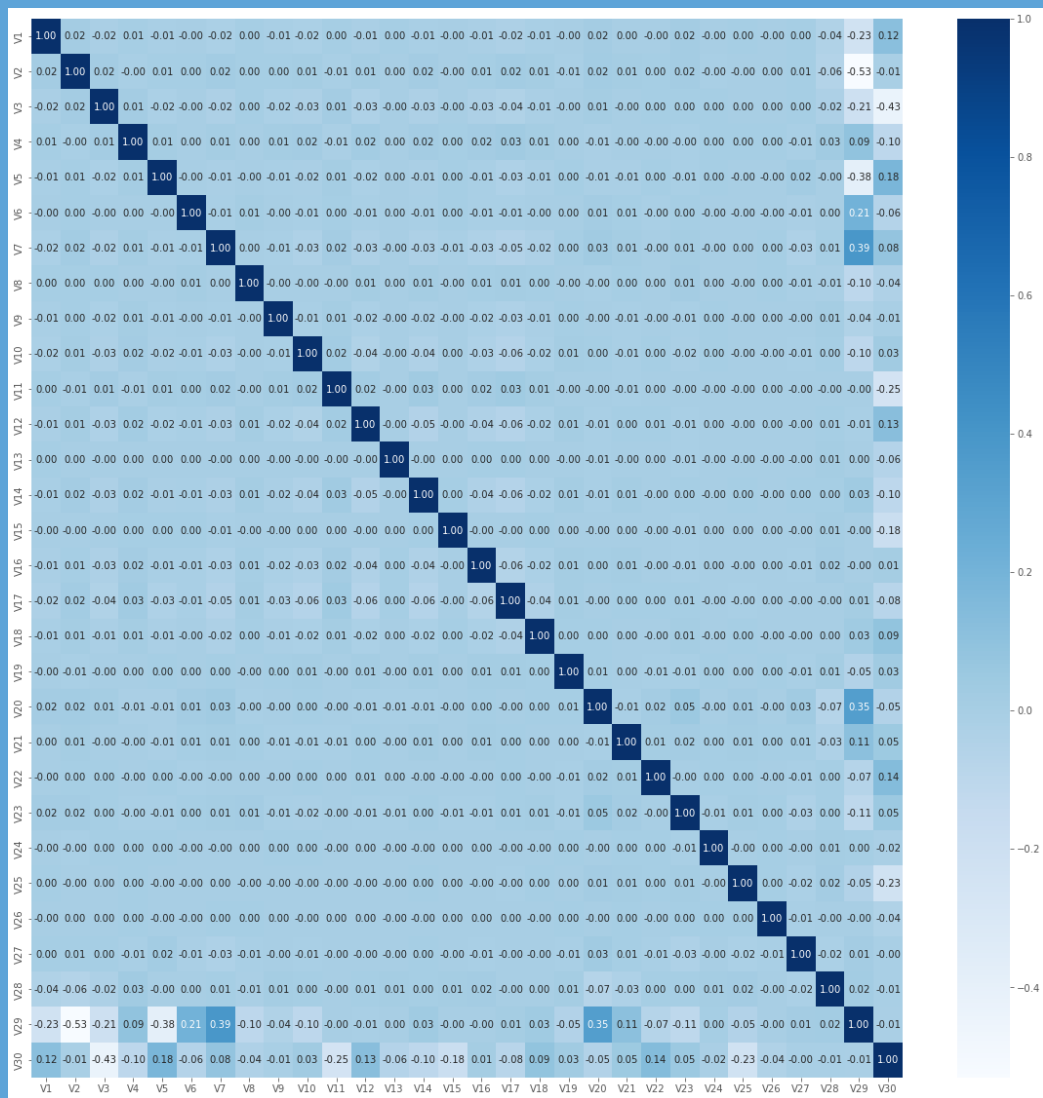
Feature: V30

평균: 0.11623
표준편차: 0.55816
중앙값: -0.00259
IQR기반 이상치 비율: 0.0%



1. 분포가 많이 차이나는 것을 볼 수 있음
2. IQR 기반으로 이상치 탐지를 했을 때 너무 많은 데이터를 이상치로 간주(검증 데이터의 이상치 비율은 0.105%)

EDA - 상관관계



Feature들 간의 상관관계가 거의 존재하지 않음!

