

```
In [ ]: # 구글 드라이브 입력
#from google.colab import drive
#drive.mount('/content/gdrive/')
```

Drive already mounted at /content/gdrive/; to attempt to forcibly remount, call drive.mount("/content/gdrive/", force_remount=True).

설치 해야하는 것들 및 함수들

```
In [ ]: # pip install 요소들 정리
# annoy는 반드시 visual studio build tools 설치
#! pip install -q annoy
#! pip install -q FRUFS
#! pip install -q pacmap
```

```
In [1]: # import할 요소들 정리
import copy
import pandas as pd
import numpy as np
import warnings
import pacmap

from FRUFS import FRUFS
from lightgbm import LGBMRegressor

from sklearn.ensemble import IsolationForest

from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import recall_score

from sklearn.decomposition import PCA
from sklearn.decomposition import KernelPCA

import matplotlib.pyplot as plt

import os

import seaborn as sns

import matplotlib.pyplot as plt
import matplotlib.font_manager
from sklearn import svm

from scipy.stats import ranksums

# 파일 위치 고정
os.chdir("open")

# Train dataset
train_df = pd.read_csv('train.csv')
train_df = train_df.iloc[:,1:]

# Validation dataset
val_df = pd.read_csv('val.csv')
ori_val_df = val_df.iloc[:,1:]
```

```

val_class = val_df.iloc[:,31]
val_df = val_df.iloc[:,1:31]

# Test dataset
test_df = pd.read_csv('test.csv')
test_df = test_df.iloc[:,1:]

warnings.filterwarnings(action='ignore')

```

```

In [2]: # IsolationForest 모델 출력 (1:정상, -1:불량(사기)) 이므로 (0:정상, 1:불량(사기))
def get_pred_label(model_pred):
    model_pred = np.where(model_pred == 1, 0, model_pred)
    model_pred = np.where(model_pred == -1, 1, model_pred)
    return model_pred

# IsolationForest 예측만 하는 함수
def iso_for_model_prediction(the_contamination, trtrtr):
    # train dataset으로 isolationforest 모델 학습
    model_only_train = IsolationForest(n_estimators=1000, contamination=the_contamination)
    model_only_train.fit(trtrtr)

    # train dataset의 isolationforest 모델로 예측
    train_pred = model_only_train.predict(trtrtr)
    train_pred = get_pred_label(train_pred)

    return train_pred

# Pacmac + IsolationForest 예측과 비교할 수 있는 함수
def pac_iso_for_model_comparing(the_contamination, trtrtr, low_dim, compare_class):
    # train dataset으로 isolationforest 모델 학습
    dlatl_embedding = pacmap.PaCMAP(n_components=low_dim, n_neighbors=None, MN_ratio=0.05)
    pac_mac_train = dlatl_embedding.fit_transform(np.array(trtrtr), init="pca")

    model_train_compare = IsolationForest(n_estimators=1000, contamination=the_contamination)
    model_train_compare.fit(pac_mac_train)

    # train dataset의 isolationforest 모델로 예측
    train_pred = model_train_compare.predict(pac_mac_train)
    train_pred = get_pred_label(train_pred)

    # train dataset의 예측치와 compare data의 수치 비교
    train_score = f1_score(compare_class, train_pred, average='macro')

    print(f'Compared Macro F1 Score : [{train_score}]')
    print(classification_report(compare_class, train_score))

# IQR Method에서 경계값을 나타낸 함수
def iqr_outlier(ddff):
    q1 = ddff.quantile(0.25)
    q3 = ddff.quantile(0.75)

    iqr = q3 - q1

    lower_bound = q1 - (1.5 * iqr)
    upper_bound = q3 + (1.5 * iqr)

    return pd.concat([lower_bound, upper_bound], axis=1).T

```

Validation dataset의 통계정보를 이용한 1차로 변수 선택합니다. 테스트 데이터 중 랜덤 샘플된 것이므로 Validation dataset의 통계정보로 충분히 정보를 얻을 수 있습니다. 아래의 기준은 outlier의 중위값이 IQR method의 범위 내에 있는지 확인합니다. 저는 outlier의 중위값이 적어도 IQR method의 범위 밖에 있어야 outlier 판단하기 쉽다고 생각하였습니다.

```
In [3]: ## 기본적인 변수 선택(1)
def first_variation_selection(dfdfdf):
    # Validation dataset의 outlier들의 중앙값이 Validation dataset의 IQR Method의 범위 내에 있으므로 1개 제외
    # class 열도 있으므로 1개 제외
    how_many_var = (len(dfdfdf.columns) - 1)
    new_var = []

    for what_val in range(how_many_var):
        if (iqr_outlier(dfdfdf).iloc[0,what_val] < dfdfdf.iloc[np.where(dfdfdf['Class'] == 1)[0],what_val]):
            continue
        else:
            new_var.append(what_val)

    return new_var

superior_var1 = first_variation_selection(ori_val_df)
print(superior_var1)
```

```
[1, 2, 3, 6, 8, 9, 10, 11, 13, 15, 16, 17]
```

이번엔 2차 변수 선택입니다. 두 가지 변수 방법을 사용하여 결과물을 합집합화 하였습니다. 첫번째는 Wilcoxon rank-sum test를 이용해 Validation dataset의 outlier들의 중앙값과 Validation dataset의 inlier들의 중앙값의 차이를 검정해보고 p-value가 가장 낮은 5개 변수를 뽑았습니다. 두번째는 FRUFS의 LGBMRegressor을 이용하여 변수 중요도를 판단하였고 중요도가 가장 높은 5개 변수를 뽑았습니다.

```
In [4]: ## 기본적인 변수 선택(2)
# 2차 변수 선택
def second_variation_selection(dfdfdf, trtrtr, nnn_var):
    ranksum_pval = []
    for what_val in nnn_var:
        ranksum_pval.append(ranksums(dfdfdf.iloc[np.where(dfdfdf['Class'] == 1)], trtrtr))

    Wilcoxon_rank_sum_pval_var = list(pd.DataFrame({'pval':ranksum_pval, 'col':dfdfdf.columns}))
    print(Wilcoxon_rank_sum_pval_var)

    # FRUFS를 이용하여 변수 선택
    # core가 많으면 n_jobs 조정을 하면 됨.
    model_frufs = FRUFS(model_r=LGBMRegressor(random_state=28), k=5, n_jobs=-1)
    df_train_pruned = model_frufs.fit_transform(trtrtr.iloc[:,nnn_var])
    FRUFS_LGBMRegressor_var = list(df_train_pruned.columns)

    #plt.figure(figsize=(5, 6), dpi=100)
    #model_frufs.feature_importance()
    print(FRUFS_LGBMRegressor_var)

    # 종합
    all_var_in_td = list(trtrtr.columns)
    new_var_set = list(set(Wilcoxon_rank_sum_pval_var + FRUFS_LGBMRegressor_var))
    new_var_list = []
```

```

for nvs in new_var_set:
    new_var_list.append(all_var_in_td.index(nvs))

new_var_list.sort()
return new_var_list

superior_var2 = second_variation_selection(ori_val_df, train_df, superior_var1)
print(superior_var2)

['V10', 'V14', 'V11', 'V4', 'V12']

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 16 concurrent workers.
['V10', 'V4', 'V14', 'V17', 'V16']
[3, 9, 10, 11, 13, 15, 16]

[Parallel(n_jobs=-1)]: Done 7 out of 12 | elapsed: 10.7s remaining: 7.6s
[Parallel(n_jobs=-1)]: Done 12 out of 12 | elapsed: 10.8s finished

```

In [5]:

```

## 기본적인 변수 선택(3)
# 선택한 변수 제외 나머지 변수 모임
inferior_var2 = [x for x in range(30) if x not in superior_var2]
print(inferior_var2)

[0, 1, 2, 4, 5, 6, 7, 8, 12, 14, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]

```

Baseline에 보시면 아시겠지만 validation set의 사기 거래 비율을 탐색하였는데 저는 이보다 조금 높은 수치를 사용하였습니다.

In [6]:

```

## validation set의 사기 거래 비율 탐색
ori_val_normal, ori_val_fraud = ori_val_df['Class'].value_counts()
ori_val_contamination = ori_val_fraud / ori_val_normal
print(f'Validation contamination:[{ori_val_contamination}]')
# 이대로 하지 않고 조정을 함

```

Validation contamination:[0.0010551491277433877]

본격적으로 pacmap과 isolation forest를 이용하여 예측하고자 합니다. pacmap은 차원 축소를 하는 과정에 있어 랜덤하게 이동합니다. isolation forest도 랜덤하게 변수를 선택하기 때문에 결과가 불분명합니다. 따라서 저는 hhmm번 진행하고 voting을 통해 결과를 도출하였습니다.

In [7]:

```

## pacmap과 isolation forest 1차 이용(1)
# pacmap과 isolation forest를 이용한 1차 예측
hhmm = 3
what_val = superior_var2
for num in range(hhmm):
    embedding_1 = pacmap.PaCMAP(n_components=len(what_val), n_neighbors=None, MN
    pacmac_train_1 = embedding_1.fit_transform(np.array(train_df.iloc[:,what_val
    pacmac_val_1 = embedding_1.transform(np.array(val_df.iloc[:,what_val]), basi
    pacmac_test_1 = embedding_1.transform(np.array(test_df.iloc[:,what_val]), ba

    pac_model_1 = IsolationForest(n_estimators=1000, contamination=0.00121, verb
    pac_model_1.fit(pacmac_train_1[:,[1,2,3]])

    if num == 0:
        train_pred_set_1 = pac_model_1.predict(pacmac_train_1[:,[1,2,3]]) # mode
        train_pred_set_1 = get_pred_label(train_pred_set_1)

        val_pred_set_1 = pac_model_1.predict(pacmac_val_1[:,[1,2,3]]) # model pr

```

```
val_pred_set_1 = get_pred_label(val_pred_set_1)

test_pred_set_1 = pac_model_1.predict(pacmac_test_1[:,[1,2,3]]) # model
test_pred_set_1 = get_pred_label(test_pred_set_1)
else:
    train_pred_1 = pac_model_1.predict(pacmac_train_1[:,[1,2,3]]) # model pr
    train_pred_1 = get_pred_label(train_pred_1)
    train_pred_set_1 = train_pred_set_1 + train_pred_1

    val_pred_1 = pac_model_1.predict(pacmac_val_1[:,[1,2,3]]) # model predic
    val_pred_1 = get_pred_label(val_pred_1)
    val_pred_set_1 = val_pred_set_1 + val_pred_1

    test_pred_1 = pac_model_1.predict(pacmac_test_1[:,[1,2,3]]) # model prea
    test_pred_1 = get_pred_label(test_pred_1)
    test_pred_set_1 = test_pred_set_1 + test_pred_1

train_pred_set_1 = train_pred_set_1/hhmm
val_pred_set_1 = val_pred_set_1/hhmm
test_pred_set_1 = test_pred_set_1/hhmm
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 1000), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2402245.000000
Iteration: 20, Loss: 2148587.750000
Iteration: 30, Loss: 2014521.750000
Iteration: 40, Loss: 1910488.750000
Iteration: 50, Loss: 1811245.750000
Iteration: 60, Loss: 1706021.750000
Iteration: 70, Loss: 1587754.500000
Iteration: 80, Loss: 1447690.750000
Iteration: 90, Loss: 1271229.000000
Iteration: 100, Loss: 1005272.125000
Iteration: 110, Loss: 1315405.500000
Iteration: 120, Loss: 1292878.500000
Iteration: 130, Loss: 1283322.250000
Iteration: 140, Loss: 1280210.125000
Iteration: 150, Loss: 1279602.750000
Iteration: 160, Loss: 1279720.875000
Iteration: 170, Loss: 1279834.750000
Iteration: 180, Loss: 1280553.875000
Iteration: 190, Loss: 1281236.125000
Iteration: 200, Loss: 1281799.625000
Iteration: 210, Loss: 539752.312500
Iteration: 220, Loss: 534655.500000
Iteration: 230, Loss: 530014.250000
Iteration: 240, Loss: 527621.125000
Iteration: 250, Loss: 525673.250000
Iteration: 260, Loss: 524034.125000
Iteration: 270, Loss: 522689.343750
Iteration: 280, Loss: 521724.093750
Iteration: 290, Loss: 520822.875000
Iteration: 300, Loss: 519876.812500
Iteration: 310, Loss: 518907.187500
Iteration: 320, Loss: 517942.812500
Iteration: 330, Loss: 517056.031250
Iteration: 340, Loss: 516300.062500
Iteration: 350, Loss: 515683.625000
Iteration: 360, Loss: 515215.937500
Iteration: 370, Loss: 514868.687500
Iteration: 380, Loss: 514606.093750
Iteration: 390, Loss: 514433.937500
Iteration: 400, Loss: 514293.062500
Iteration: 410, Loss: 514151.937500
Iteration: 420, Loss: 513987.687500
Iteration: 430, Loss: 513796.312500
Iteration: 440, Loss: 513574.687500
Iteration: 450, Loss: 513352.125000
Iteration: 460, Loss: 513130.687500
Iteration: 470, Loss: 512925.843750
Iteration: 480, Loss: 512739.250000
Iteration: 490, Loss: 512572.031250
```

Iteration: 500, Loss: 512405.062500
Iteration: 510, Loss: 512237.250000
Iteration: 520, Loss: 512067.562500
Iteration: 530, Loss: 511908.531250
Iteration: 540, Loss: 511762.875000
Iteration: 550, Loss: 511631.843750
Iteration: 560, Loss: 511510.062500
Iteration: 570, Loss: 511397.375000
Iteration: 580, Loss: 511303.781250
Iteration: 590, Loss: 511225.750000
Iteration: 600, Loss: 511157.250000
Iteration: 610, Loss: 511101.187500
Iteration: 620, Loss: 511056.312500
Iteration: 630, Loss: 511024.843750
Iteration: 640, Loss: 511006.687500
Iteration: 650, Loss: 510996.000000
Iteration: 660, Loss: 510989.937500
Iteration: 670, Loss: 510984.750000
Iteration: 680, Loss: 510980.156250
Iteration: 690, Loss: 510982.812500
Iteration: 700, Loss: 510988.687500
Iteration: 710, Loss: 510991.093750
Iteration: 720, Loss: 510995.093750
Iteration: 730, Loss: 511005.312500
Iteration: 740, Loss: 511016.968750
Iteration: 750, Loss: 511028.968750
Iteration: 760, Loss: 511042.187500
Iteration: 770, Loss: 511055.000000
Iteration: 780, Loss: 511056.500000
Iteration: 790, Loss: 511053.750000
Iteration: 800, Loss: 511048.750000
Iteration: 810, Loss: 511043.406250
Iteration: 820, Loss: 511036.625000
Iteration: 830, Loss: 511033.437500
Iteration: 840, Loss: 511029.843750
Iteration: 850, Loss: 511028.031250
Iteration: 860, Loss: 511022.718750
Iteration: 870, Loss: 511014.968750
Iteration: 880, Loss: 511006.250000
Iteration: 890, Loss: 510994.156250
Iteration: 900, Loss: 510979.375000
Iteration: 910, Loss: 510961.843750
Iteration: 920, Loss: 510941.468750
Iteration: 930, Loss: 510916.156250
Iteration: 940, Loss: 510888.968750
Iteration: 950, Loss: 510859.656250
Iteration: 960, Loss: 510829.187500
Iteration: 970, Loss: 510797.062500
Iteration: 980, Loss: 510765.718750
Iteration: 990, Loss: 510733.812500
Iteration: 1000, Loss: 510702.781250
Iteration: 1010, Loss: 510672.843750
Iteration: 1020, Loss: 510642.437500
Iteration: 1030, Loss: 510613.562500
Iteration: 1040, Loss: 510585.343750
Iteration: 1050, Loss: 510558.781250
Iteration: 1060, Loss: 510532.531250
Iteration: 1070, Loss: 510508.843750
Iteration: 1080, Loss: 510487.562500
Iteration: 1090, Loss: 510466.687500

```
Iteration: 1100, Loss: 510452.906250
Iteration: 1110, Loss: 510453.625000
Iteration: 1120, Loss: 510508.781250
Iteration: 1130, Loss: 510713.031250
Iteration: 1140, Loss: 510853.187500
Iteration: 1150, Loss: 510930.187500
Iteration: 1160, Loss: 510938.406250
Iteration: 1170, Loss: 510912.906250
Iteration: 1180, Loss: 510955.343750
Iteration: 1190, Loss: 511026.312500
Iteration: 1200, Loss: 511163.437500
Elapsed time: 294.41s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1407240.75
Iteration: 10, Loss: 694445.500000
Iteration: 20, Loss: 400909.312500
Iteration: 30, Loss: 261723.062500
Iteration: 40, Loss: 226566.406250
Iteration: 50, Loss: 211543.265625
Iteration: 60, Loss: 206841.687500
Iteration: 70, Loss: 205440.796875
Iteration: 80, Loss: 204775.156250
Iteration: 90, Loss: 204594.625000
Iteration: 100, Loss: 204521.671875
Iteration: 110, Loss: 307940.968750
Iteration: 120, Loss: 307255.750000
Iteration: 130, Loss: 307064.562500
Iteration: 140, Loss: 307037.812500
Iteration: 150, Loss: 307038.187500
Iteration: 160, Loss: 307025.125000
Iteration: 170, Loss: 307017.718750
Iteration: 180, Loss: 307010.750000
Iteration: 190, Loss: 307024.750000
Iteration: 200, Loss: 307047.531250
Iteration: 210, Loss: 102321.789062
Iteration: 220, Loss: 102257.695312
Iteration: 230, Loss: 102243.562500
Iteration: 240, Loss: 102238.789062
Iteration: 250, Loss: 102237.718750
Iteration: 260, Loss: 102236.710938
Iteration: 270, Loss: 102236.296875
Iteration: 280, Loss: 102236.015625
Iteration: 290, Loss: 102236.046875
Iteration: 300, Loss: 102235.890625
Iteration: 310, Loss: 102235.812500
Iteration: 320, Loss: 102235.703125
Iteration: 330, Loss: 102235.757812
Iteration: 340, Loss: 102235.875000
Iteration: 350, Loss: 102235.835938
Iteration: 360, Loss: 102235.796875
Iteration: 370, Loss: 102235.804688
Iteration: 380, Loss: 102235.789062
Iteration: 390, Loss: 102235.789062
Iteration: 400, Loss: 102235.789062
Iteration: 410, Loss: 102235.804688
Iteration: 420, Loss: 102235.804688
Iteration: 430, Loss: 102235.804688
```

