(34-A) SKLearn Practice A [1/3]

- **Sklearn 강의 PPT**에서 배운 알고리즘을 선택하여 데이터의 class 분류 모델
  - 앞서 시각화 과제 결과를 바탕으로 과제 진행 (32-C)
  - 2개 이상의 알고리즘 선택

- 보고서 양식
    - 데이터 요약
    - 데이터 전처리
    - 알고리즘 별 실험 과정 및 결과 설명
    - 총 정리
  - 코드 : py 파일, python notebook

- 제출 기한 : To be announced

(34-A) SKLearn Practice A [2/3]

- Data Set – sklearn_pratice.csv (28만 lines)
  - 284,807 records, 31 attributes
  - 각 record는 0,1로 표기된 두 개의 class로 분류
  - 각 attribute는 정보는 비공개

Time	V1	V2	V3	...	V28	Amount	Class
0	-1.35981	-0.07278	2.536347	...	-0.02105	149.62	0
0	1.191857	0.266151	0.16648	...	0.014724	2.69	0
1	-1.35835	-1.34016	1.773209	...	-0.05975	378.66	0
1	-0.96627	-0.18523	1.792993	...	0.061458	123.5	0
2	-1.15823	0.877737	1.548718	...	0.215153	69.99	0
2	-0.42597	0.960523	1.141109	...	0.08108	3.67	0
4	1.229658	0.141004	0.045371	...	0.005168	4.99	0

(34-A) SKLearn Practice A [3/3]

- Guide Lines
  1. 데이터 읽기
     - Pandas.read_csv 등
  2. 시각화 과제 결과를 기준으로 데이터 전처리
     - 불필요한 attribute 제거
     - Feature engineering
     - 알고리즘에 맞는 전처리
     - 학습/테스트 데이터 분리 등
  3. 모델 학습
     - 알고리즘 parameter 조정하며 반복 실험
  4. 실험 결과 보고서 작성
     - 실험 과정 및 결과
     - 시각화 과제와의 연관성 등

목차

목적

- 주어진 데이터를 통해 카드 거래 이상 탐지를 할 수 있는 모델을 학습 하시오.
  - Fraud/Non-Fraud를 구분할 수 있는 모델 학습
  - Binary Classification

목차

Dataset Detail

- 2013년 9월 신용카드 거래 기록
- 2일 거래 기록 (492 frauds / 284,807 transactions.)
  - unbalanced, 0.172% the positive class (frauds)

- 기밀성 유지를 위해, original features 와 background 정보 제공 X
  - the result of a PCA 변환. (V1 – V28)
  - 'Time' 은 첫번째 거래로부터 현재 거래까지의 시간
  - 'Amount'는 거래량
  - 'Class' 는 1 (Fraud, 사기) 또는 0 (normal, 일반)

❖ 본 과제에서는 '(34-data A) card_fraud.csv'로 제공

목차

데이터 읽고 파악하기

- 주어진 csv 파일 pandas module로 읽기

```
import pandas as pd

fraud_df = pd.read_csv(' (34-data A) card_fraud.csv')
```

- 데이터의 형태 파악

```
fraud_df.head()
fraud_df.tail()
```

- Attribute 확인

```
Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',
       'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20',
       'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount',
       'Class'],
      dtype='object')
```

데이터 읽고 파악하기

- 통계 값 확인

	Time	V1	V2	V3	V4	V5	V6	V7	V8	
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070
mean	94813.859575	1.165980e-15	3.416908e-16	-1.373150e-15	2.086869e-15	9.604066e-16	1.490107e-15	-5.556467e-16	1.177556e-16	-2.406455
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+00	1.332271e+00	1.237094e+00	1.194353e+00	1.098632
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+02	-2.616051e+01	-4.355724e+01	-7.321672e+01	-1.343407
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-01	-7.682956e-01	-5.540759e-01	-2.086297e-01	-6.430976
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-02	-2.741871e-01	4.010308e-02	2.235804e-02	-5.142873
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-01	3.985649e-01	5.704361e-01	3.273459e-01	5.971390
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+01	7.330163e+01	1.205895e+02	2.000721e+01	1.559499

데이터 읽고 파악하기

- 결측치 확인

  fraud_df.isna().sum()

- 데이터 attribute 경향성 확인

  e.g.) min max에 비해 mean이 기울어있고, std가 낮음
  # Hint – 통계치와 histogram등과 같은 그림을 이용하면 쉽게 확인 가능