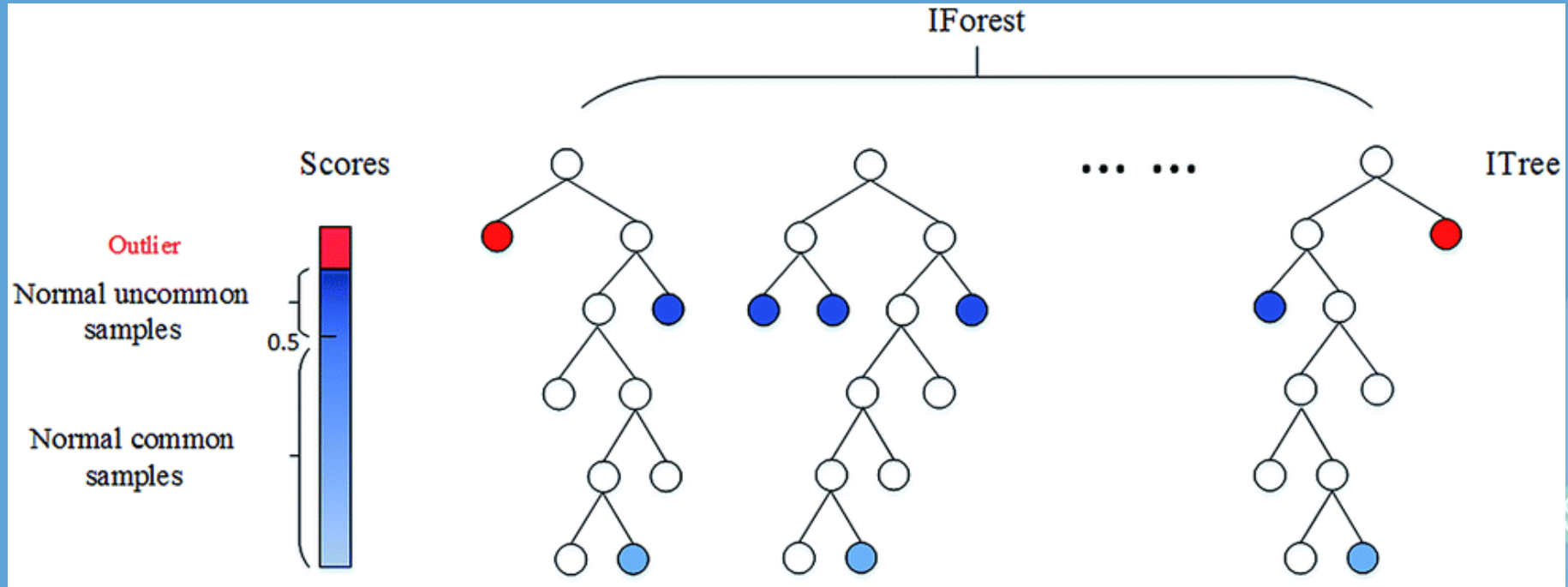


03

Modeling(1)



Isolation Forest



여러 개의 의사결정나무(decision tree)를 종합한 앙상블 기반의 이상탐지 기법
의사결정나무를 지속적으로 split하며 데이터 관측치의 고립 정도 여부에 따라 이상치 판별



Isolation Forest

- 이상 징후를 감지하기 위해 거리 또는 밀도 측정을 사용하지 않는다. 따라서 거리 및 밀도 측정 사용법보다 연산량이 적어 비교적 빠르다.
- 트리 모델 기반이기 때문에 데이터의 크기가 크거나 고차원 데이터에서도 효율적이다.