Iteration: 440, Loss: 102235.812500
Iteration: 450, Loss: 102235.812500
Iteration: 460, Loss: 102235.804688
Iteration: 470, Loss: 102235.804688
Iteration: 480, Loss: 102235.812500
Iteration: 490, Loss: 102235.812500
Iteration: 500, Loss: 102235.820312
Iteration: 510, Loss: 102235.804688
Iteration: 520, Loss: 102235.804688
Iteration: 530, Loss: 102235.804688
Iteration: 540, Loss: 102235.804688
Iteration: 550, Loss: 102235.812500
Iteration: 560, Loss: 102235.812500
Iteration: 570, Loss: 102235.812500
Iteration: 580, Loss: 102235.804688
Iteration: 590, Loss: 102235.804688
Iteration: 600, Loss: 102235.804688
Iteration: 610, Loss: 102235.804688
Iteration: 620, Loss: 102235.796875
Iteration: 630, Loss: 102235.796875
Iteration: 640, Loss: 102235.804688
Iteration: 650, Loss: 102235.812500
Iteration: 660, Loss: 102235.796875
Iteration: 670, Loss: 102235.796875
Iteration: 680, Loss: 102235.804688
Iteration: 690, Loss: 102235.804688
Iteration: 700, Loss: 102235.804688
Iteration: 710, Loss: 102235.804688
Iteration: 720, Loss: 102235.804688
Iteration: 730, Loss: 102235.796875
Iteration: 740, Loss: 102235.804688
Iteration: 750, Loss: 102235.804688
Iteration: 760, Loss: 102235.796875
Iteration: 770, Loss: 102235.796875
Iteration: 780, Loss: 102235.804688
Iteration: 790, Loss: 102235.812500
Iteration: 800, Loss: 102235.804688
Iteration: 810, Loss: 102235.804688
Iteration: 820, Loss: 102235.804688
Iteration: 830, Loss: 102235.804688
Iteration: 840, Loss: 102235.796875
Iteration: 850, Loss: 102235.796875
Iteration: 860, Loss: 102235.804688
Iteration: 870, Loss: 102235.804688
Iteration: 880, Loss: 102235.804688
Iteration: 890, Loss: 102235.796875
Iteration: 900, Loss: 102235.789062
Iteration: 910, Loss: 102235.796875
Iteration: 920, Loss: 102235.812500
Iteration: 930, Loss: 102235.796875
Iteration: 940, Loss: 102235.796875
Iteration: 950, Loss: 102235.804688
Iteration: 960, Loss: 102235.796875
Iteration: 970, Loss: 102235.796875
Iteration: 980, Loss: 102235.804688
Iteration: 990, Loss: 102235.804688
Iteration: 1000, Loss: 102235.796875
Iteration: 1010, Loss: 102235.804688
Iteration: 1020, Loss: 102235.812500
Iteration: 1030, Loss: 102235.804688

```
Iteration: 1040, Loss: 102235.789062
Iteration: 1050, Loss: 102236.015625
Iteration: 1060, Loss: 102236.679688
Iteration: 1070, Loss: 102238.273438
Iteration: 1080, Loss: 102244.578125
Iteration: 1090, Loss: 102252.921875
Iteration: 1100, Loss: 102264.023438
Iteration: 1110, Loss: 102278.570312
Iteration: 1120, Loss: 102302.179688
Iteration: 1130, Loss: 102331.796875
Iteration: 1140, Loss: 102355.335938
Iteration: 1150, Loss: 102368.390625
Iteration: 1160, Loss: 102371.234375
Iteration: 1170, Loss: 102363.765625
Iteration: 1180, Loss: 102356.992188
Iteration: 1190, Loss: 102350.132812
Iteration: 1200, Loss: 102345.585938
Elapsed time: 0:00:18.635549
X is normalized.
X is normalized.
Found nearest neighbor
(3705078, 2)
Initial Loss: 7154854.0
Iteration: 10, Loss: 3475613.250000
Iteration: 20, Loss: 2006372.500000
Iteration: 30, Loss: 1315405.875000
Iteration: 40, Loss: 1135492.750000
Iteration: 50, Loss: 1055844.000000
Iteration: 60, Loss: 1030391.812500
Iteration: 70, Loss: 1024440.062500
Iteration: 80, Loss: 1022169.687500
Iteration: 90, Loss: 1021563.437500
Iteration: 100, Loss: 1021319.062500
Iteration: 110, Loss: 1524250.000000
Iteration: 120, Loss: 1519830.625000
Iteration: 130, Loss: 1518763.500000
Iteration: 140, Loss: 1518641.500000
Iteration: 150, Loss: 1518639.875000
Iteration: 160, Loss: 1518590.625000
Iteration: 170, Loss: 1518545.625000
Iteration: 180, Loss: 1518577.625000
Iteration: 190, Loss: 1518648.875000
Iteration: 200, Loss: 1518704.375000
Iteration: 210, Loss: 510897.468750
Iteration: 220, Loss: 510651.562500
Iteration: 230, Loss: 510603.593750
Iteration: 240, Loss: 510586.781250
Iteration: 250, Loss: 510582.468750
Iteration: 260, Loss: 510582.000000
Iteration: 270, Loss: 510579.593750
Iteration: 280, Loss: 510579.468750
Iteration: 290, Loss: 510578.281250
Iteration: 300, Loss: 510578.281250
Iteration: 310, Loss: 510578.343750
Iteration: 320, Loss: 510577.968750
Iteration: 330, Loss: 510578.343750
Iteration: 340, Loss: 510578.593750
Iteration: 350, Loss: 510578.968750
Iteration: 360, Loss: 510578.593750
Iteration: 370, Loss: 510578.875000
```

Iteration: 380, Loss: 510578.656250
Iteration: 390, Loss: 510578.656250
Iteration: 400, Loss: 510578.687500
Iteration: 410, Loss: 510578.687500
Iteration: 420, Loss: 510578.593750
Iteration: 430, Loss: 510578.656250
Iteration: 440, Loss: 510578.625000
Iteration: 450, Loss: 510578.687500
Iteration: 460, Loss: 510578.625000
Iteration: 470, Loss: 510578.718750
Iteration: 480, Loss: 510578.593750
Iteration: 490, Loss: 510578.656250
Iteration: 500, Loss: 510578.656250
Iteration: 510, Loss: 510578.687500
Iteration: 520, Loss: 510578.656250
Iteration: 530, Loss: 510578.687500
Iteration: 540, Loss: 510578.625000
Iteration: 550, Loss: 510578.625000
Iteration: 560, Loss: 510578.656250
Iteration: 570, Loss: 510578.687500
Iteration: 580, Loss: 510578.656250
Iteration: 590, Loss: 510578.593750
Iteration: 600, Loss: 510578.687500
Iteration: 610, Loss: 510578.656250
Iteration: 620, Loss: 510578.687500
Iteration: 630, Loss: 510578.593750
Iteration: 640, Loss: 510578.687500
Iteration: 650, Loss: 510578.718750
Iteration: 660, Loss: 510578.687500
Iteration: 670, Loss: 510578.750000
Iteration: 680, Loss: 510578.718750
Iteration: 690, Loss: 510578.625000
Iteration: 700, Loss: 510578.687500
Iteration: 710, Loss: 510578.687500
Iteration: 720, Loss: 510578.687500
Iteration: 730, Loss: 510578.687500
Iteration: 740, Loss: 510578.687500
Iteration: 750, Loss: 510578.687500
Iteration: 760, Loss: 510578.656250
Iteration: 770, Loss: 510578.656250
Iteration: 780, Loss: 510578.718750
Iteration: 790, Loss: 510578.656250
Iteration: 800, Loss: 510578.625000
Iteration: 810, Loss: 510578.656250
Iteration: 820, Loss: 510578.687500
Iteration: 830, Loss: 510578.718750
Iteration: 840, Loss: 510578.687500
Iteration: 850, Loss: 510578.687500
Iteration: 860, Loss: 510578.625000
Iteration: 870, Loss: 510578.687500
Iteration: 880, Loss: 510578.656250
Iteration: 890, Loss: 510578.687500
Iteration: 900, Loss: 510578.718750
Iteration: 910, Loss: 510578.687500
Iteration: 920, Loss: 510578.656250
Iteration: 930, Loss: 510578.687500
Iteration: 940, Loss: 510578.625000
Iteration: 950, Loss: 510578.656250
Iteration: 960, Loss: 510578.656250
Iteration: 970, Loss: 510578.687500

```
Iteration: 980, Loss: 510578.656250
Iteration: 990, Loss: 510578.625000
Iteration: 1000, Loss: 510578.718750
Iteration: 1010, Loss: 510578.656250
Iteration: 1020, Loss: 510578.718750
Iteration: 1030, Loss: 510578.843750
Iteration: 1040, Loss: 510578.906250
Iteration: 1050, Loss: 510579.093750
Iteration: 1060, Loss: 510581.625000
Iteration: 1070, Loss: 510588.843750
Iteration: 1080, Loss: 510615.531250
Iteration: 1090, Loss: 510642.625000
Iteration: 1100, Loss: 510677.125000
Iteration: 1110, Loss: 510733.343750
Iteration: 1120, Loss: 510828.250000
Iteration: 1130, Loss: 510929.906250
Iteration: 1140, Loss: 511016.187500
Iteration: 1150, Loss: 511085.781250
Iteration: 1160, Loss: 511109.437500
Iteration: 1170, Loss: 511091.062500
Iteration: 1180, Loss: 511057.312500
Iteration: 1190, Loss: 511026.875000
Iteration: 1200, Loss: 510997.531250
Elapsed time: 0:01:22.373028
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 1000), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2402752.000000
Iteration: 20, Loss: 2148467.500000
Iteration: 30, Loss: 2014797.750000
Iteration: 40, Loss: 1910795.000000
Iteration: 50, Loss: 1811478.500000
Iteration: 60, Loss: 1706067.000000
Iteration: 70, Loss: 1587951.750000
Iteration: 80, Loss: 1447679.250000
Iteration: 90, Loss: 1271184.250000
Iteration: 100, Loss: 1005147.812500
Iteration: 110, Loss: 1315296.500000
Iteration: 120, Loss: 1292914.000000
Iteration: 130, Loss: 1283412.000000
Iteration: 140, Loss: 1280270.750000
Iteration: 150, Loss: 1279687.750000
Iteration: 160, Loss: 1279834.000000
Iteration: 170, Loss: 1279905.250000
Iteration: 180, Loss: 1280610.500000
Iteration: 190, Loss: 1281425.000000
Iteration: 200, Loss: 1281896.125000
Iteration: 210, Loss: 539806.437500
Iteration: 220, Loss: 534731.187500
Iteration: 230, Loss: 530122.562500
Iteration: 240, Loss: 527704.437500
Iteration: 250, Loss: 525754.375000
Iteration: 260, Loss: 524067.250000
Iteration: 270, Loss: 522719.750000
Iteration: 280, Loss: 521741.468750
Iteration: 290, Loss: 520844.375000
Iteration: 300, Loss: 519896.468750
Iteration: 310, Loss: 518918.406250
Iteration: 320, Loss: 517953.062500
Iteration: 330, Loss: 517076.093750
Iteration: 340, Loss: 516315.625000
Iteration: 350, Loss: 515721.281250
Iteration: 360, Loss: 515279.625000
Iteration: 370, Loss: 514926.937500
Iteration: 380, Loss: 514675.062500
Iteration: 390, Loss: 514506.906250
Iteration: 400, Loss: 514366.500000
Iteration: 410, Loss: 514215.343750
Iteration: 420, Loss: 514050.593750
Iteration: 430, Loss: 513847.156250
Iteration: 440, Loss: 513617.687500
Iteration: 450, Loss: 513390.312500
Iteration: 460, Loss: 513174.250000
Iteration: 470, Loss: 512975.562500
Iteration: 480, Loss: 512797.625000
Iteration: 490, Loss: 512630.000000
```

Iteration: 500, Loss: 512469.843750
Iteration: 510, Loss: 512303.281250
Iteration: 520, Loss: 512136.406250
Iteration: 530, Loss: 511977.125000
Iteration: 540, Loss: 511832.750000
Iteration: 550, Loss: 511700.000000
Iteration: 560, Loss: 511581.125000
Iteration: 570, Loss: 511472.312500
Iteration: 580, Loss: 511379.312500
Iteration: 590, Loss: 511301.968750
Iteration: 600, Loss: 511233.531250
Iteration: 610, Loss: 511179.843750
Iteration: 620, Loss: 511136.062500
Iteration: 630, Loss: 511105.968750
Iteration: 640, Loss: 511088.250000
Iteration: 650, Loss: 511076.968750
Iteration: 660, Loss: 511069.187500
Iteration: 670, Loss: 511064.875000
Iteration: 680, Loss: 511059.937500
Iteration: 690, Loss: 511064.625000
Iteration: 700, Loss: 511069.437500
Iteration: 710, Loss: 511072.937500
Iteration: 720, Loss: 511077.375000
Iteration: 730, Loss: 511089.593750
Iteration: 740, Loss: 511100.625000
Iteration: 750, Loss: 511118.062500
Iteration: 760, Loss: 511130.937500
Iteration: 770, Loss: 511143.625000
Iteration: 780, Loss: 511145.625000
Iteration: 790, Loss: 511141.625000
Iteration: 800, Loss: 511135.562500
Iteration: 810, Loss: 511130.312500
Iteration: 820, Loss: 511125.156250
Iteration: 830, Loss: 511120.093750
Iteration: 840, Loss: 511117.312500
Iteration: 850, Loss: 511114.687500
Iteration: 860, Loss: 511109.281250
Iteration: 870, Loss: 511100.812500
Iteration: 880, Loss: 511091.281250
Iteration: 890, Loss: 511078.218750
Iteration: 900, Loss: 511062.593750
Iteration: 910, Loss: 511043.937500
Iteration: 920, Loss: 511021.968750
Iteration: 930, Loss: 510997.906250
Iteration: 940, Loss: 510971.031250
Iteration: 950, Loss: 510941.562500
Iteration: 960, Loss: 510911.093750
Iteration: 970, Loss: 510879.218750
Iteration: 980, Loss: 510848.000000
Iteration: 990, Loss: 510816.656250
Iteration: 1000, Loss: 510786.593750
Iteration: 1010, Loss: 510755.968750
Iteration: 1020, Loss: 510726.000000
Iteration: 1030, Loss: 510697.406250
Iteration: 1040, Loss: 510669.406250
Iteration: 1050, Loss: 510642.593750
Iteration: 1060, Loss: 510616.593750
Iteration: 1070, Loss: 510593.687500
Iteration: 1080, Loss: 510573.343750
Iteration: 1090, Loss: 510553.312500

```
Iteration: 1100, Loss: 510538.625000
Iteration: 1110, Loss: 510537.156250
Iteration: 1120, Loss: 510596.906250
Iteration: 1130, Loss: 510776.500000
Iteration: 1140, Loss: 510886.437500
Iteration: 1150, Loss: 510995.906250
Iteration: 1160, Loss: 511017.843750
Iteration: 1170, Loss: 510998.000000
Iteration: 1180, Loss: 511032.156250
Iteration: 1190, Loss: 511118.468750
Iteration: 1200, Loss: 511207.937500
Elapsed time: 299.30s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1407038.0
Iteration: 10, Loss: 693347.062500
Iteration: 20, Loss: 402442.781250
Iteration: 30, Loss: 261006.265625
Iteration: 40, Loss: 226782.421875
Iteration: 50, Loss: 211539.062500
Iteration: 60, Loss: 206744.546875
Iteration: 70, Loss: 205379.156250
Iteration: 80, Loss: 204701.765625
Iteration: 90, Loss: 204521.500000
Iteration: 100, Loss: 204448.312500
Iteration: 110, Loss: 307852.718750
Iteration: 120, Loss: 307155.906250
Iteration: 130, Loss: 306962.718750
Iteration: 140, Loss: 306928.750000
Iteration: 150, Loss: 306906.906250
Iteration: 160, Loss: 306912.093750
Iteration: 170, Loss: 306896.500000
Iteration: 180, Loss: 306899.937500
Iteration: 190, Loss: 306910.406250
Iteration: 200, Loss: 306931.937500
Iteration: 210, Loss: 102282.070312
Iteration: 220, Loss: 102220.351562
Iteration: 230, Loss: 102206.882812
Iteration: 240, Loss: 102202.328125
Iteration: 250, Loss: 102200.382812
Iteration: 260, Loss: 102199.437500
Iteration: 270, Loss: 102199.343750
Iteration: 280, Loss: 102199.367188
Iteration: 290, Loss: 102199.328125
Iteration: 300, Loss: 102199.375000
Iteration: 310, Loss: 102199.265625
Iteration: 320, Loss: 102199.265625
Iteration: 330, Loss: 102199.281250
Iteration: 340, Loss: 102199.257812
Iteration: 350, Loss: 102199.289062
Iteration: 360, Loss: 102199.281250
Iteration: 370, Loss: 102199.289062
Iteration: 380, Loss: 102199.304688
Iteration: 390, Loss: 102199.273438
Iteration: 400, Loss: 102199.296875
Iteration: 410, Loss: 102199.289062
Iteration: 420, Loss: 102199.296875
Iteration: 430, Loss: 102199.296875
```