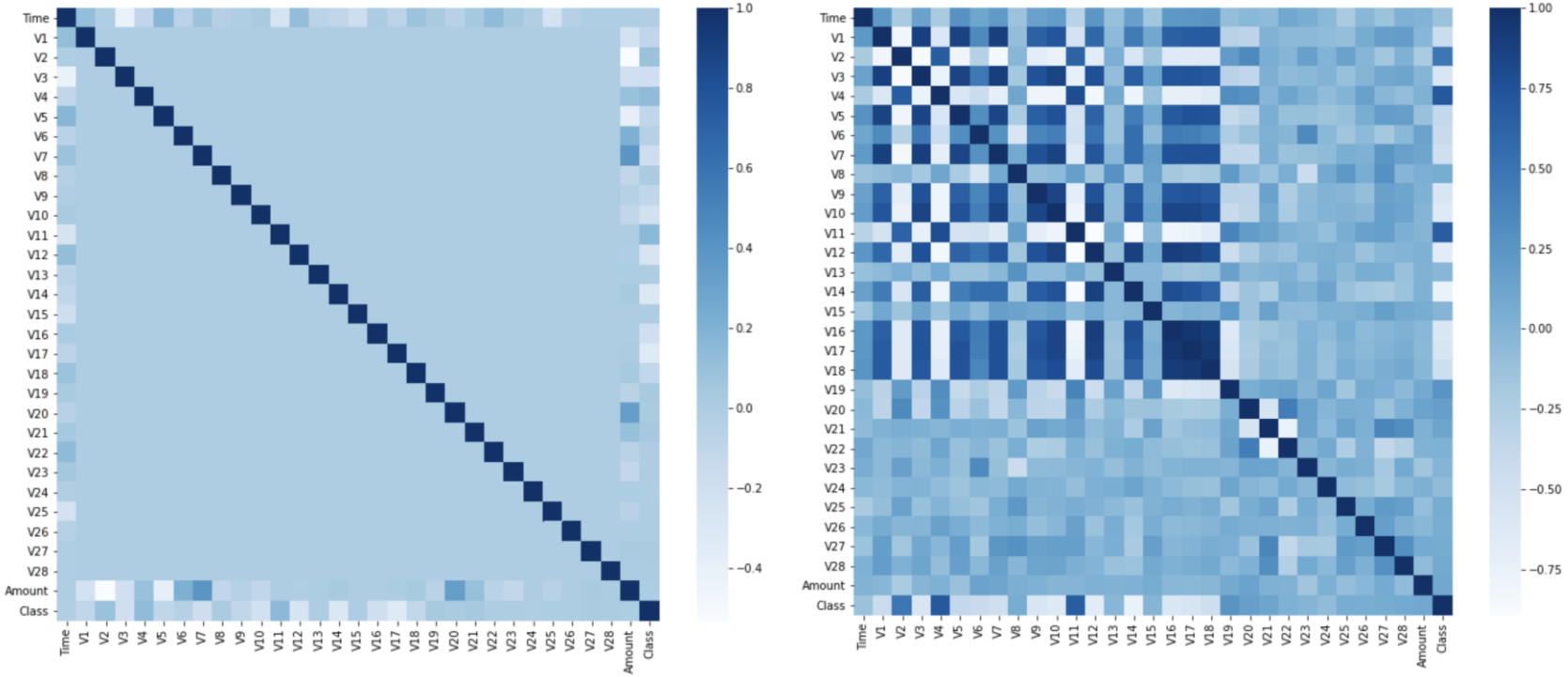
- Class 비율 확인

```
Class 0: 99.827251437
Class 1: 0.172748563
```