Isolation Forest

```
[ ] from sklearn.ensemble import IsolationForest
    from sklearn.metrics import f1_score, confusion_matrix
```

```
IF = IsolationForest(n_estimators=125, max_samples=len(train_x),
                    contamination=val_contamination, verbose=0)
IF.fit(train_x)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but
  "X does not have valid feature names, but"
IsolationForest(contamination=0.0010551491277433877, max_samples=113842,
                n_estimators=125, random_state=42)
```

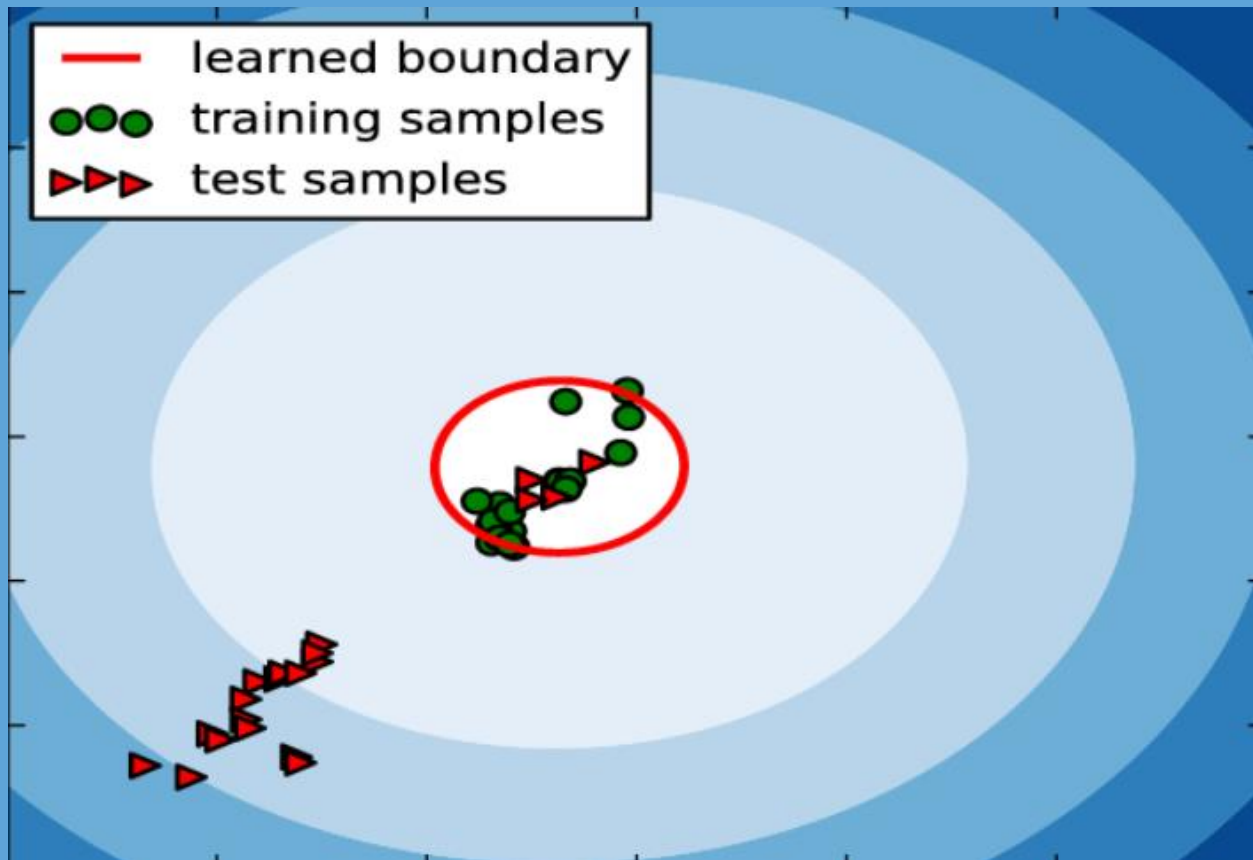
```
[ ] val_pred = IF.predict(val_x) # model prediction
    val_pred = get_pred_label(val_pred)
    val_score = f1_score(val_y, val_pred, average='macro')