Iteration: 440, Loss: 102199.281250
Iteration: 450, Loss: 102199.304688
Iteration: 460, Loss: 102199.289062
Iteration: 470, Loss: 102199.296875
Iteration: 480, Loss: 102199.296875
Iteration: 490, Loss: 102199.296875
Iteration: 500, Loss: 102199.289062
Iteration: 510, Loss: 102199.296875
Iteration: 520, Loss: 102199.296875
Iteration: 530, Loss: 102199.304688
Iteration: 540, Loss: 102199.289062
Iteration: 550, Loss: 102199.296875
Iteration: 560, Loss: 102199.289062
Iteration: 570, Loss: 102199.289062
Iteration: 580, Loss: 102199.289062
Iteration: 590, Loss: 102199.296875
Iteration: 600, Loss: 102199.296875
Iteration: 610, Loss: 102199.296875
Iteration: 620, Loss: 102199.289062
Iteration: 630, Loss: 102199.281250
Iteration: 640, Loss: 102199.296875
Iteration: 650, Loss: 102199.289062
Iteration: 660, Loss: 102199.289062
Iteration: 670, Loss: 102199.289062
Iteration: 680, Loss: 102199.296875
Iteration: 690, Loss: 102199.289062
Iteration: 700, Loss: 102199.289062
Iteration: 710, Loss: 102199.289062
Iteration: 720, Loss: 102199.304688
Iteration: 730, Loss: 102199.289062
Iteration: 740, Loss: 102199.296875
Iteration: 750, Loss: 102199.296875
Iteration: 760, Loss: 102199.289062
Iteration: 770, Loss: 102199.296875
Iteration: 780, Loss: 102199.273438
Iteration: 790, Loss: 102199.289062
Iteration: 800, Loss: 102199.304688
Iteration: 810, Loss: 102199.289062
Iteration: 820, Loss: 102199.289062
Iteration: 830, Loss: 102199.296875
Iteration: 840, Loss: 102199.304688
Iteration: 850, Loss: 102199.296875
Iteration: 860, Loss: 102199.289062
Iteration: 870, Loss: 102199.289062
Iteration: 880, Loss: 102199.281250
Iteration: 890, Loss: 102199.296875
Iteration: 900, Loss: 102199.312500
Iteration: 910, Loss: 102199.289062
Iteration: 920, Loss: 102199.296875
Iteration: 930, Loss: 102199.289062
Iteration: 940, Loss: 102199.281250
Iteration: 950, Loss: 102199.289062
Iteration: 960, Loss: 102199.296875
Iteration: 970, Loss: 102199.296875
Iteration: 980, Loss: 102199.296875
Iteration: 990, Loss: 102199.296875
Iteration: 1000, Loss: 102199.296875
Iteration: 1010, Loss: 102199.281250
Iteration: 1020, Loss: 102199.296875
Iteration: 1030, Loss: 102199.296875

```
Iteration: 1040, Loss: 102199.320312
Iteration: 1050, Loss: 102199.492188
Iteration: 1060, Loss: 102200.101562
Iteration: 1070, Loss: 102201.992188
Iteration: 1080, Loss: 102209.093750
Iteration: 1090, Loss: 102217.781250
Iteration: 1100, Loss: 102227.195312
Iteration: 1110, Loss: 102243.664062
Iteration: 1120, Loss: 102267.070312
Iteration: 1130, Loss: 102297.992188
Iteration: 1140, Loss: 102319.968750
Iteration: 1150, Loss: 102329.140625
Iteration: 1160, Loss: 102331.148438
Iteration: 1170, Loss: 102330.726562
Iteration: 1180, Loss: 102325.421875
Iteration: 1190, Loss: 102317.195312
Iteration: 1200, Loss: 102308.054688
Elapsed time: 0:00:18.509617
X is normalized.
X is normalized.
Found nearest neighbor
(3705078, 2)
Initial Loss: 7152665.5
Iteration: 10, Loss: 3470081.500000
Iteration: 20, Loss: 2015265.875000
Iteration: 30, Loss: 1312612.000000
Iteration: 40, Loss: 1136841.625000
Iteration: 50, Loss: 1056041.500000
Iteration: 60, Loss: 1030129.437500
Iteration: 70, Loss: 1024320.062500
Iteration: 80, Loss: 1022036.187500
Iteration: 90, Loss: 1021422.937500
Iteration: 100, Loss: 1021208.312500
Iteration: 110, Loss: 1524337.750000
Iteration: 120, Loss: 1519566.000000
Iteration: 130, Loss: 1518517.875000
Iteration: 140, Loss: 1518337.625000
Iteration: 150, Loss: 1518353.750000
Iteration: 160, Loss: 1518263.625000
Iteration: 170, Loss: 1518208.875000
Iteration: 180, Loss: 1518125.375000
Iteration: 190, Loss: 1518206.500000
Iteration: 200, Loss: 1518354.125000
Iteration: 210, Loss: 510817.875000
Iteration: 220, Loss: 510598.000000
Iteration: 230, Loss: 510546.718750
Iteration: 240, Loss: 510533.125000
Iteration: 250, Loss: 510523.812500
Iteration: 260, Loss: 510522.187500
Iteration: 270, Loss: 510520.125000
Iteration: 280, Loss: 510520.718750
Iteration: 290, Loss: 510520.531250
Iteration: 300, Loss: 510520.937500
Iteration: 310, Loss: 510519.843750
Iteration: 320, Loss: 510519.687500
Iteration: 330, Loss: 510519.406250
Iteration: 340, Loss: 510519.500000
Iteration: 350, Loss: 510519.281250
Iteration: 360, Loss: 510519.312500
Iteration: 370, Loss: 510519.312500
```

Iteration: 380, Loss: 510519.250000
Iteration: 390, Loss: 510519.156250
Iteration: 400, Loss: 510519.250000
Iteration: 410, Loss: 510519.281250
Iteration: 420, Loss: 510519.250000
Iteration: 430, Loss: 510519.250000
Iteration: 440, Loss: 510519.218750
Iteration: 450, Loss: 510519.187500
Iteration: 460, Loss: 510519.218750
Iteration: 470, Loss: 510519.250000
Iteration: 480, Loss: 510519.250000
Iteration: 490, Loss: 510519.250000
Iteration: 500, Loss: 510519.156250
Iteration: 510, Loss: 510519.281250
Iteration: 520, Loss: 510519.218750
Iteration: 530, Loss: 510519.312500
Iteration: 540, Loss: 510519.281250
Iteration: 550, Loss: 510519.312500
Iteration: 560, Loss: 510519.281250
Iteration: 570, Loss: 510519.250000
Iteration: 580, Loss: 510519.281250
Iteration: 590, Loss: 510519.250000
Iteration: 600, Loss: 510519.312500
Iteration: 610, Loss: 510519.281250
Iteration: 620, Loss: 510519.281250
Iteration: 630, Loss: 510519.312500
Iteration: 640, Loss: 510519.312500
Iteration: 650, Loss: 510519.312500
Iteration: 660, Loss: 510519.281250
Iteration: 670, Loss: 510519.218750
Iteration: 680, Loss: 510519.281250
Iteration: 690, Loss: 510519.281250
Iteration: 700, Loss: 510519.250000
Iteration: 710, Loss: 510519.281250
Iteration: 720, Loss: 510519.281250
Iteration: 730, Loss: 510519.281250
Iteration: 740, Loss: 510519.250000
Iteration: 750, Loss: 510519.218750
Iteration: 760, Loss: 510519.218750
Iteration: 770, Loss: 510519.281250
Iteration: 780, Loss: 510519.281250
Iteration: 790, Loss: 510519.250000
Iteration: 800, Loss: 510519.250000
Iteration: 810, Loss: 510519.281250
Iteration: 820, Loss: 510519.250000
Iteration: 830, Loss: 510519.250000
Iteration: 840, Loss: 510519.281250
Iteration: 850, Loss: 510519.250000
Iteration: 860, Loss: 510519.281250
Iteration: 870, Loss: 510519.250000
Iteration: 880, Loss: 510519.250000
Iteration: 890, Loss: 510519.218750
Iteration: 900, Loss: 510519.250000
Iteration: 910, Loss: 510519.281250
Iteration: 920, Loss: 510519.281250
Iteration: 930, Loss: 510519.281250
Iteration: 940, Loss: 510519.250000
Iteration: 950, Loss: 510519.281250
Iteration: 960, Loss: 510519.250000
Iteration: 970, Loss: 510519.281250

```
Iteration: 980, Loss: 510519.218750
Iteration: 990, Loss: 510519.250000
Iteration: 1000, Loss: 510519.218750
Iteration: 1010, Loss: 510519.218750
Iteration: 1020, Loss: 510519.187500
Iteration: 1030, Loss: 510519.406250
Iteration: 1040, Loss: 510519.437500
Iteration: 1050, Loss: 510520.187500
Iteration: 1060, Loss: 510523.218750
Iteration: 1070, Loss: 510531.625000
Iteration: 1080, Loss: 510557.031250
Iteration: 1090, Loss: 510583.156250
Iteration: 1100, Loss: 510616.187500
Iteration: 1110, Loss: 510679.593750
Iteration: 1120, Loss: 510762.843750
Iteration: 1130, Loss: 510872.281250
Iteration: 1140, Loss: 510966.250000
Iteration: 1150, Loss: 511027.125000
Iteration: 1160, Loss: 511045.593750
Iteration: 1170, Loss: 511025.625000
Iteration: 1180, Loss: 510998.875000
Iteration: 1190, Loss: 510970.000000
Iteration: 1200, Loss: 510940.718750
Elapsed time: 0:01:22.375509
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 1000), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2402950.750000
Iteration: 20, Loss: 2148642.500000
Iteration: 30, Loss: 2014852.000000
Iteration: 40, Loss: 1910849.500000
Iteration: 50, Loss: 1811382.500000
Iteration: 60, Loss: 1706113.125000
Iteration: 70, Loss: 1587824.000000
Iteration: 80, Loss: 1447700.750000
Iteration: 90, Loss: 1271206.000000
Iteration: 100, Loss: 1005083.125000
Iteration: 110, Loss: 1315143.125000
Iteration: 120, Loss: 1292623.250000
Iteration: 130, Loss: 1283149.125000
Iteration: 140, Loss: 1279961.000000
Iteration: 150, Loss: 1279464.875000
Iteration: 160, Loss: 1279524.750000
Iteration: 170, Loss: 1279671.125000
Iteration: 180, Loss: 1280368.500000
Iteration: 190, Loss: 1281180.875000
Iteration: 200, Loss: 1281620.750000
Iteration: 210, Loss: 539679.750000
Iteration: 220, Loss: 534569.000000
Iteration: 230, Loss: 529933.562500
Iteration: 240, Loss: 527534.687500
Iteration: 250, Loss: 525581.812500
Iteration: 260, Loss: 523897.875000
Iteration: 270, Loss: 522570.750000
Iteration: 280, Loss: 521599.250000
Iteration: 290, Loss: 520691.687500
Iteration: 300, Loss: 519756.843750
Iteration: 310, Loss: 518789.875000
Iteration: 320, Loss: 517816.562500
Iteration: 330, Loss: 516935.500000
Iteration: 340, Loss: 516177.437500
Iteration: 350, Loss: 515562.000000
Iteration: 360, Loss: 515099.812500
Iteration: 370, Loss: 514755.406250
Iteration: 380, Loss: 514501.218750
Iteration: 390, Loss: 514333.187500
Iteration: 400, Loss: 514185.937500
Iteration: 410, Loss: 514036.468750
Iteration: 420, Loss: 513867.312500
Iteration: 430, Loss: 513666.593750
Iteration: 440, Loss: 513443.062500
Iteration: 450, Loss: 513218.031250
Iteration: 460, Loss: 513002.593750
Iteration: 470, Loss: 512806.125000
Iteration: 480, Loss: 512633.500000
Iteration: 490, Loss: 512467.437500
```

Iteration: 500, Loss: 512309.781250
Iteration: 510, Loss: 512147.625000
Iteration: 520, Loss: 511982.687500
Iteration: 530, Loss: 511824.125000
Iteration: 540, Loss: 511679.562500
Iteration: 550, Loss: 511545.625000
Iteration: 560, Loss: 511422.437500
Iteration: 570, Loss: 511312.468750
Iteration: 580, Loss: 511218.625000
Iteration: 590, Loss: 511138.812500
Iteration: 600, Loss: 511072.468750
Iteration: 610, Loss: 511018.000000
Iteration: 620, Loss: 510975.875000
Iteration: 630, Loss: 510945.343750
Iteration: 640, Loss: 510926.625000
Iteration: 650, Loss: 510917.843750
Iteration: 660, Loss: 510910.312500
Iteration: 670, Loss: 510906.625000
Iteration: 680, Loss: 510904.937500
Iteration: 690, Loss: 510907.812500
Iteration: 700, Loss: 510913.875000
Iteration: 710, Loss: 510919.875000
Iteration: 720, Loss: 510926.843750
Iteration: 730, Loss: 510936.937500
Iteration: 740, Loss: 510950.812500
Iteration: 750, Loss: 510963.812500
Iteration: 760, Loss: 510978.500000
Iteration: 770, Loss: 510989.312500
Iteration: 780, Loss: 510990.625000
Iteration: 790, Loss: 510987.687500
Iteration: 800, Loss: 510981.656250
Iteration: 810, Loss: 510977.156250
Iteration: 820, Loss: 510972.562500
Iteration: 830, Loss: 510967.500000
Iteration: 840, Loss: 510964.500000
Iteration: 850, Loss: 510962.062500
Iteration: 860, Loss: 510956.187500
Iteration: 870, Loss: 510948.156250
Iteration: 880, Loss: 510938.562500
Iteration: 890, Loss: 510926.468750
Iteration: 900, Loss: 510910.968750
Iteration: 910, Loss: 510892.093750
Iteration: 920, Loss: 510870.937500
Iteration: 930, Loss: 510847.031250
Iteration: 940, Loss: 510819.062500
Iteration: 950, Loss: 510789.687500
Iteration: 960, Loss: 510759.843750
Iteration: 970, Loss: 510728.937500
Iteration: 980, Loss: 510697.593750
Iteration: 990, Loss: 510666.156250
Iteration: 1000, Loss: 510635.125000
Iteration: 1010, Loss: 510604.687500
Iteration: 1020, Loss: 510574.312500
Iteration: 1030, Loss: 510545.000000
Iteration: 1040, Loss: 510516.906250
Iteration: 1050, Loss: 510488.781250
Iteration: 1060, Loss: 510463.187500
Iteration: 1070, Loss: 510439.406250
Iteration: 1080, Loss: 510419.031250
Iteration: 1090, Loss: 510398.093750

```
Iteration: 1100, Loss: 510388.250000
Iteration: 1110, Loss: 510388.250000
Iteration: 1120, Loss: 510426.312500
Iteration: 1130, Loss: 510596.375000
Iteration: 1140, Loss: 510795.718750
Iteration: 1150, Loss: 510931.937500
Iteration: 1160, Loss: 510906.343750
Iteration: 1170, Loss: 510847.500000
Iteration: 1180, Loss: 510855.718750
Iteration: 1190, Loss: 510960.156250
Iteration: 1200, Loss: 511060.343750
Elapsed time: 292.04s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1407020.0
Iteration: 10, Loss: 691880.812500
Iteration: 20, Loss: 403866.437500
Iteration: 30, Loss: 261382.125000
Iteration: 40, Loss: 227020.250000
Iteration: 50, Loss: 211478.390625
Iteration: 60, Loss: 206755.875000
Iteration: 70, Loss: 205364.843750
Iteration: 80, Loss: 204686.859375
Iteration: 90, Loss: 204508.484375
Iteration: 100, Loss: 204433.328125
Iteration: 110, Loss: 307820.312500
Iteration: 120, Loss: 307133.906250
Iteration: 130, Loss: 306951.156250
Iteration: 140, Loss: 306920.125000
Iteration: 150, Loss: 306907.718750
Iteration: 160, Loss: 306899.468750
Iteration: 170, Loss: 306901.000000
Iteration: 180, Loss: 306894.312500
Iteration: 190, Loss: 306896.093750
Iteration: 200, Loss: 306927.562500
Iteration: 210, Loss: 102278.046875
Iteration: 220, Loss: 102214.562500
Iteration: 230, Loss: 102200.406250
Iteration: 240, Loss: 102195.289062
Iteration: 250, Loss: 102193.960938
Iteration: 260, Loss: 102193.328125
Iteration: 270, Loss: 102193.304688
Iteration: 280, Loss: 102193.109375
Iteration: 290, Loss: 102193.078125
Iteration: 300, Loss: 102192.976562
Iteration: 310, Loss: 102193.148438
Iteration: 320, Loss: 102193.132812
Iteration: 330, Loss: 102193.101562
Iteration: 340, Loss: 102193.015625
Iteration: 350, Loss: 102193.101562
Iteration: 360, Loss: 102193.093750
Iteration: 370, Loss: 102193.046875
Iteration: 380, Loss: 102193.054688
Iteration: 390, Loss: 102193.070312
Iteration: 400, Loss: 102193.054688
Iteration: 410, Loss: 102193.070312
Iteration: 420, Loss: 102193.062500
Iteration: 430, Loss: 102193.046875
```