데이터 읽고 파악하기

- 데이터 attribute 간 상관관계 파악하기

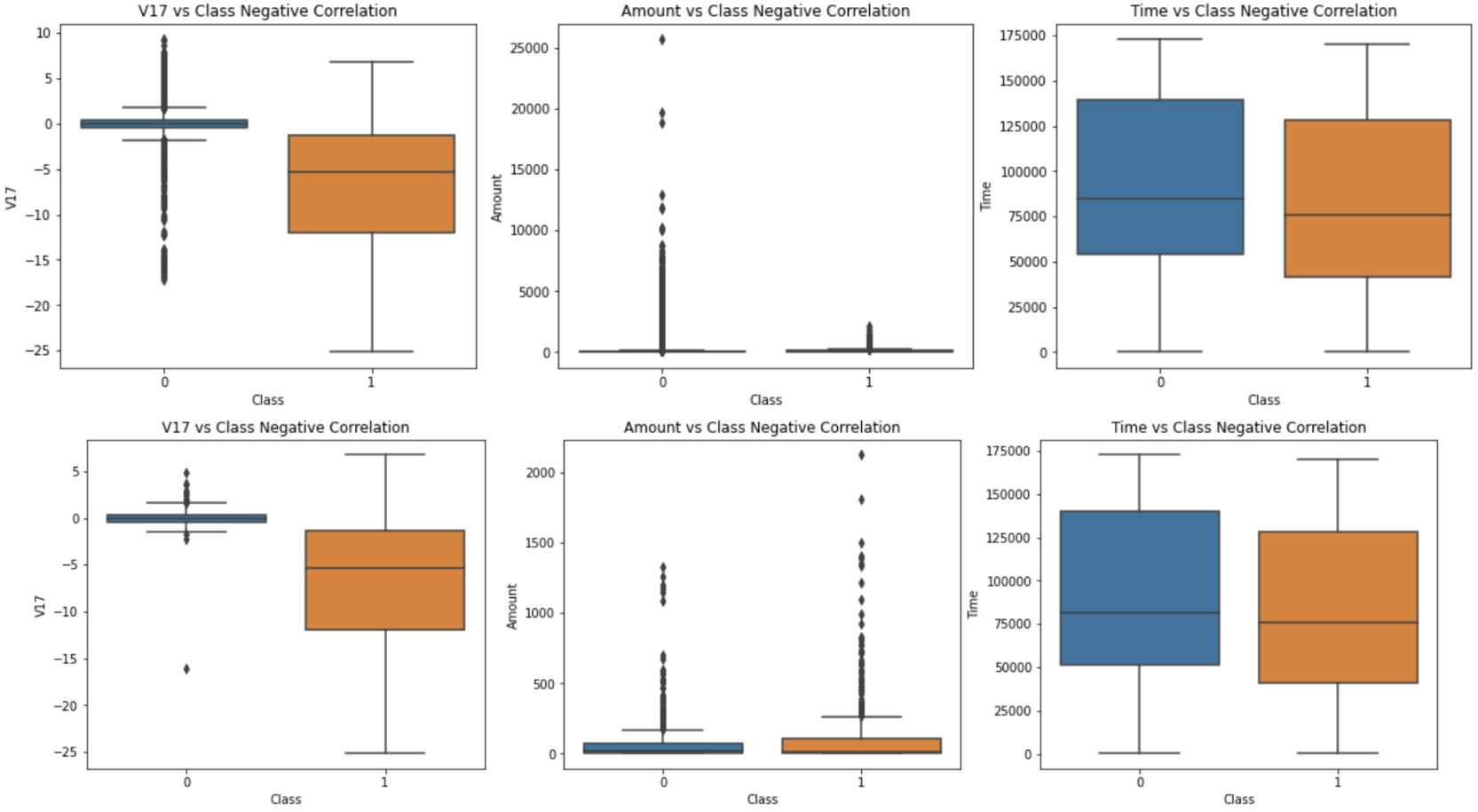
  # Hint – heatmap, boxplot 등과 같은 그림을 이용하면 쉽게 확인 가능



데이터 읽고 파악하기

- 데이터 attribute 간 상관관계 파악하기

Hint – heatmap, boxplot 등과 같은 그림을 이용하면 쉽게 확인 가능



목차

데이터 전처리

- 데이터의 불균형 해결

  # Hint – 데이터를 undersampling 또는 oversampling을 통해 균형을 맞춤

  ```
  fraud = fraud_df[fraud_df.Class == 1]
  non_fraud = fraud_df[fraud_df.Class == 0]
  sampled_non_fraud =
  balanced_df = pd.concat([fraud, sampled_non_fraud])
  ```

- 사용할 attribute 설정

  # Hint – 파악한 데이터를 바탕으로 분류에 활용할 데이터 attribute 설정

데이터 전처리

- 데이터 값 Scaling
  - 데이터 attribute 마다 값의 범위가 다르기 때문에, scaling을 통해 조정

    e.g.)

```
# balanced_df 와 using_col은 이전 과정에서 구할 수 있음
_df = balanced_df[using_col]
_df['Amount'] /= _df['Amount'].max()
```

- 학습에 사용할 데이터와 테스트에 사용할 데이터를 나눔

```
from sklearn.model_selection import train_test_split

X = _df.drop('Class', axis=1)
y = _df['Class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

X_train = X_train.values
X_test = X_test.values
y_train = y_train.values
y_test = y_test.values
```

목차

모델 선택 및 구현

- Transaction의 class를 0과 1로 classification하는 문제
  - Logistic Regression
  - KNN
  - Decision Tree Classifier
  - Neural Network Classifier
  - Etc..

- 구현
  - Sklearn의 모듈 활용하여 구현

```
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPclassifier
```

모델 선택 및 구현(예시)

```
# Logistic Regression 을 활용하여 Classification

from sklearn.linear_model import LogisticRegression

logistic_C = LogisticRegression()
logistic_C.fit(X_train, y_train)

print("training data score: ")
print(logistic_C.score(X_train, y_train))

print("test score: ")
print(logistic_C.score(X_test, y_test))
```

목차

모델 학습

- 변경할 수 있는 parameter를 변경하면서 모델 학습 진행
  - Regularization 없이 학습

```
# Logistic Regression 을 활용하여 Classification
from sklearn.linear_model import LogisticRegression

logistic_C = LogisticRegression(penalty='none')
logistic_C.fit(X_train, y_train)

print("training data score: ")
print(logistic_C.score(X_train, y_train))

print("test score: ")
print(logistic_C.score(X_test, y_test))
```

목차

모델 평가

- 두가지 이상의 알고리즘 선택하여 성능 비교 진행
  - 전처리 방법
  - Parameter 변경