    print(f'Validation F1 Score : [{val_score}]')
    print(classification_report(val_y, val_pred))
```

```
Validation F1 Score : [0.7030820840915222]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	28432
1	0.41	0.40	0.41	30
accuracy			1.00	28462
macro avg	0.71	0.70	0.70	28462
weighted avg	1.00	1.00	1.00	28462



One-Class SVM



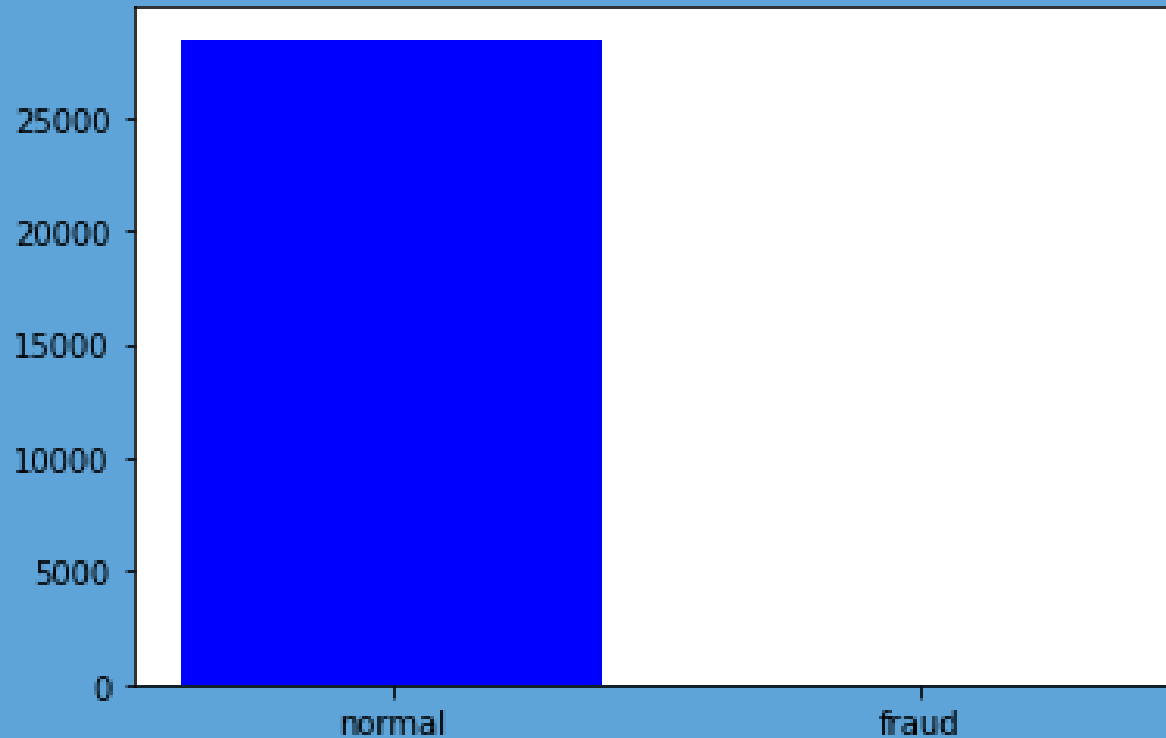
주어진 데이터를 잘 설명할 수 있는 최적의 support vector를 구하고 이 영역 밖의 데이터들은 outlier로 간주하는 방식

이상치 탐지, 이미지 검색, 문서/ 텍스트 분류 등에 사용



One-Class SVM

Class Distribution of validation dataset



```
index  Class
0      0   28432
1      1     30
```

fraud_ratio: 0.0010551491277433877



One-Class SVM