Iteration: 440, Loss: 102193.054688
Iteration: 450, Loss: 102193.062500
Iteration: 460, Loss: 102193.062500
Iteration: 470, Loss: 102193.070312
Iteration: 480, Loss: 102193.054688
Iteration: 490, Loss: 102193.054688
Iteration: 500, Loss: 102193.070312
Iteration: 510, Loss: 102193.062500
Iteration: 520, Loss: 102193.054688
Iteration: 530, Loss: 102193.062500
Iteration: 540, Loss: 102193.062500
Iteration: 550, Loss: 102193.062500
Iteration: 560, Loss: 102193.054688
Iteration: 570, Loss: 102193.062500
Iteration: 580, Loss: 102193.054688
Iteration: 590, Loss: 102193.054688
Iteration: 600, Loss: 102193.078125
Iteration: 610, Loss: 102193.070312
Iteration: 620, Loss: 102193.062500
Iteration: 630, Loss: 102193.054688
Iteration: 640, Loss: 102193.054688
Iteration: 650, Loss: 102193.062500
Iteration: 660, Loss: 102193.046875
Iteration: 670, Loss: 102193.046875
Iteration: 680, Loss: 102193.054688
Iteration: 690, Loss: 102193.054688
Iteration: 700, Loss: 102193.070312
Iteration: 710, Loss: 102193.062500
Iteration: 720, Loss: 102193.062500
Iteration: 730, Loss: 102193.070312
Iteration: 740, Loss: 102193.070312
Iteration: 750, Loss: 102193.046875
Iteration: 760, Loss: 102193.054688
Iteration: 770, Loss: 102193.054688
Iteration: 780, Loss: 102193.054688
Iteration: 790, Loss: 102193.062500
Iteration: 800, Loss: 102193.054688
Iteration: 810, Loss: 102193.046875
Iteration: 820, Loss: 102193.062500
Iteration: 830, Loss: 102193.062500
Iteration: 840, Loss: 102193.062500
Iteration: 850, Loss: 102193.062500
Iteration: 860, Loss: 102193.062500
Iteration: 870, Loss: 102193.062500
Iteration: 880, Loss: 102193.078125
Iteration: 890, Loss: 102193.054688
Iteration: 900, Loss: 102193.062500
Iteration: 910, Loss: 102193.062500
Iteration: 920, Loss: 102193.062500
Iteration: 930, Loss: 102193.046875
Iteration: 940, Loss: 102193.054688
Iteration: 950, Loss: 102193.054688
Iteration: 960, Loss: 102193.054688
Iteration: 970, Loss: 102193.062500
Iteration: 980, Loss: 102193.054688
Iteration: 990, Loss: 102193.054688
Iteration: 1000, Loss: 102193.062500
Iteration: 1010, Loss: 102193.054688
Iteration: 1020, Loss: 102193.062500
Iteration: 1030, Loss: 102193.085938

```
Iteration: 1040, Loss: 102193.093750
Iteration: 1050, Loss: 102193.195312
Iteration: 1060, Loss: 102193.773438
Iteration: 1070, Loss: 102195.476562
Iteration: 1080, Loss: 102200.750000
Iteration: 1090, Loss: 102211.093750
Iteration: 1100, Loss: 102222.085938
Iteration: 1110, Loss: 102238.125000
Iteration: 1120, Loss: 102260.750000
Iteration: 1130, Loss: 102289.656250
Iteration: 1140, Loss: 102316.859375
Iteration: 1150, Loss: 102327.953125
Iteration: 1160, Loss: 102330.804688
Iteration: 1170, Loss: 102324.531250
Iteration: 1180, Loss: 102317.359375
Iteration: 1190, Loss: 102305.664062
Iteration: 1200, Loss: 102297.500000
Elapsed time: 0:00:18.491483
X is normalized.
X is normalized.
Found nearest neighbor
(3705078, 2)
Initial Loss: 7153320.0
Iteration: 10, Loss: 3468874.750000
Iteration: 20, Loss: 2021453.125000
Iteration: 30, Loss: 1312356.875000
Iteration: 40, Loss: 1137209.000000
Iteration: 50, Loss: 1055057.625000
Iteration: 60, Loss: 1029155.500000
Iteration: 70, Loss: 1023210.875000
Iteration: 80, Loss: 1020900.375000
Iteration: 90, Loss: 1020305.625000
Iteration: 100, Loss: 1020071.125000
Iteration: 110, Loss: 1522454.500000
Iteration: 120, Loss: 1518068.250000
Iteration: 130, Loss: 1516994.875000
Iteration: 140, Loss: 1516902.125000
Iteration: 150, Loss: 1516799.625000
Iteration: 160, Loss: 1516749.000000
Iteration: 170, Loss: 1516735.625000
Iteration: 180, Loss: 1516639.750000
Iteration: 190, Loss: 1516726.625000
Iteration: 200, Loss: 1516860.500000
Iteration: 210, Loss: 510263.187500
Iteration: 220, Loss: 510031.531250
Iteration: 230, Loss: 509976.781250
Iteration: 240, Loss: 509955.093750
Iteration: 250, Loss: 509949.843750
Iteration: 260, Loss: 509946.375000
Iteration: 270, Loss: 509945.687500
Iteration: 280, Loss: 509945.812500
Iteration: 290, Loss: 509945.906250
Iteration: 300, Loss: 509946.125000
Iteration: 310, Loss: 509945.437500
Iteration: 320, Loss: 509945.468750
Iteration: 330, Loss: 509945.843750
Iteration: 340, Loss: 509945.937500
Iteration: 350, Loss: 509945.937500
Iteration: 360, Loss: 509946.156250
Iteration: 370, Loss: 509946.187500
```

Iteration: 380, Loss: 509946.218750
Iteration: 390, Loss: 509946.093750
Iteration: 400, Loss: 509946.187500
Iteration: 410, Loss: 509946.218750
Iteration: 420, Loss: 509946.218750
Iteration: 430, Loss: 509946.218750
Iteration: 440, Loss: 509946.218750
Iteration: 450, Loss: 509946.250000
Iteration: 460, Loss: 509946.218750
Iteration: 470, Loss: 509946.218750
Iteration: 480, Loss: 509946.281250
Iteration: 490, Loss: 509946.187500
Iteration: 500, Loss: 509946.218750
Iteration: 510, Loss: 509946.250000
Iteration: 520, Loss: 509946.187500
Iteration: 530, Loss: 509946.250000
Iteration: 540, Loss: 509946.250000
Iteration: 550, Loss: 509946.218750
Iteration: 560, Loss: 509946.218750
Iteration: 570, Loss: 509946.250000
Iteration: 580, Loss: 509946.250000
Iteration: 590, Loss: 509946.218750
Iteration: 600, Loss: 509946.250000
Iteration: 610, Loss: 509946.218750
Iteration: 620, Loss: 509946.281250
Iteration: 630, Loss: 509946.250000
Iteration: 640, Loss: 509946.250000
Iteration: 650, Loss: 509946.281250
Iteration: 660, Loss: 509946.281250
Iteration: 670, Loss: 509946.312500
Iteration: 680, Loss: 509946.187500
Iteration: 690, Loss: 509946.218750
Iteration: 700, Loss: 509946.281250
Iteration: 710, Loss: 509946.250000
Iteration: 720, Loss: 509946.218750
Iteration: 730, Loss: 509946.218750
Iteration: 740, Loss: 509946.250000
Iteration: 750, Loss: 509946.281250
Iteration: 760, Loss: 509946.187500
Iteration: 770, Loss: 509946.281250
Iteration: 780, Loss: 509946.187500
Iteration: 790, Loss: 509946.250000
Iteration: 800, Loss: 509946.250000
Iteration: 810, Loss: 509946.312500
Iteration: 820, Loss: 509946.250000
Iteration: 830, Loss: 509946.281250
Iteration: 840, Loss: 509946.312500
Iteration: 850, Loss: 509946.312500
Iteration: 860, Loss: 509946.250000
Iteration: 870, Loss: 509946.281250
Iteration: 880, Loss: 509946.218750
Iteration: 890, Loss: 509946.187500
Iteration: 900, Loss: 509946.281250
Iteration: 910, Loss: 509946.312500
Iteration: 920, Loss: 509946.250000
Iteration: 930, Loss: 509946.250000
Iteration: 940, Loss: 509946.250000
Iteration: 950, Loss: 509946.281250
Iteration: 960, Loss: 509946.281250
Iteration: 970, Loss: 509946.187500

```

Iteration: 980, Loss: 509946.218750
Iteration: 990, Loss: 509946.250000
Iteration: 1000, Loss: 509946.250000
Iteration: 1010, Loss: 509946.250000
Iteration: 1020, Loss: 509946.312500
Iteration: 1030, Loss: 509946.281250
Iteration: 1040, Loss: 509946.312500
Iteration: 1050, Loss: 509947.500000
Iteration: 1060, Loss: 509949.750000
Iteration: 1070, Loss: 509958.218750
Iteration: 1080, Loss: 509982.406250
Iteration: 1090, Loss: 510014.156250
Iteration: 1100, Loss: 510050.187500
Iteration: 1110, Loss: 510110.625000
Iteration: 1120, Loss: 510196.281250
Iteration: 1130, Loss: 510308.843750
Iteration: 1140, Loss: 510388.437500
Iteration: 1150, Loss: 510454.375000
Iteration: 1160, Loss: 510481.937500
Iteration: 1170, Loss: 510451.500000
Iteration: 1180, Loss: 510415.937500
Iteration: 1190, Loss: 510380.156250
Iteration: 1200, Loss: 510356.281250
Elapsed time: 0:01:21.977704

```

```
In [12]: ## pacmap과 isolation forest 1차 이용(2)
val_score_1 = f1_score(ori_val_df['Class'], np.round(val_pred_set_1), average='macro')
print(f'Validation F1 Score : [{val_score_1}]')
print(classification_report(ori_val_df['Class'], np.round(val_pred_set_1)))
print(confusion_matrix(ori_val_df['Class'], np.round(val_pred_set_1)))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	28432
1	0.89	0.80	0.84	30
accuracy			1.00	28462
macro avg	0.94	0.90	0.92	28462
weighted avg	1.00	1.00	1.00	28462

```
[[28429      3]
 [    6     24]]
```

```
In [13]: ## pacmap과 isolation forest 1차 이용(3)
# 1차 저장
chujung_train_1 = pd.DataFrame({'Class':np.round(train_pred_set_1)})
chujung_val_1 = pd.DataFrame({'Class':np.round(val_pred_set_1)})
chujung_test_1 = pd.DataFrame({'Class':np.round(test_pred_set_1)})

result_train_1 = pd.concat([train_df,chujung_train_1], axis=1)
result_val_1 = pd.concat([val_df,chujung_val_1], axis=1)
result_test_1 = pd.concat([test_df,chujung_test_1], axis=1)

result_train_1.to_csv('result_train_1.csv', index=False)
result_val_1.to_csv('result_val_1.csv', index=False)
result_test_1.to_csv('result_test_1.csv', index=False)
```

```
In [14]: ## pacmap과 isolation forest 1차 이용(4)
# 1차 불러오기
train_pred_set_1 = np.array(pd.read_csv('result_train_1.csv')['Class'])
val_pred_set_1 = np.array(pd.read_csv('result_val_1.csv')['Class'])
test_pred_set_1 = np.array(pd.read_csv('result_test_1.csv')['Class'])
```

저는 이에 더해 변수를 좀 더 추가하면서 val score를 높일 수 있을까 생각하였습니다. 그래서 저는 설령 outlier들이 inlier안에 숨어있을지라도 inlier들이 모여있으면 isolation forest가 판단하는데 있어서 도움이 된다고 들었습니다. 이에 따라 저는 outlier들이 inlier안에 들어있지만 최대한 outlier들의 분산이 적으면서 최대한 inlier들의 분산을 넓은 변수를 찾고자 하였습니다. (i.e. (inlier들의 분산/outlier들의 분산)이 높은 것들). 역시 validation set의 통계정보를 이용하였습니다. 그 다음, outlier의 중앙값을 기준으로 inlier들을 나누어서, 각 inlier들의 중앙값이 outlier의 중앙값과 차이가 큰지(wilcoxon rank sum test), 각 inlier들의 분산이 작은지(해당 분산이 작으면 inlier와 구별할 수 있다고 생각하였습니다.)를 보았습니다. 변수를 최대한 적게 선택하고자 이번엔 해당 방법들에서 나온 변수들의 교집합을 선택하였습니다.

```
In [15]: ## pacmap과 isolation forest 2차 이용
inside_in_inlier= []

for what_val in range(30):
    if ori_val_df.iloc[np.where(ori_val_df['Class'] == 1)].max()[what_val] < or
        if ori_val_df.iloc[np.where(ori_val_df['Class'] == 1)].min()[what_val] >
            inside_in_inlier.append(what_val)

print(inside_in_inlier)

var_chai = []
for what_val in inside_in_inlier:
    print(what_val , ori_val_df.iloc[np.where(ori_val_df['Class'] == 0)[0],what_
    var_chai.append(ori_val_df.iloc[np.where(ori_val_df['Class'] == 0)[0],what_v

old_born_idx = np.argsort((-1)*np.array(var_chai))[:5]
old_born_idx = np.array(inside_in_inlier)[list(old_born_idx)]
old_born_idx = list(old_born_idx)
print(old_born_idx)

left_side_l = []
right_side_l = []
for jjkk in inside_in_inlier:
    the_one_the_one = ori_val_df.iloc[np.where(ori_val_df['Class'] == 1)[0],jjkk
    median_the_one = the_one_the_one.median()
    the_zero_the_zero = ori_val_df.iloc[np.where(ori_val_df['Class'] == 0)[0],jj
    left_zero_val_df = the_zero_the_zero.iloc[np.where(the_zero_the_zero < media
    right_zero_val_df = the_zero_the_zero.iloc[np.where(the_zero_the_zero > medi
    left_ppp = (ranksums(left_zero_val_df, the_one_the_one).pvalue)
    right_ppp = (ranksums(right_zero_val_df, the_one_the_one).pvalue)

    left_side_l.append((left_ppp*1000)*((left_zero_val_df.var())))
    right_side_l.append((right_ppp*1000)*((right_zero_val_df.var())))

left_born_idx = np.argsort(np.array(left_side_l))[:5]
left_born_idx = np.array(inside_in_inlier)[list(left_born_idx)]
left_born_idx = list(left_born_idx)

right_born_idx = np.argsort(np.array(right_side_l))[:5]
```

```

right_born_idx = np.array(inside_in_inlier)[list(right_born_idx)]
right_born_idx = list(right_born_idx)

print(left_born_idx)
print(right_born_idx)

dhk_add_list = list(set(old_born_idx) & set(left_born_idx) & set(right_born_idx))
print(dhk_add_list)

```

[0, 4, 5, 12, 14, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 29]
0 0.056611459159604575
4 0.04223885928524464
5 0.7704461525574464
12 0.7649192536979115
14 0.5924508408804137
18 0.20398714517248465
19 0.37151998169343714
20 0.06681438471306114
21 0.47076015284507583
22 0.5165263768776028
23 1.3711184736256365
24 0.46796859661491885
25 0.7132699705783758
26 0.03718584298750135
27 0.2518467335386625
29 0.7823436145492879
[np.int64(23), np.int64(29), np.int64(5), np.int64(12), np.int64(25)]
[np.int64(25), np.int64(21), np.int64(23), np.int64(24), np.int64(20)]
[np.int64(5), np.int64(0), np.int64(24), np.int64(23), np.int64(29)]
[np.int64(23)]

지금까지 선택한 변수들을 바탕으로 pacmap과 isolation forest를 돌려 예측합니다.

```

In [16]: ## pacmap과 isolation forest 3차 이용(1)
# pacmap과 isolation forest를 이용한 2차 예측
hhmm = 7
vlskffo_var = superior_var2 + dhk_add_list
what_val = vlskffo_var

for num in range(hhmm):
    embedding_3 = pacmap.PaCMAP(n_components=len(what_val), n_neighbors=None, MN
pacmac_train_3 = embedding_3.fit_transform(np.array(train_df.iloc[:,what_val
pacmac_val_3 = embedding_3.transform(np.array(val_df.iloc[:,what_val]), basi
pacmac_test_3 = embedding_3.transform(np.array(test_df.iloc[:,what_val]), ba

pac_model_3 = IsolationForest(n_estimators=1000, contamination=0.00121, verb
pac_model_3.fit(pacmac_train_3[:,[1,2,3]])

if num == 0:
    train_pred_set_3 = pac_model_3.predict(pacmac_train_3[:,[1,2,3]]) # model
    train_pred_set_3 = get_pred_label(train_pred_set_3)

    val_pred_set_3 = pac_model_3.predict(pacmac_val_3[:,[1,2,3]]) # model pr
    val_pred_set_3 = get_pred_label(val_pred_set_3)

    test_pred_set_3 = pac_model_3.predict(pacmac_test_3[:,[1,2,3]]) # model
    test_pred_set_3 = get_pred_label(test_pred_set_3)
else:
    train_pred_3 = pac_model_3.predict(pacmac_train_3[:,[1,2,3]]) # model pr
    train_pred_3 = get_pred_label(train_pred_3)

```

```
train_pred_set_3 = train_pred_set_3 + train_pred_3

val_pred_3 = pac_model_3.predict(pacmac_val_3[:,[1,2,3]]) # model predict
val_pred_3 = get_pred_label(val_pred_3)
val_pred_set_3 = val_pred_set_3 + val_pred_3

test_pred_3 = pac_model_3.predict(pacmac_test_3[:,[1,2,3]]) # model predict
test_pred_3 = get_pred_label(test_pred_3)
test_pred_set_3 = test_pred_set_3 + test_pred_3

train_pred_set_3 = train_pred_set_3/hhmm
val_pred_set_3 = val_pred_set_3/hhmm
test_pred_set_3 = test_pred_set_3/hhmm
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 600), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2463065.500000
Iteration: 20, Loss: 2168820.000000
Iteration: 30, Loss: 2027100.250000
Iteration: 40, Loss: 1920327.500000
Iteration: 50, Loss: 1820272.125000
Iteration: 60, Loss: 1714928.000000
Iteration: 70, Loss: 1597360.875000
Iteration: 80, Loss: 1458421.125000
Iteration: 90, Loss: 1283208.625000
Iteration: 100, Loss: 1019862.625000
Iteration: 110, Loss: 1339004.500000
Iteration: 120, Loss: 1314413.000000
Iteration: 130, Loss: 1302774.875000
Iteration: 140, Loss: 1298921.250000
Iteration: 150, Loss: 1298035.375000
Iteration: 160, Loss: 1297977.375000
Iteration: 170, Loss: 1297912.750000
Iteration: 180, Loss: 1298625.875000
Iteration: 190, Loss: 1299667.000000
Iteration: 200, Loss: 1300169.375000
Iteration: 210, Loss: 549576.062500
Iteration: 220, Loss: 543433.375000
Iteration: 230, Loss: 538795.687500
Iteration: 240, Loss: 536168.187500
Iteration: 250, Loss: 533976.437500
Iteration: 260, Loss: 532234.375000
Iteration: 270, Loss: 530947.500000
Iteration: 280, Loss: 529970.812500
Iteration: 290, Loss: 529124.250000
Iteration: 300, Loss: 528219.000000
Iteration: 310, Loss: 527234.187500
Iteration: 320, Loss: 526213.312500
Iteration: 330, Loss: 525292.625000
Iteration: 340, Loss: 524469.437500
Iteration: 350, Loss: 523795.250000
Iteration: 360, Loss: 523276.187500
Iteration: 370, Loss: 522889.031250
Iteration: 380, Loss: 522625.625000
Iteration: 390, Loss: 522412.312500
Iteration: 400, Loss: 522192.500000
Iteration: 410, Loss: 521944.812500
Iteration: 420, Loss: 521661.000000
Iteration: 430, Loss: 521398.375000
Iteration: 440, Loss: 521143.000000
Iteration: 450, Loss: 520898.687500
Iteration: 460, Loss: 520672.968750
Iteration: 470, Loss: 520472.937500
Iteration: 480, Loss: 520287.468750
Iteration: 490, Loss: 520116.093750
```

```
Iteration: 500, Loss: 519939.625000
Iteration: 510, Loss: 519763.843750
Iteration: 520, Loss: 519593.375000
Iteration: 530, Loss: 519424.406250
Iteration: 540, Loss: 519243.437500
Iteration: 550, Loss: 519066.437500
Iteration: 560, Loss: 518903.250000
Iteration: 570, Loss: 518760.687500
Iteration: 580, Loss: 518637.281250
Iteration: 590, Loss: 518518.437500
Iteration: 600, Loss: 518426.406250
Iteration: 610, Loss: 518354.218750
Iteration: 620, Loss: 518310.718750
Iteration: 630, Loss: 518272.062500
Iteration: 640, Loss: 518239.375000
Iteration: 650, Loss: 518221.093750
Iteration: 660, Loss: 518203.562500
Iteration: 670, Loss: 518191.312500
Iteration: 680, Loss: 518188.093750
Iteration: 690, Loss: 518189.343750
Iteration: 700, Loss: 518197.906250
Iteration: 710, Loss: 518219.937500
Iteration: 720, Loss: 518248.781250
Iteration: 730, Loss: 518274.687500
Iteration: 740, Loss: 518285.062500
Iteration: 750, Loss: 518296.843750
Iteration: 760, Loss: 518299.468750
Iteration: 770, Loss: 518295.687500
Iteration: 780, Loss: 518290.875000
Iteration: 790, Loss: 518284.062500
Iteration: 800, Loss: 518276.406250
Elapsed time: 207.61s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1388097.25
Iteration: 10, Loss: 719907.437500
Iteration: 20, Loss: 356721.718750
Iteration: 30, Loss: 257458.953125
Iteration: 40, Loss: 222019.218750
Iteration: 50, Loss: 212443.078125
Iteration: 60, Loss: 209281.281250
Iteration: 70, Loss: 207830.437500
Iteration: 80, Loss: 207462.687500
Iteration: 90, Loss: 207255.203125
Iteration: 100, Loss: 207201.906250
Iteration: 110, Loss: 312576.875000
Iteration: 120, Loss: 311598.750000
Iteration: 130, Loss: 311351.656250
Iteration: 140, Loss: 311280.375000
Iteration: 150, Loss: 311219.500000
Iteration: 160, Loss: 311193.093750
Iteration: 170, Loss: 311179.375000
Iteration: 180, Loss: 311175.156250
Iteration: 190, Loss: 311198.187500
Iteration: 200, Loss: 311224.000000
Iteration: 210, Loss: 103693.531250
Iteration: 220, Loss: 103611.695312
Iteration: 230, Loss: 103594.398438
```

```
Iteration: 240, Loss: 103588.304688
Iteration: 250, Loss: 103586.265625
Iteration: 260, Loss: 103585.445312
Iteration: 270, Loss: 103585.132812
Iteration: 280, Loss: 103585.210938
Iteration: 290, Loss: 103585.054688
Iteration: 300, Loss: 103584.875000
Iteration: 310, Loss: 103584.789062
Iteration: 320, Loss: 103584.804688
Iteration: 330, Loss: 103584.773438
Iteration: 340, Loss: 103584.734375
Iteration: 350, Loss: 103584.781250
Iteration: 360, Loss: 103584.789062
Iteration: 370, Loss: 103584.781250
Iteration: 380, Loss: 103584.765625
Iteration: 390, Loss: 103584.773438
Iteration: 400, Loss: 103584.789062
Iteration: 410, Loss: 103584.789062
Iteration: 420, Loss: 103584.789062
Iteration: 430, Loss: 103584.789062
Iteration: 440, Loss: 103584.781250
Iteration: 450, Loss: 103584.789062
Iteration: 460, Loss: 103584.781250
Iteration: 470, Loss: 103584.773438
Iteration: 480, Loss: 103584.781250
Iteration: 490, Loss: 103584.781250
Iteration: 500, Loss: 103584.781250
Iteration: 510, Loss: 103584.781250
Iteration: 520, Loss: 103584.789062
Iteration: 530, Loss: 103584.781250
Iteration: 540, Loss: 103584.789062
Iteration: 550, Loss: 103584.789062
Iteration: 560, Loss: 103584.773438
Iteration: 570, Loss: 103584.781250
Iteration: 580, Loss: 103584.789062
Iteration: 590, Loss: 103584.789062
Iteration: 600, Loss: 103584.773438
Iteration: 610, Loss: 103584.781250
Iteration: 620, Loss: 103584.789062
Iteration: 630, Loss: 103584.789062
Iteration: 640, Loss: 103584.789062
Iteration: 650, Loss: 103584.773438
Iteration: 660, Loss: 103584.773438
Iteration: 670, Loss: 103584.773438
Iteration: 680, Loss: 103584.789062
Iteration: 690, Loss: 103584.773438
Iteration: 700, Loss: 103584.789062
Iteration: 710, Loss: 103584.789062
Iteration: 720, Loss: 103584.781250
Iteration: 730, Loss: 103584.781250
Iteration: 740, Loss: 103584.781250
Iteration: 750, Loss: 103584.781250
Iteration: 760, Loss: 103584.781250
Iteration: 770, Loss: 103584.773438
Iteration: 780, Loss: 103584.789062
Iteration: 790, Loss: 103584.781250
Iteration: 800, Loss: 103584.789062
Elapsed time: 0:00:12.924615
X is normalized.
X is normalized.
```

```
Found nearest neighbor
(3705078, 2)
Initial Loss: 7105749.5
Iteration: 10, Loss: 3591179.250000
Iteration: 20, Loss: 1788278.625000
Iteration: 30, Loss: 1298152.750000
Iteration: 40, Loss: 1109341.375000
Iteration: 50, Loss: 1059805.250000
Iteration: 60, Loss: 1043670.625000
Iteration: 70, Loss: 1038663.375000
Iteration: 80, Loss: 1037385.062500
Iteration: 90, Loss: 1036777.312500
Iteration: 100, Loss: 1036596.250000
Iteration: 110, Loss: 1544772.500000
Iteration: 120, Loss: 1537469.375000
Iteration: 130, Loss: 1535710.125000
Iteration: 140, Loss: 1535380.750000
Iteration: 150, Loss: 1535093.000000
Iteration: 160, Loss: 1534933.750000
Iteration: 170, Loss: 1534926.375000
Iteration: 180, Loss: 1534952.125000
Iteration: 190, Loss: 1535031.750000
Iteration: 200, Loss: 1535232.375000
Iteration: 210, Loss: 518589.218750
Iteration: 220, Loss: 518331.468750
Iteration: 230, Loss: 518273.718750
Iteration: 240, Loss: 518253.812500
Iteration: 250, Loss: 518247.000000
Iteration: 260, Loss: 518245.781250
Iteration: 270, Loss: 518245.187500
Iteration: 280, Loss: 518245.062500
Iteration: 290, Loss: 518244.937500
Iteration: 300, Loss: 518245.968750
Iteration: 310, Loss: 518246.062500
Iteration: 320, Loss: 518245.812500
Iteration: 330, Loss: 518245.343750
Iteration: 340, Loss: 518245.562500
Iteration: 350, Loss: 518245.531250
Iteration: 360, Loss: 518245.468750
Iteration: 370, Loss: 518245.437500
Iteration: 380, Loss: 518245.375000
Iteration: 390, Loss: 518245.437500
Iteration: 400, Loss: 518245.406250
Iteration: 410, Loss: 518245.375000
Iteration: 420, Loss: 518245.406250
Iteration: 430, Loss: 518245.406250
Iteration: 440, Loss: 518245.406250
Iteration: 450, Loss: 518245.406250
Iteration: 460, Loss: 518245.437500
Iteration: 470, Loss: 518245.437500
Iteration: 480, Loss: 518245.406250
Iteration: 490, Loss: 518245.437500
Iteration: 500, Loss: 518245.437500
Iteration: 510, Loss: 518245.406250
Iteration: 520, Loss: 518245.406250
Iteration: 530, Loss: 518245.437500
Iteration: 540, Loss: 518245.437500
Iteration: 550, Loss: 518245.437500
Iteration: 560, Loss: 518245.406250
Iteration: 570, Loss: 518245.437500
```

```
Iteration: 580, Loss: 518245.437500
Iteration: 590, Loss: 518245.375000
Iteration: 600, Loss: 518245.406250
Iteration: 610, Loss: 518245.437500
Iteration: 620, Loss: 518245.437500
Iteration: 630, Loss: 518245.406250
Iteration: 640, Loss: 518245.406250
Iteration: 650, Loss: 518245.437500
Iteration: 660, Loss: 518245.343750
Iteration: 670, Loss: 518245.375000
Iteration: 680, Loss: 518245.437500
Iteration: 690, Loss: 518245.406250
Iteration: 700, Loss: 518245.406250
Iteration: 710, Loss: 518245.406250
Iteration: 720, Loss: 518245.406250
Iteration: 730, Loss: 518245.406250
Iteration: 740, Loss: 518245.437500
Iteration: 750, Loss: 518245.406250
Iteration: 760, Loss: 518245.406250
Iteration: 770, Loss: 518245.375000
Iteration: 780, Loss: 518245.375000
Iteration: 790, Loss: 518245.406250
Iteration: 800, Loss: 518245.343750
Elapsed time: 0:00:53.203068
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 600), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2462551.250000
Iteration: 20, Loss: 2168994.250000
Iteration: 30, Loss: 2027328.750000
Iteration: 40, Loss: 1920638.250000
Iteration: 50, Loss: 1820541.750000
Iteration: 60, Loss: 1715210.750000
Iteration: 70, Loss: 1597552.625000
Iteration: 80, Loss: 1458523.625000
Iteration: 90, Loss: 1283240.500000
Iteration: 100, Loss: 1019776.000000
Iteration: 110, Loss: 1338759.500000
Iteration: 120, Loss: 1314167.500000
Iteration: 130, Loss: 1302822.000000
Iteration: 140, Loss: 1299116.000000
Iteration: 150, Loss: 1298421.625000
Iteration: 160, Loss: 1298676.000000
Iteration: 170, Loss: 1298793.750000
Iteration: 180, Loss: 1299684.000000
Iteration: 190, Loss: 1300816.875000
Iteration: 200, Loss: 1301264.500000
Iteration: 210, Loss: 549762.625000
Iteration: 220, Loss: 543726.125000
Iteration: 230, Loss: 538973.875000
Iteration: 240, Loss: 536383.125000
Iteration: 250, Loss: 534376.187500
Iteration: 260, Loss: 532749.375000
Iteration: 270, Loss: 531471.875000
Iteration: 280, Loss: 530408.750000
Iteration: 290, Loss: 529474.000000
Iteration: 300, Loss: 528463.750000
Iteration: 310, Loss: 527427.625000
Iteration: 320, Loss: 526399.125000
Iteration: 330, Loss: 525460.750000
Iteration: 340, Loss: 524626.125000
Iteration: 350, Loss: 523912.437500
Iteration: 360, Loss: 523380.125000
Iteration: 370, Loss: 522980.406250
Iteration: 380, Loss: 522708.031250
Iteration: 390, Loss: 522500.906250
Iteration: 400, Loss: 522303.125000
Iteration: 410, Loss: 522085.562500
Iteration: 420, Loss: 521847.718750
Iteration: 430, Loss: 521603.937500
Iteration: 440, Loss: 521358.562500
Iteration: 450, Loss: 521117.000000
Iteration: 460, Loss: 520874.593750
Iteration: 470, Loss: 520647.468750
Iteration: 480, Loss: 520432.781250
Iteration: 490, Loss: 520234.000000
```

```
Iteration: 500, Loss: 520035.187500
Iteration: 510, Loss: 519842.531250
Iteration: 520, Loss: 519655.000000
Iteration: 530, Loss: 519473.312500
Iteration: 540, Loss: 519289.781250
Iteration: 550, Loss: 519117.562500
Iteration: 560, Loss: 518961.000000
Iteration: 570, Loss: 518818.187500
Iteration: 580, Loss: 518696.750000
Iteration: 590, Loss: 518594.875000
Iteration: 600, Loss: 518511.250000
Iteration: 610, Loss: 518447.375000
Iteration: 620, Loss: 518394.812500
Iteration: 630, Loss: 518360.187500
Iteration: 640, Loss: 518329.125000
Iteration: 650, Loss: 518303.531250
Iteration: 660, Loss: 518285.750000
Iteration: 670, Loss: 518283.500000
Iteration: 680, Loss: 518282.031250
Iteration: 690, Loss: 518285.062500
Iteration: 700, Loss: 518300.375000
Iteration: 710, Loss: 518323.156250
Iteration: 720, Loss: 518351.562500
Iteration: 730, Loss: 518375.437500
Iteration: 740, Loss: 518387.812500
Iteration: 750, Loss: 518394.968750
Iteration: 760, Loss: 518398.812500
Iteration: 770, Loss: 518396.062500
Iteration: 780, Loss: 518392.312500
Iteration: 790, Loss: 518385.718750
Iteration: 800, Loss: 518379.312500
Elapsed time: 182.30s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1388176.0
Iteration: 10, Loss: 720202.437500
Iteration: 20, Loss: 349703.125000
Iteration: 30, Loss: 255004.250000
Iteration: 40, Loss: 222447.109375
Iteration: 50, Loss: 212605.156250
Iteration: 60, Loss: 209194.296875
Iteration: 70, Loss: 207887.453125
Iteration: 80, Loss: 207472.453125
Iteration: 90, Loss: 207287.875000
Iteration: 100, Loss: 207230.093750
Iteration: 110, Loss: 312561.281250
Iteration: 120, Loss: 311638.687500
Iteration: 130, Loss: 311396.843750
Iteration: 140, Loss: 311310.125000
Iteration: 150, Loss: 311261.406250
Iteration: 160, Loss: 311226.562500
Iteration: 170, Loss: 311226.593750
Iteration: 180, Loss: 311233.281250
Iteration: 190, Loss: 311240.343750
Iteration: 200, Loss: 311264.593750
Iteration: 210, Loss: 103707.726562
Iteration: 220, Loss: 103626.515625
Iteration: 230, Loss: 103610.398438
```