```
[ ] from sklearn.svm import OneClassSVM

clf = OneClassSVM(nu = 0.001, kernel = "rbf", gamma = 0.0001)
clf.fit(train_x)
```

```
OneClassSVM(gamma=0.0001, nu=0.001)
```

```
[ ] val_pred = clf.predict(val_x) # model prediction
val_pred = get_pred_label(val_pred)
val_score = f1_score(val_y, val_pred, average='macro')
print(f'Validation F1 Score : [{val_score}]')
print(classification_report(val_y, val_pred))
```

```
Validation F1 Score : [0.703422360288898]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	28432
1	0.46	0.37	0.41	30
accuracy			1.00	28462
macro avg	0.73	0.68	0.70	28462
weighted avg	1.00	1.00	1.00	28462



04

Modeling(2)



ML-based Model

1. Local Outlier Factor

Density based method - 상대적인 밀도를 고려

```
1 # LocalOutlierFactor
2
3 from sklearn.neighbors import LocalOutlierFactor
4
5 lof = LocalOutlierFactor(n_neighbors=20, contamination=val_contamination, novelty=True)
6 lof.fit(X_train_svd)
```

`sklearn.neighbors.LocalOutlierFactor`

```
class sklearn.neighbors.LocalOutlierFactor(n_neighbors=20, *, algorithm='auto', leaf_size=30, metric='minkowski', p=2,
metric_params=None, contamination='auto', novelty=False, n_jobs=None) \[source\]
```



ML-based Model

2. Elliptic Envelope

가우시안 분포의 데이터셋을 가정

```
1 # EllipticEnvelope
2
3 from sklearn.covariance import EllipticEnvelope
4
5 ee = EllipticEnvelope(support_fraction = 0.994, contamination = val_contamination, random_state = 42)
6 ee.fit(train_pca)
```

sklearn.covariance.EllipticEnvelope

`class sklearn.covariance.EllipticEnvelope(*, store_precision=True, assume_centered=False, support_fraction=None, contamination=0.1, random_state=None)`

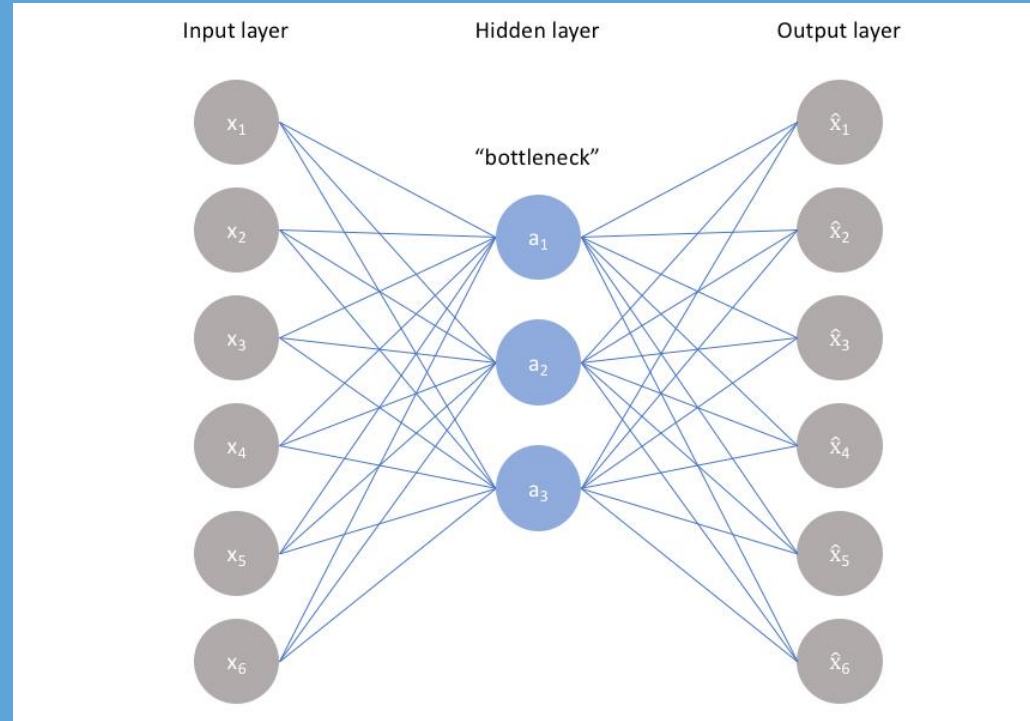
[\[source\]](#)