```
Iteration: 240, Loss: 103603.593750
Iteration: 250, Loss: 103602.453125
Iteration: 260, Loss: 103601.500000
Iteration: 270, Loss: 103601.195312
Iteration: 280, Loss: 103601.296875
Iteration: 290, Loss: 103601.187500
Iteration: 300, Loss: 103601.164062
Iteration: 310, Loss: 103601.062500
Iteration: 320, Loss: 103601.132812
Iteration: 330, Loss: 103601.093750
Iteration: 340, Loss: 103601.078125
Iteration: 350, Loss: 103601.101562
Iteration: 360, Loss: 103601.125000
Iteration: 370, Loss: 103601.140625
Iteration: 380, Loss: 103601.117188
Iteration: 390, Loss: 103601.117188
Iteration: 400, Loss: 103601.117188
Iteration: 410, Loss: 103601.125000
Iteration: 420, Loss: 103601.125000
Iteration: 430, Loss: 103601.109375
Iteration: 440, Loss: 103601.125000
Iteration: 450, Loss: 103601.109375
Iteration: 460, Loss: 103601.125000
Iteration: 470, Loss: 103601.125000
Iteration: 480, Loss: 103601.125000
Iteration: 490, Loss: 103601.125000
Iteration: 500, Loss: 103601.125000
Iteration: 510, Loss: 103601.125000
Iteration: 520, Loss: 103601.117188
Iteration: 530, Loss: 103601.109375
Iteration: 540, Loss: 103601.125000
Iteration: 550, Loss: 103601.125000
Iteration: 560, Loss: 103601.117188
Iteration: 570, Loss: 103601.117188
Iteration: 580, Loss: 103601.125000
Iteration: 590, Loss: 103601.109375
Iteration: 600, Loss: 103601.125000
Iteration: 610, Loss: 103601.117188
Iteration: 620, Loss: 103601.125000
Iteration: 630, Loss: 103601.117188
Iteration: 640, Loss: 103601.117188
Iteration: 650, Loss: 103601.117188
Iteration: 660, Loss: 103601.109375
Iteration: 670, Loss: 103601.117188
Iteration: 680, Loss: 103601.125000
Iteration: 690, Loss: 103601.125000
Iteration: 700, Loss: 103601.125000
Iteration: 710, Loss: 103601.125000
Iteration: 720, Loss: 103601.117188
Iteration: 730, Loss: 103601.125000
Iteration: 740, Loss: 103601.125000
Iteration: 750, Loss: 103601.117188
Iteration: 760, Loss: 103601.117188
Iteration: 770, Loss: 103601.125000
Iteration: 780, Loss: 103601.132812
Iteration: 790, Loss: 103601.117188
Iteration: 800, Loss: 103601.117188
Elapsed time: 0:00:11.734374
X is normalized.
X is normalized.
```

```
Found nearest neighbor  
(3705078, 2)  
Initial Loss: 7105150.5  
Iteration: 10, Loss: 3592503.750000  
Iteration: 20, Loss: 1754146.500000  
Iteration: 30, Loss: 1286833.250000  
Iteration: 40, Loss: 1112357.750000  
Iteration: 50, Loss: 1060660.000000  
Iteration: 60, Loss: 1043848.187500  
Iteration: 70, Loss: 1039404.625000  
Iteration: 80, Loss: 1037948.750000  
Iteration: 90, Loss: 1037351.062500  
Iteration: 100, Loss: 1037161.687500  
Iteration: 110, Loss: 1545399.000000  
Iteration: 120, Loss: 1538386.000000  
Iteration: 130, Loss: 1536718.125000  
Iteration: 140, Loss: 1536347.500000  
Iteration: 150, Loss: 1536054.000000  
Iteration: 160, Loss: 1535896.000000  
Iteration: 170, Loss: 1535855.875000  
Iteration: 180, Loss: 1535989.250000  
Iteration: 190, Loss: 1535968.625000  
Iteration: 200, Loss: 1536174.500000  
Iteration: 210, Loss: 518861.781250  
Iteration: 220, Loss: 518617.031250  
Iteration: 230, Loss: 518555.750000  
Iteration: 240, Loss: 518532.437500  
Iteration: 250, Loss: 518528.031250  
Iteration: 260, Loss: 518525.187500  
Iteration: 270, Loss: 518524.218750  
Iteration: 280, Loss: 518523.312500  
Iteration: 290, Loss: 518522.843750  
Iteration: 300, Loss: 518522.562500  
Iteration: 310, Loss: 518522.750000  
Iteration: 320, Loss: 518522.375000  
Iteration: 330, Loss: 518522.375000  
Iteration: 340, Loss: 518522.250000  
Iteration: 350, Loss: 518522.250000  
Iteration: 360, Loss: 518521.968750  
Iteration: 370, Loss: 518522.156250  
Iteration: 380, Loss: 518522.156250  
Iteration: 390, Loss: 518522.156250  
Iteration: 400, Loss: 518522.125000  
Iteration: 410, Loss: 518522.156250  
Iteration: 420, Loss: 518522.156250  
Iteration: 430, Loss: 518522.125000  
Iteration: 440, Loss: 518522.125000  
Iteration: 450, Loss: 518522.156250  
Iteration: 460, Loss: 518522.156250  
Iteration: 470, Loss: 518522.125000  
Iteration: 480, Loss: 518522.156250  
Iteration: 490, Loss: 518522.125000  
Iteration: 500, Loss: 518522.156250  
Iteration: 510, Loss: 518522.187500  
Iteration: 520, Loss: 518522.125000  
Iteration: 530, Loss: 518522.125000  
Iteration: 540, Loss: 518522.187500  
Iteration: 550, Loss: 518522.156250  
Iteration: 560, Loss: 518522.125000  
Iteration: 570, Loss: 518522.156250
```

```
Iteration: 580, Loss: 518522.125000
Iteration: 590, Loss: 518522.156250
Iteration: 600, Loss: 518522.156250
Iteration: 610, Loss: 518522.156250
Iteration: 620, Loss: 518522.156250
Iteration: 630, Loss: 518522.187500
Iteration: 640, Loss: 518522.156250
Iteration: 650, Loss: 518522.156250
Iteration: 660, Loss: 518522.156250
Iteration: 670, Loss: 518522.156250
Iteration: 680, Loss: 518522.156250
Iteration: 690, Loss: 518522.156250
Iteration: 700, Loss: 518522.156250
Iteration: 710, Loss: 518522.156250
Iteration: 720, Loss: 518522.156250
Iteration: 730, Loss: 518522.156250
Iteration: 740, Loss: 518522.156250
Iteration: 750, Loss: 518522.156250
Iteration: 760, Loss: 518522.156250
Iteration: 770, Loss: 518522.156250
Iteration: 780, Loss: 518522.156250
Iteration: 790, Loss: 518522.125000
Iteration: 800, Loss: 518522.156250
Elapsed time: 0:00:50.957568
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 600), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2460713.250000
Iteration: 20, Loss: 2167951.500000
Iteration: 30, Loss: 2026775.625000
Iteration: 40, Loss: 1920198.000000
Iteration: 50, Loss: 1820220.750000
Iteration: 60, Loss: 1714923.000000
Iteration: 70, Loss: 1597271.750000
Iteration: 80, Loss: 1458360.125000
Iteration: 90, Loss: 1283153.000000
Iteration: 100, Loss: 1019723.375000
Iteration: 110, Loss: 1338871.875000
Iteration: 120, Loss: 1314116.000000
Iteration: 130, Loss: 1302497.000000
Iteration: 140, Loss: 1298393.125000
Iteration: 150, Loss: 1297529.375000
Iteration: 160, Loss: 1297421.000000
Iteration: 170, Loss: 1297492.375000
Iteration: 180, Loss: 1298341.250000
Iteration: 190, Loss: 1299238.875000
Iteration: 200, Loss: 1299744.750000
Iteration: 210, Loss: 549376.375000
Iteration: 220, Loss: 543244.687500
Iteration: 230, Loss: 538617.437500
Iteration: 240, Loss: 535975.312500
Iteration: 250, Loss: 533769.312500
Iteration: 260, Loss: 532075.875000
Iteration: 270, Loss: 530786.062500
Iteration: 280, Loss: 529825.625000
Iteration: 290, Loss: 529010.750000
Iteration: 300, Loss: 528137.625000
Iteration: 310, Loss: 527136.000000
Iteration: 320, Loss: 526125.437500
Iteration: 330, Loss: 525204.500000
Iteration: 340, Loss: 524376.187500
Iteration: 350, Loss: 523693.812500
Iteration: 360, Loss: 523177.062500
Iteration: 370, Loss: 522800.750000
Iteration: 380, Loss: 522534.500000
Iteration: 390, Loss: 522315.062500
Iteration: 400, Loss: 522095.187500
Iteration: 410, Loss: 521839.343750
Iteration: 420, Loss: 521560.812500
Iteration: 430, Loss: 521290.750000
Iteration: 440, Loss: 521025.562500
Iteration: 450, Loss: 520780.187500
Iteration: 460, Loss: 520557.750000
Iteration: 470, Loss: 520356.250000
Iteration: 480, Loss: 520175.125000
Iteration: 490, Loss: 519996.187500
```

```
Iteration: 500, Loss: 519816.843750
Iteration: 510, Loss: 519639.093750
Iteration: 520, Loss: 519457.906250
Iteration: 530, Loss: 519281.125000
Iteration: 540, Loss: 519101.156250
Iteration: 550, Loss: 518924.718750
Iteration: 560, Loss: 518768.531250
Iteration: 570, Loss: 518630.937500
Iteration: 580, Loss: 518508.500000
Iteration: 590, Loss: 518401.875000
Iteration: 600, Loss: 518315.812500
Iteration: 610, Loss: 518255.062500
Iteration: 620, Loss: 518212.125000
Iteration: 630, Loss: 518179.218750
Iteration: 640, Loss: 518153.843750
Iteration: 650, Loss: 518125.281250
Iteration: 660, Loss: 518096.000000
Iteration: 670, Loss: 518075.875000
Iteration: 680, Loss: 518066.937500
Iteration: 690, Loss: 518069.468750
Iteration: 700, Loss: 518082.000000
Iteration: 710, Loss: 518109.312500
Iteration: 720, Loss: 518139.093750
Iteration: 730, Loss: 518159.437500
Iteration: 740, Loss: 518166.718750
Iteration: 750, Loss: 518173.562500
Iteration: 760, Loss: 518173.875000
Iteration: 770, Loss: 518169.250000
Iteration: 780, Loss: 518162.156250
Iteration: 790, Loss: 518154.531250
Iteration: 800, Loss: 518147.875000
Elapsed time: 176.97s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1388114.625
Iteration: 10, Loss: 720638.937500
Iteration: 20, Loss: 357080.312500
Iteration: 30, Loss: 257000.406250
Iteration: 40, Loss: 221701.109375
Iteration: 50, Loss: 212476.890625
Iteration: 60, Loss: 209267.515625
Iteration: 70, Loss: 207825.921875
Iteration: 80, Loss: 207463.875000
Iteration: 90, Loss: 207264.312500
Iteration: 100, Loss: 207209.984375
Iteration: 110, Loss: 312608.343750
Iteration: 120, Loss: 311611.593750
Iteration: 130, Loss: 311359.312500
Iteration: 140, Loss: 311277.093750
Iteration: 150, Loss: 311219.156250
Iteration: 160, Loss: 311194.312500
Iteration: 170, Loss: 311192.656250
Iteration: 180, Loss: 311191.000000
Iteration: 190, Loss: 311198.500000
Iteration: 200, Loss: 311215.750000
Iteration: 210, Loss: 103687.812500
Iteration: 220, Loss: 103613.523438
Iteration: 230, Loss: 103595.851562
```

```
Iteration: 240, Loss: 103591.312500
Iteration: 250, Loss: 103589.250000
Iteration: 260, Loss: 103589.101562
Iteration: 270, Loss: 103588.960938
Iteration: 280, Loss: 103588.765625
Iteration: 290, Loss: 103588.726562
Iteration: 300, Loss: 103588.687500
Iteration: 310, Loss: 103588.632812
Iteration: 320, Loss: 103588.632812
Iteration: 330, Loss: 103588.679688
Iteration: 340, Loss: 103588.617188
Iteration: 350, Loss: 103588.625000
Iteration: 360, Loss: 103588.625000
Iteration: 370, Loss: 103588.640625
Iteration: 380, Loss: 103588.648438
Iteration: 390, Loss: 103588.609375
Iteration: 400, Loss: 103588.632812
Iteration: 410, Loss: 103588.632812
Iteration: 420, Loss: 103588.625000
Iteration: 430, Loss: 103588.625000
Iteration: 440, Loss: 103588.632812
Iteration: 450, Loss: 103588.625000
Iteration: 460, Loss: 103588.632812
Iteration: 470, Loss: 103588.632812
Iteration: 480, Loss: 103588.632812
Iteration: 490, Loss: 103588.625000
Iteration: 500, Loss: 103588.625000
Iteration: 510, Loss: 103588.632812
Iteration: 520, Loss: 103588.625000
Iteration: 530, Loss: 103588.640625
Iteration: 540, Loss: 103588.625000
Iteration: 550, Loss: 103588.625000
Iteration: 560, Loss: 103588.617188
Iteration: 570, Loss: 103588.632812
Iteration: 580, Loss: 103588.632812
Iteration: 590, Loss: 103588.632812
Iteration: 600, Loss: 103588.632812
Iteration: 610, Loss: 103588.640625
Iteration: 620, Loss: 103588.632812
Iteration: 630, Loss: 103588.625000
Iteration: 640, Loss: 103588.632812
Iteration: 650, Loss: 103588.632812
Iteration: 660, Loss: 103588.632812
Iteration: 670, Loss: 103588.632812
Iteration: 680, Loss: 103588.640625
Iteration: 690, Loss: 103588.625000
Iteration: 700, Loss: 103588.632812
Iteration: 710, Loss: 103588.632812
Iteration: 720, Loss: 103588.632812
Iteration: 730, Loss: 103588.632812
Iteration: 740, Loss: 103588.625000
Iteration: 750, Loss: 103588.625000
Iteration: 760, Loss: 103588.632812
Iteration: 770, Loss: 103588.632812
Iteration: 780, Loss: 103588.632812
Iteration: 790, Loss: 103588.632812
Iteration: 800, Loss: 103588.632812
Elapsed time: 0:00:12.254405
X is normalized.
X is normalized.
```

```
Found nearest neighbor
(3705078, 2)
Initial Loss: 7106178.0
Iteration: 10, Loss: 3594661.250000
Iteration: 20, Loss: 1791943.875000
Iteration: 30, Loss: 1296395.875000
Iteration: 40, Loss: 1108557.125000
Iteration: 50, Loss: 1059979.125000
Iteration: 60, Loss: 1043708.500000
Iteration: 70, Loss: 1038843.750000
Iteration: 80, Loss: 1037547.937500
Iteration: 90, Loss: 1036940.375000
Iteration: 100, Loss: 1036755.937500
Iteration: 110, Loss: 1545060.500000
Iteration: 120, Loss: 1537808.125000
Iteration: 130, Loss: 1535932.625000
Iteration: 140, Loss: 1535535.125000
Iteration: 150, Loss: 1535276.250000
Iteration: 160, Loss: 1535167.750000
Iteration: 170, Loss: 1535212.250000
Iteration: 180, Loss: 1535188.500000
Iteration: 190, Loss: 1535166.875000
Iteration: 200, Loss: 1535321.875000
Iteration: 210, Loss: 518644.687500
Iteration: 220, Loss: 518405.343750
Iteration: 230, Loss: 518350.656250
Iteration: 240, Loss: 518330.593750
Iteration: 250, Loss: 518325.937500
Iteration: 260, Loss: 518325.781250
Iteration: 270, Loss: 518325.031250
Iteration: 280, Loss: 518324.531250
Iteration: 290, Loss: 518324.062500
Iteration: 300, Loss: 518324.843750
Iteration: 310, Loss: 518324.375000
Iteration: 320, Loss: 518324.218750
Iteration: 330, Loss: 518324.125000
Iteration: 340, Loss: 518323.937500
Iteration: 350, Loss: 518323.906250
Iteration: 360, Loss: 518323.875000
Iteration: 370, Loss: 518323.906250
Iteration: 380, Loss: 518323.843750
Iteration: 390, Loss: 518323.812500
Iteration: 400, Loss: 518323.843750
Iteration: 410, Loss: 518323.812500
Iteration: 420, Loss: 518323.781250
Iteration: 430, Loss: 518323.812500
Iteration: 440, Loss: 518323.812500
Iteration: 450, Loss: 518323.812500
Iteration: 460, Loss: 518323.875000
Iteration: 470, Loss: 518323.812500
Iteration: 480, Loss: 518323.812500
Iteration: 490, Loss: 518323.843750
Iteration: 500, Loss: 518323.812500
Iteration: 510, Loss: 518323.812500
Iteration: 520, Loss: 518323.843750
Iteration: 530, Loss: 518323.812500
Iteration: 540, Loss: 518323.781250
Iteration: 550, Loss: 518323.843750
Iteration: 560, Loss: 518323.843750
Iteration: 570, Loss: 518323.812500
```

```
Iteration: 580, Loss: 518323.781250
Iteration: 590, Loss: 518323.843750
Iteration: 600, Loss: 518323.812500
Iteration: 610, Loss: 518323.781250
Iteration: 620, Loss: 518323.812500
Iteration: 630, Loss: 518323.781250
Iteration: 640, Loss: 518323.812500
Iteration: 650, Loss: 518323.812500
Iteration: 660, Loss: 518323.812500
Iteration: 670, Loss: 518323.812500
Iteration: 680, Loss: 518323.812500
Iteration: 690, Loss: 518323.812500
Iteration: 700, Loss: 518323.812500
Iteration: 710, Loss: 518323.875000
Iteration: 720, Loss: 518323.812500
Iteration: 730, Loss: 518323.812500
Iteration: 740, Loss: 518323.812500
Iteration: 750, Loss: 518323.812500
Iteration: 760, Loss: 518323.812500
Iteration: 770, Loss: 518323.781250
Iteration: 780, Loss: 518323.781250
Iteration: 790, Loss: 518323.875000
Iteration: 800, Loss: 518323.875000
Elapsed time: 0:00:50.826368
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 600), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2460667.750000
Iteration: 20, Loss: 2168033.250000
Iteration: 30, Loss: 2026873.625000
Iteration: 40, Loss: 1920231.000000
Iteration: 50, Loss: 1820205.750000
Iteration: 60, Loss: 1714953.500000
Iteration: 70, Loss: 1597304.000000
Iteration: 80, Loss: 1458470.000000
Iteration: 90, Loss: 1283265.250000
Iteration: 100, Loss: 1019958.687500
Iteration: 110, Loss: 1338912.750000
Iteration: 120, Loss: 1314096.000000
Iteration: 130, Loss: 1302732.125000
Iteration: 140, Loss: 1298848.125000
Iteration: 150, Loss: 1298312.500000
Iteration: 160, Loss: 1298533.500000
Iteration: 170, Loss: 1298659.500000
Iteration: 180, Loss: 1299681.250000
Iteration: 190, Loss: 1300634.250000
Iteration: 200, Loss: 1300945.875000
Iteration: 210, Loss: 549645.250000
Iteration: 220, Loss: 543619.437500
Iteration: 230, Loss: 538872.750000
Iteration: 240, Loss: 536274.000000
Iteration: 250, Loss: 534252.937500
Iteration: 260, Loss: 532679.250000
Iteration: 270, Loss: 531445.312500
Iteration: 280, Loss: 530390.062500
Iteration: 290, Loss: 529439.000000
Iteration: 300, Loss: 528483.312500
Iteration: 310, Loss: 527491.812500
Iteration: 320, Loss: 526453.875000
Iteration: 330, Loss: 525514.250000
Iteration: 340, Loss: 524663.500000
Iteration: 350, Loss: 523945.562500
Iteration: 360, Loss: 523380.562500
Iteration: 370, Loss: 522959.437500
Iteration: 380, Loss: 522667.656250
Iteration: 390, Loss: 522449.312500
Iteration: 400, Loss: 522234.781250
Iteration: 410, Loss: 522016.562500
Iteration: 420, Loss: 521785.812500
Iteration: 430, Loss: 521542.062500
Iteration: 440, Loss: 521307.593750
Iteration: 450, Loss: 521068.937500
Iteration: 460, Loss: 520836.500000
Iteration: 470, Loss: 520613.781250
Iteration: 480, Loss: 520393.125000
Iteration: 490, Loss: 520187.656250
```

```
Iteration: 500, Loss: 519990.937500
Iteration: 510, Loss: 519793.500000
Iteration: 520, Loss: 519603.000000
Iteration: 530, Loss: 519422.937500
Iteration: 540, Loss: 519245.500000
Iteration: 550, Loss: 519073.218750
Iteration: 560, Loss: 518912.062500
Iteration: 570, Loss: 518772.468750
Iteration: 580, Loss: 518651.125000
Iteration: 590, Loss: 518546.406250
Iteration: 600, Loss: 518455.937500
Iteration: 610, Loss: 518387.812500
Iteration: 620, Loss: 518337.687500
Iteration: 630, Loss: 518293.500000
Iteration: 640, Loss: 518263.343750
Iteration: 650, Loss: 518242.750000
Iteration: 660, Loss: 518223.125000
Iteration: 670, Loss: 518216.500000
Iteration: 680, Loss: 518215.250000
Iteration: 690, Loss: 518219.031250
Iteration: 700, Loss: 518236.062500
Iteration: 710, Loss: 518258.437500
Iteration: 720, Loss: 518287.281250
Iteration: 730, Loss: 518311.875000
Iteration: 740, Loss: 518324.406250
Iteration: 750, Loss: 518335.531250
Iteration: 760, Loss: 518340.062500
Iteration: 770, Loss: 518337.125000
Iteration: 780, Loss: 518332.250000
Iteration: 790, Loss: 518325.218750
Iteration: 800, Loss: 518318.687500
Elapsed time: 177.78s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1388271.25
Iteration: 10, Loss: 722087.375000
Iteration: 20, Loss: 350073.187500
Iteration: 30, Loss: 255204.343750
Iteration: 40, Loss: 222647.609375
Iteration: 50, Loss: 212633.281250
Iteration: 60, Loss: 209292.390625
Iteration: 70, Loss: 207950.281250
Iteration: 80, Loss: 207542.843750
Iteration: 90, Loss: 207357.015625
Iteration: 100, Loss: 207302.171875
Iteration: 110, Loss: 312652.312500
Iteration: 120, Loss: 311712.406250
Iteration: 130, Loss: 311486.218750
Iteration: 140, Loss: 311416.531250
Iteration: 150, Loss: 311350.937500
Iteration: 160, Loss: 311334.562500
Iteration: 170, Loss: 311328.875000
Iteration: 180, Loss: 311338.375000
Iteration: 190, Loss: 311349.593750
Iteration: 200, Loss: 311364.125000
Iteration: 210, Loss: 103737.296875
Iteration: 220, Loss: 103660.242188
Iteration: 230, Loss: 103644.531250
```

```
Iteration: 240, Loss: 103637.929688
Iteration: 250, Loss: 103635.656250
Iteration: 260, Loss: 103635.578125
Iteration: 270, Loss: 103634.992188
Iteration: 280, Loss: 103634.921875
Iteration: 290, Loss: 103634.867188
Iteration: 300, Loss: 103635.000000
Iteration: 310, Loss: 103635.015625
Iteration: 320, Loss: 103634.960938
Iteration: 330, Loss: 103634.890625
Iteration: 340, Loss: 103634.859375
Iteration: 350, Loss: 103634.890625
Iteration: 360, Loss: 103634.835938
Iteration: 370, Loss: 103634.843750
Iteration: 380, Loss: 103634.835938
Iteration: 390, Loss: 103634.843750
Iteration: 400, Loss: 103634.843750
Iteration: 410, Loss: 103634.835938
Iteration: 420, Loss: 103634.828125
Iteration: 430, Loss: 103634.843750
Iteration: 440, Loss: 103634.828125
Iteration: 450, Loss: 103634.835938
Iteration: 460, Loss: 103634.828125
Iteration: 470, Loss: 103634.843750
Iteration: 480, Loss: 103634.835938
Iteration: 490, Loss: 103634.835938
Iteration: 500, Loss: 103634.835938
Iteration: 510, Loss: 103634.828125
Iteration: 520, Loss: 103634.835938
Iteration: 530, Loss: 103634.828125
Iteration: 540, Loss: 103634.843750
Iteration: 550, Loss: 103634.835938
Iteration: 560, Loss: 103634.828125
Iteration: 570, Loss: 103634.835938
Iteration: 580, Loss: 103634.835938
Iteration: 590, Loss: 103634.835938
Iteration: 600, Loss: 103634.828125
Iteration: 610, Loss: 103634.843750
Iteration: 620, Loss: 103634.843750
Iteration: 630, Loss: 103634.828125
Iteration: 640, Loss: 103634.835938
Iteration: 650, Loss: 103634.835938
Iteration: 660, Loss: 103634.835938
Iteration: 670, Loss: 103634.835938
Iteration: 680, Loss: 103634.835938
Iteration: 690, Loss: 103634.843750
Iteration: 700, Loss: 103634.828125
Iteration: 710, Loss: 103634.835938
Iteration: 720, Loss: 103634.835938
Iteration: 730, Loss: 103634.843750
Iteration: 740, Loss: 103634.828125
Iteration: 750, Loss: 103634.835938
Iteration: 760, Loss: 103634.835938
Iteration: 770, Loss: 103634.843750
Iteration: 780, Loss: 103634.843750
Iteration: 790, Loss: 103634.828125
Iteration: 800, Loss: 103634.835938
Elapsed time: 0:00:11.641820
X is normalized.
X is normalized.
```

```
Found nearest neighbor
(3705078, 2)
Initial Loss: 7105690.5
Iteration: 10, Loss: 3598223.500000
Iteration: 20, Loss: 1755814.625000
Iteration: 30, Loss: 1287649.750000
Iteration: 40, Loss: 1113395.625000
Iteration: 50, Loss: 1060659.625000
Iteration: 60, Loss: 1044361.187500
Iteration: 70, Loss: 1039749.250000
Iteration: 80, Loss: 1038300.187500
Iteration: 90, Loss: 1037752.125000
Iteration: 100, Loss: 1037539.687500
Iteration: 110, Loss: 1545650.750000
Iteration: 120, Loss: 1538872.500000
Iteration: 130, Loss: 1537278.000000
Iteration: 140, Loss: 1536919.250000
Iteration: 150, Loss: 1536615.125000
Iteration: 160, Loss: 1536446.625000
Iteration: 170, Loss: 1536529.250000
Iteration: 180, Loss: 1536598.625000
Iteration: 190, Loss: 1536578.750000
Iteration: 200, Loss: 1536739.375000
Iteration: 210, Loss: 519061.937500
Iteration: 220, Loss: 518804.937500
Iteration: 230, Loss: 518744.281250
Iteration: 240, Loss: 518726.937500
Iteration: 250, Loss: 518723.062500
Iteration: 260, Loss: 518720.781250
Iteration: 270, Loss: 518721.000000
Iteration: 280, Loss: 518722.437500
Iteration: 290, Loss: 518722.687500
Iteration: 300, Loss: 518722.312500
Iteration: 310, Loss: 518722.562500
Iteration: 320, Loss: 518722.093750
Iteration: 330, Loss: 518721.875000
Iteration: 340, Loss: 518721.968750
Iteration: 350, Loss: 518722.000000
Iteration: 360, Loss: 518722.062500
Iteration: 370, Loss: 518722.031250
Iteration: 380, Loss: 518721.968750
Iteration: 390, Loss: 518722.000000
Iteration: 400, Loss: 518721.968750
Iteration: 410, Loss: 518722.000000
Iteration: 420, Loss: 518721.968750
Iteration: 430, Loss: 518722.000000
Iteration: 440, Loss: 518722.000000
Iteration: 450, Loss: 518722.000000
Iteration: 460, Loss: 518722.000000
Iteration: 470, Loss: 518722.000000
Iteration: 480, Loss: 518722.031250
Iteration: 490, Loss: 518721.968750
Iteration: 500, Loss: 518722.000000
Iteration: 510, Loss: 518721.968750
Iteration: 520, Loss: 518722.000000
Iteration: 530, Loss: 518722.031250
Iteration: 540, Loss: 518722.000000
Iteration: 550, Loss: 518722.000000
Iteration: 560, Loss: 518722.000000
Iteration: 570, Loss: 518721.968750
```

```
Iteration: 580, Loss: 518722.000000
Iteration: 590, Loss: 518721.968750
Iteration: 600, Loss: 518722.000000
Iteration: 610, Loss: 518722.000000
Iteration: 620, Loss: 518722.000000
Iteration: 630, Loss: 518721.968750
Iteration: 640, Loss: 518722.000000
Iteration: 650, Loss: 518722.000000
Iteration: 660, Loss: 518722.000000
Iteration: 670, Loss: 518722.031250
Iteration: 680, Loss: 518721.968750
Iteration: 690, Loss: 518722.000000
Iteration: 700, Loss: 518722.031250
Iteration: 710, Loss: 518722.031250
Iteration: 720, Loss: 518722.031250
Iteration: 730, Loss: 518722.000000
Iteration: 740, Loss: 518722.093750
Iteration: 750, Loss: 518722.000000
Iteration: 760, Loss: 518722.031250
Iteration: 770, Loss: 518722.031250
Iteration: 780, Loss: 518722.000000
Iteration: 790, Loss: 518722.000000
Iteration: 800, Loss: 518721.968750
Elapsed time: 0:00:52.412726
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 600), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2460899.750000
Iteration: 20, Loss: 2168486.000000
Iteration: 30, Loss: 2027366.250000
Iteration: 40, Loss: 1920592.875000
Iteration: 50, Loss: 1820417.500000
Iteration: 60, Loss: 1715151.750000
Iteration: 70, Loss: 1597455.750000
Iteration: 80, Loss: 1458494.750000
Iteration: 90, Loss: 1283199.750000
Iteration: 100, Loss: 1019557.625000
Iteration: 110, Loss: 1338808.625000
Iteration: 120, Loss: 1314110.375000
Iteration: 130, Loss: 1302629.750000
Iteration: 140, Loss: 1298776.125000
Iteration: 150, Loss: 1298169.500000
Iteration: 160, Loss: 1298379.375000
Iteration: 170, Loss: 1298619.375000
Iteration: 180, Loss: 1299452.250000
Iteration: 190, Loss: 1300569.250000
Iteration: 200, Loss: 1300948.125000
Iteration: 210, Loss: 549645.812500
Iteration: 220, Loss: 543623.312500
Iteration: 230, Loss: 538870.125000
Iteration: 240, Loss: 536254.312500
Iteration: 250, Loss: 534231.750000
Iteration: 260, Loss: 532616.687500
Iteration: 270, Loss: 531370.125000
Iteration: 280, Loss: 530295.250000
Iteration: 290, Loss: 529340.000000
Iteration: 300, Loss: 528345.937500
Iteration: 310, Loss: 527349.875000
Iteration: 320, Loss: 526322.437500
Iteration: 330, Loss: 525382.312500
Iteration: 340, Loss: 524558.812500
Iteration: 350, Loss: 523867.375000
Iteration: 360, Loss: 523336.500000
Iteration: 370, Loss: 522935.406250
Iteration: 380, Loss: 522669.875000
Iteration: 390, Loss: 522460.718750
Iteration: 400, Loss: 522259.312500
Iteration: 410, Loss: 522038.500000
Iteration: 420, Loss: 521807.125000
Iteration: 430, Loss: 521566.562500
Iteration: 440, Loss: 521322.937500
Iteration: 450, Loss: 521083.937500
Iteration: 460, Loss: 520852.125000
Iteration: 470, Loss: 520628.281250
Iteration: 480, Loss: 520415.625000
Iteration: 490, Loss: 520210.406250
```

```
Iteration: 500, Loss: 520012.281250
Iteration: 510, Loss: 519823.156250
Iteration: 520, Loss: 519636.062500
Iteration: 530, Loss: 519444.906250
Iteration: 540, Loss: 519264.218750
Iteration: 550, Loss: 519092.656250
Iteration: 560, Loss: 518925.468750
Iteration: 570, Loss: 518784.906250
Iteration: 580, Loss: 518664.562500
Iteration: 590, Loss: 518548.593750
Iteration: 600, Loss: 518457.812500
Iteration: 610, Loss: 518390.062500
Iteration: 620, Loss: 518335.531250
Iteration: 630, Loss: 518290.750000
Iteration: 640, Loss: 518262.062500
Iteration: 650, Loss: 518240.750000
Iteration: 660, Loss: 518221.406250
Iteration: 670, Loss: 518214.937500
Iteration: 680, Loss: 518210.906250
Iteration: 690, Loss: 518217.718750
Iteration: 700, Loss: 518237.156250
Iteration: 710, Loss: 518257.875000
Iteration: 720, Loss: 518285.812500
Iteration: 730, Loss: 518309.781250
Iteration: 740, Loss: 518322.593750
Iteration: 750, Loss: 518331.125000
Iteration: 760, Loss: 518333.093750
Iteration: 770, Loss: 518328.437500
Iteration: 780, Loss: 518322.375000
Iteration: 790, Loss: 518314.312500
Iteration: 800, Loss: 518307.156250
Elapsed time: 179.03s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1388185.0
Iteration: 10, Loss: 720844.437500
Iteration: 20, Loss: 343992.281250
Iteration: 30, Loss: 253667.625000
Iteration: 40, Loss: 223575.484375
Iteration: 50, Loss: 212671.078125
Iteration: 60, Loss: 209185.546875
Iteration: 70, Loss: 207953.000000
Iteration: 80, Loss: 207497.828125
Iteration: 90, Loss: 207330.312500
Iteration: 100, Loss: 207269.906250
Iteration: 110, Loss: 312845.000000
Iteration: 120, Loss: 311775.218750
Iteration: 130, Loss: 311471.250000
Iteration: 140, Loss: 311381.437500
Iteration: 150, Loss: 311321.843750
Iteration: 160, Loss: 311303.500000
Iteration: 170, Loss: 311277.437500
Iteration: 180, Loss: 311296.187500
Iteration: 190, Loss: 311305.187500
Iteration: 200, Loss: 311334.437500
Iteration: 210, Loss: 103726.437500
Iteration: 220, Loss: 103645.867188
Iteration: 230, Loss: 103627.539062
```