An object for detecting outliers in a Gaussian distributed dataset.



DL-based Model

3. Autoencoder

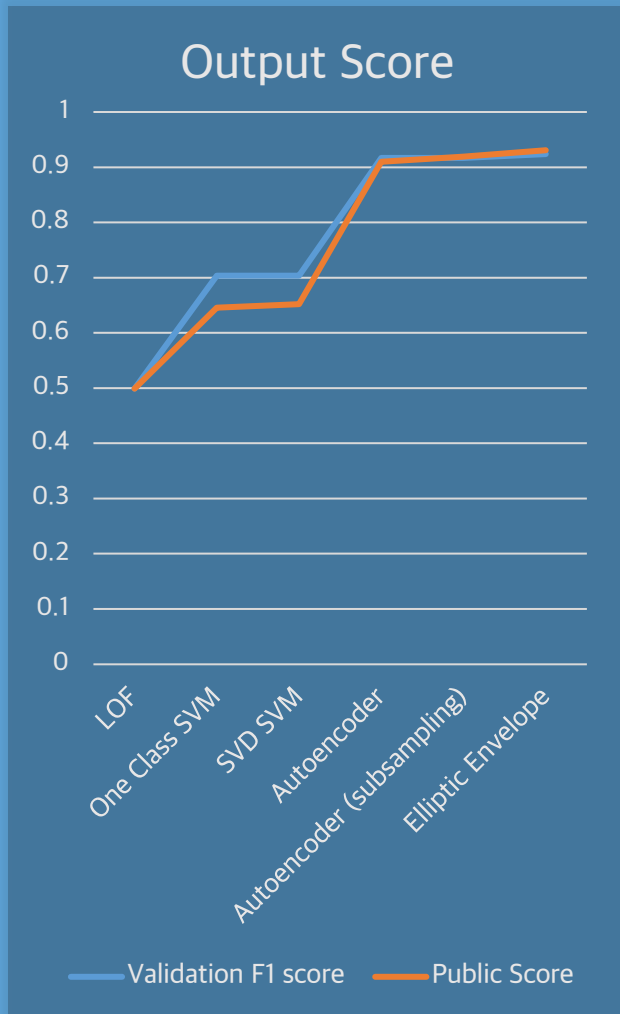


05

정리 및 결론



Final Output



	LOF	One Class SVM	SVD SVM	Autoencoder	Autoencoder (subsampling)	Elliptic Envelope
Validation F1 score	0.49897	0.70342	0.70342	0.91658	0.91658	0.92365
Public Score	0.49916	0.64585	0.65223	0.91005	0.91900	0.93052

- One Model
- Ensemble
- Subsampling

→ Elliptic Envelope 모델의 성능이 0.93052로 가장 높았다.



Conclusion

“비식별화된 신용카드 거래 데이터로부터 사기 거래를 탐지하는 AI 솔루션을 개발”하는 DACON 대회 참가

1. 모델 개발에는 태스크와 데이터에 대한 이해가 선행되어야 한다.
2. 모델의 성능 향상을 위해 도메인 지식과 각 태스크에 적합한 모델의 종류, 그리고 그 작동방식까지 아는 것이 중요하다.
3. 이상치 탐지와 비지도학습을 공부하며 비타민 2학기를 준비!



감사합니다!

강수민 김성윤 신현호