```
Iteration: 240, Loss: 103620.992188
Iteration: 250, Loss: 103618.492188
Iteration: 260, Loss: 103617.882812
Iteration: 270, Loss: 103617.617188
Iteration: 280, Loss: 103617.257812
Iteration: 290, Loss: 103617.117188
Iteration: 300, Loss: 103617.062500
Iteration: 310, Loss: 103617.156250
Iteration: 320, Loss: 103617.210938
Iteration: 330, Loss: 103617.125000
Iteration: 340, Loss: 103617.101562
Iteration: 350, Loss: 103617.101562
Iteration: 360, Loss: 103617.093750
Iteration: 370, Loss: 103617.109375
Iteration: 380, Loss: 103617.085938
Iteration: 390, Loss: 103617.101562
Iteration: 400, Loss: 103617.109375
Iteration: 410, Loss: 103617.101562
Iteration: 420, Loss: 103617.101562
Iteration: 430, Loss: 103617.109375
Iteration: 440, Loss: 103617.101562
Iteration: 450, Loss: 103617.101562
Iteration: 460, Loss: 103617.101562
Iteration: 470, Loss: 103617.109375
Iteration: 480, Loss: 103617.117188
Iteration: 490, Loss: 103617.093750
Iteration: 500, Loss: 103617.101562
Iteration: 510, Loss: 103617.093750
Iteration: 520, Loss: 103617.101562
Iteration: 530, Loss: 103617.117188
Iteration: 540, Loss: 103617.093750
Iteration: 550, Loss: 103617.093750
Iteration: 560, Loss: 103617.109375
Iteration: 570, Loss: 103617.101562
Iteration: 580, Loss: 103617.093750
Iteration: 590, Loss: 103617.101562
Iteration: 600, Loss: 103617.101562
Iteration: 610, Loss: 103617.109375
Iteration: 620, Loss: 103617.093750
Iteration: 630, Loss: 103617.093750
Iteration: 640, Loss: 103617.101562
Iteration: 650, Loss: 103617.101562
Iteration: 660, Loss: 103617.101562
Iteration: 670, Loss: 103617.109375
Iteration: 680, Loss: 103617.093750
Iteration: 690, Loss: 103617.101562
Iteration: 700, Loss: 103617.109375
Iteration: 710, Loss: 103617.101562
Iteration: 720, Loss: 103617.093750
Iteration: 730, Loss: 103617.093750
Iteration: 740, Loss: 103617.101562
Iteration: 750, Loss: 103617.101562
Iteration: 760, Loss: 103617.093750
Iteration: 770, Loss: 103617.109375
Iteration: 780, Loss: 103617.101562
Iteration: 790, Loss: 103617.101562
Iteration: 800, Loss: 103617.101562
Elapsed time: 0:00:11.780440
X is normalized.
X is normalized.
```

```
Found nearest neighbor  
(3705078, 2)  
Initial Loss: 7105464.0  
Iteration: 10, Loss: 3594635.750000  
Iteration: 20, Loss: 1725196.000000  
Iteration: 30, Loss: 1279091.250000  
Iteration: 40, Loss: 1118370.000000  
Iteration: 50, Loss: 1061111.625000  
Iteration: 60, Loss: 1043937.062500  
Iteration: 70, Loss: 1039764.687500  
Iteration: 80, Loss: 1038179.062500  
Iteration: 90, Loss: 1037662.187500  
Iteration: 100, Loss: 1037445.125000  
Iteration: 110, Loss: 1547059.125000  
Iteration: 120, Loss: 1539242.000000  
Iteration: 130, Loss: 1537282.625000  
Iteration: 140, Loss: 1536791.375000  
Iteration: 150, Loss: 1536437.500000  
Iteration: 160, Loss: 1536295.875000  
Iteration: 170, Loss: 1536267.000000  
Iteration: 180, Loss: 1536338.375000  
Iteration: 190, Loss: 1536472.875000  
Iteration: 200, Loss: 1536664.250000  
Iteration: 210, Loss: 519025.343750  
Iteration: 220, Loss: 518760.218750  
Iteration: 230, Loss: 518699.093750  
Iteration: 240, Loss: 518675.218750  
Iteration: 250, Loss: 518668.718750  
Iteration: 260, Loss: 518667.656250  
Iteration: 270, Loss: 518666.437500  
Iteration: 280, Loss: 518666.843750  
Iteration: 290, Loss: 518667.156250  
Iteration: 300, Loss: 518666.312500  
Iteration: 310, Loss: 518665.937500  
Iteration: 320, Loss: 518666.062500  
Iteration: 330, Loss: 518666.062500  
Iteration: 340, Loss: 518666.000000  
Iteration: 350, Loss: 518666.156250  
Iteration: 360, Loss: 518666.125000  
Iteration: 370, Loss: 518666.093750  
Iteration: 380, Loss: 518666.250000  
Iteration: 390, Loss: 518666.250000  
Iteration: 400, Loss: 518666.250000  
Iteration: 410, Loss: 518666.218750  
Iteration: 420, Loss: 518666.218750  
Iteration: 430, Loss: 518666.218750  
Iteration: 440, Loss: 518666.218750  
Iteration: 450, Loss: 518666.187500  
Iteration: 460, Loss: 518666.218750  
Iteration: 470, Loss: 518666.218750  
Iteration: 480, Loss: 518666.218750  
Iteration: 490, Loss: 518666.187500  
Iteration: 500, Loss: 518666.218750  
Iteration: 510, Loss: 518666.250000  
Iteration: 520, Loss: 518666.218750  
Iteration: 530, Loss: 518666.218750  
Iteration: 540, Loss: 518666.218750  
Iteration: 550, Loss: 518666.218750  
Iteration: 560, Loss: 518666.218750  
Iteration: 570, Loss: 518666.218750
```

```
Iteration: 580, Loss: 518666.218750
Iteration: 590, Loss: 518666.250000
Iteration: 600, Loss: 518666.218750
Iteration: 610, Loss: 518666.218750
Iteration: 620, Loss: 518666.250000
Iteration: 630, Loss: 518666.187500
Iteration: 640, Loss: 518666.218750
Iteration: 650, Loss: 518666.250000
Iteration: 660, Loss: 518666.281250
Iteration: 670, Loss: 518666.250000
Iteration: 680, Loss: 518666.250000
Iteration: 690, Loss: 518666.312500
Iteration: 700, Loss: 518666.218750
Iteration: 710, Loss: 518666.250000
Iteration: 720, Loss: 518666.250000
Iteration: 730, Loss: 518666.218750
Iteration: 740, Loss: 518666.250000
Iteration: 750, Loss: 518666.218750
Iteration: 760, Loss: 518666.218750
Iteration: 770, Loss: 518666.187500
Iteration: 780, Loss: 518666.218750
Iteration: 790, Loss: 518666.250000
Iteration: 800, Loss: 518666.250000
Elapsed time: 0:00:52.882249
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 600), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2460967.250000
Iteration: 20, Loss: 2167751.750000
Iteration: 30, Loss: 2026763.250000
Iteration: 40, Loss: 1920084.125000
Iteration: 50, Loss: 1820063.750000
Iteration: 60, Loss: 1714706.000000
Iteration: 70, Loss: 1597132.875000
Iteration: 80, Loss: 1458246.875000
Iteration: 90, Loss: 1283080.375000
Iteration: 100, Loss: 1019854.000000
Iteration: 110, Loss: 1338688.125000
Iteration: 120, Loss: 1313925.125000
Iteration: 130, Loss: 1302681.250000
Iteration: 140, Loss: 1298635.000000
Iteration: 150, Loss: 1297794.250000
Iteration: 160, Loss: 1297687.250000
Iteration: 170, Loss: 1297727.000000
Iteration: 180, Loss: 1298518.125000
Iteration: 190, Loss: 1299400.250000
Iteration: 200, Loss: 1299832.125000
Iteration: 210, Loss: 549431.875000
Iteration: 220, Loss: 543335.500000
Iteration: 230, Loss: 538745.812500
Iteration: 240, Loss: 536089.812500
Iteration: 250, Loss: 533891.750000
Iteration: 260, Loss: 532160.625000
Iteration: 270, Loss: 530904.937500
Iteration: 280, Loss: 529940.875000
Iteration: 290, Loss: 529124.125000
Iteration: 300, Loss: 528244.812500
Iteration: 310, Loss: 527255.437500
Iteration: 320, Loss: 526243.750000
Iteration: 330, Loss: 525310.875000
Iteration: 340, Loss: 524503.625000
Iteration: 350, Loss: 523822.375000
Iteration: 360, Loss: 523293.250000
Iteration: 370, Loss: 522911.093750
Iteration: 380, Loss: 522632.000000
Iteration: 390, Loss: 522425.968750
Iteration: 400, Loss: 522199.968750
Iteration: 410, Loss: 521945.156250
Iteration: 420, Loss: 521666.687500
Iteration: 430, Loss: 521396.062500
Iteration: 440, Loss: 521136.468750
Iteration: 450, Loss: 520895.656250
Iteration: 460, Loss: 520674.093750
Iteration: 470, Loss: 520478.875000
Iteration: 480, Loss: 520295.187500
Iteration: 490, Loss: 520120.250000
```

```
Iteration: 500, Loss: 519933.531250
Iteration: 510, Loss: 519756.468750
Iteration: 520, Loss: 519577.312500
Iteration: 530, Loss: 519402.968750
Iteration: 540, Loss: 519224.000000
Iteration: 550, Loss: 519053.531250
Iteration: 560, Loss: 518891.437500
Iteration: 570, Loss: 518744.875000
Iteration: 580, Loss: 518619.812500
Iteration: 590, Loss: 518513.593750
Iteration: 600, Loss: 518422.968750
Iteration: 610, Loss: 518356.468750
Iteration: 620, Loss: 518301.250000
Iteration: 630, Loss: 518257.937500
Iteration: 640, Loss: 518221.968750
Iteration: 650, Loss: 518200.312500
Iteration: 660, Loss: 518178.312500
Iteration: 670, Loss: 518166.875000
Iteration: 680, Loss: 518162.000000
Iteration: 690, Loss: 518167.375000
Iteration: 700, Loss: 518181.750000
Iteration: 710, Loss: 518198.093750
Iteration: 720, Loss: 518228.281250
Iteration: 730, Loss: 518253.656250
Iteration: 740, Loss: 518266.625000
Iteration: 750, Loss: 518276.312500
Iteration: 760, Loss: 518274.343750
Iteration: 770, Loss: 518272.468750
Iteration: 780, Loss: 518268.093750
Iteration: 790, Loss: 518260.531250
Iteration: 800, Loss: 518252.000000
Elapsed time: 173.93s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1388212.875
Iteration: 10, Loss: 720627.750000
Iteration: 20, Loss: 358059.781250
Iteration: 30, Loss: 257735.250000
Iteration: 40, Loss: 221755.781250
Iteration: 50, Loss: 212465.859375
Iteration: 60, Loss: 209317.125000
Iteration: 70, Loss: 207833.609375
Iteration: 80, Loss: 207487.296875
Iteration: 90, Loss: 207280.500000
Iteration: 100, Loss: 207226.000000
Iteration: 110, Loss: 312597.218750
Iteration: 120, Loss: 311616.406250
Iteration: 130, Loss: 311389.718750
Iteration: 140, Loss: 311299.625000
Iteration: 150, Loss: 311250.343750
Iteration: 160, Loss: 311214.343750
Iteration: 170, Loss: 311229.562500
Iteration: 180, Loss: 311234.687500
Iteration: 190, Loss: 311236.250000
Iteration: 200, Loss: 311249.250000
Iteration: 210, Loss: 103700.914062
Iteration: 220, Loss: 103623.632812
Iteration: 230, Loss: 103606.773438
```

```
Iteration: 240, Loss: 103601.445312
Iteration: 250, Loss: 103598.726562
Iteration: 260, Loss: 103597.781250
Iteration: 270, Loss: 103597.406250
Iteration: 280, Loss: 103597.273438
Iteration: 290, Loss: 103597.539062
Iteration: 300, Loss: 103597.656250
Iteration: 310, Loss: 103597.609375
Iteration: 320, Loss: 103597.546875
Iteration: 330, Loss: 103597.429688
Iteration: 340, Loss: 103597.445312
Iteration: 350, Loss: 103597.437500
Iteration: 360, Loss: 103597.398438
Iteration: 370, Loss: 103597.414062
Iteration: 380, Loss: 103597.406250
Iteration: 390, Loss: 103597.429688
Iteration: 400, Loss: 103597.414062
Iteration: 410, Loss: 103597.414062
Iteration: 420, Loss: 103597.406250
Iteration: 430, Loss: 103597.406250
Iteration: 440, Loss: 103597.406250
Iteration: 450, Loss: 103597.406250
Iteration: 460, Loss: 103597.414062
Iteration: 470, Loss: 103597.414062
Iteration: 480, Loss: 103597.406250
Iteration: 490, Loss: 103597.406250
Iteration: 500, Loss: 103597.414062
Iteration: 510, Loss: 103597.406250
Iteration: 520, Loss: 103597.406250
Iteration: 530, Loss: 103597.414062
Iteration: 540, Loss: 103597.414062
Iteration: 550, Loss: 103597.406250
Iteration: 560, Loss: 103597.421875
Iteration: 570, Loss: 103597.421875
Iteration: 580, Loss: 103597.406250
Iteration: 590, Loss: 103597.414062
Iteration: 600, Loss: 103597.414062
Iteration: 610, Loss: 103597.414062
Iteration: 620, Loss: 103597.414062
Iteration: 630, Loss: 103597.421875
Iteration: 640, Loss: 103597.414062
Iteration: 650, Loss: 103597.406250
Iteration: 660, Loss: 103597.414062
Iteration: 670, Loss: 103597.398438
Iteration: 680, Loss: 103597.406250
Iteration: 690, Loss: 103597.414062
Iteration: 700, Loss: 103597.406250
Iteration: 710, Loss: 103597.406250
Iteration: 720, Loss: 103597.414062
Iteration: 730, Loss: 103597.414062
Iteration: 740, Loss: 103597.414062
Iteration: 750, Loss: 103597.421875
Iteration: 760, Loss: 103597.421875
Iteration: 770, Loss: 103597.414062
Iteration: 780, Loss: 103597.414062
Iteration: 790, Loss: 103597.414062
Iteration: 800, Loss: 103597.421875
Elapsed time: 0:00:11.373514
X is normalized.
X is normalized.
```

```
Found nearest neighbor
(3705078, 2)
Initial Loss: 7105483.0
Iteration: 10, Loss: 3595568.750000
Iteration: 20, Loss: 1795970.000000
Iteration: 30, Loss: 1300301.875000
Iteration: 40, Loss: 1108598.625000
Iteration: 50, Loss: 1059729.125000
Iteration: 60, Loss: 1043922.062500
Iteration: 70, Loss: 1038895.625000
Iteration: 80, Loss: 1037643.625000
Iteration: 90, Loss: 1037013.875000
Iteration: 100, Loss: 1036842.625000
Iteration: 110, Loss: 1544917.000000
Iteration: 120, Loss: 1537802.250000
Iteration: 130, Loss: 1536104.125000
Iteration: 140, Loss: 1535727.250000
Iteration: 150, Loss: 1535415.125000
Iteration: 160, Loss: 1535241.375000
Iteration: 170, Loss: 1535312.750000
Iteration: 180, Loss: 1535345.875000
Iteration: 190, Loss: 1535358.625000
Iteration: 200, Loss: 1535478.750000
Iteration: 210, Loss: 518697.625000
Iteration: 220, Loss: 518456.031250
Iteration: 230, Loss: 518396.468750
Iteration: 240, Loss: 518371.656250
Iteration: 250, Loss: 518368.218750
Iteration: 260, Loss: 518364.562500
Iteration: 270, Loss: 518365.687500
Iteration: 280, Loss: 518367.468750
Iteration: 290, Loss: 518367.218750
Iteration: 300, Loss: 518367.250000
Iteration: 310, Loss: 518367.093750
Iteration: 320, Loss: 518366.750000
Iteration: 330, Loss: 518366.906250
Iteration: 340, Loss: 518367.343750
Iteration: 350, Loss: 518367.312500
Iteration: 360, Loss: 518367.281250
Iteration: 370, Loss: 518367.375000
Iteration: 380, Loss: 518367.281250
Iteration: 390, Loss: 518367.281250
Iteration: 400, Loss: 518367.187500
Iteration: 410, Loss: 518367.312500
Iteration: 420, Loss: 518367.218750
Iteration: 430, Loss: 518367.281250
Iteration: 440, Loss: 518367.218750
Iteration: 450, Loss: 518367.281250
Iteration: 460, Loss: 518367.250000
Iteration: 470, Loss: 518367.218750
Iteration: 480, Loss: 518367.218750
Iteration: 490, Loss: 518367.187500
Iteration: 500, Loss: 518367.218750
Iteration: 510, Loss: 518367.250000
Iteration: 520, Loss: 518367.218750
Iteration: 530, Loss: 518367.218750
Iteration: 540, Loss: 518367.187500
Iteration: 550, Loss: 518367.218750
Iteration: 560, Loss: 518367.187500
Iteration: 570, Loss: 518367.187500
```

```
Iteration: 580, Loss: 518367.218750
Iteration: 590, Loss: 518367.218750
Iteration: 600, Loss: 518367.250000
Iteration: 610, Loss: 518367.187500
Iteration: 620, Loss: 518367.250000
Iteration: 630, Loss: 518367.218750
Iteration: 640, Loss: 518367.218750
Iteration: 650, Loss: 518367.281250
Iteration: 660, Loss: 518367.218750
Iteration: 670, Loss: 518367.156250
Iteration: 680, Loss: 518367.218750
Iteration: 690, Loss: 518367.218750
Iteration: 700, Loss: 518367.218750
Iteration: 710, Loss: 518367.250000
Iteration: 720, Loss: 518367.250000
Iteration: 730, Loss: 518367.187500
Iteration: 740, Loss: 518367.250000
Iteration: 750, Loss: 518367.187500
Iteration: 760, Loss: 518367.250000
Iteration: 770, Loss: 518367.218750
Iteration: 780, Loss: 518367.218750
Iteration: 790, Loss: 518367.218750
Iteration: 800, Loss: 518367.218750
Elapsed time: 0:00:50.560072
```

Note: `n_components != 2` have not been thoroughly tested.

```
X is normalized
PaCMAP(n_neighbors=26, n_MN=13, n_FP=52, distance=euclidean, lr=1.0, n_iters=(10
0, 100, 600), apply_pca=True, opt_method='adam', verbose=True, intermediate=False,
seed=None)
Finding pairs
Found nearest neighbor
Calculated sigma
Found scaled dist
Pairs sampled successfully.
((2959892, 2), (1479946, 2), (5919784, 2))
Initial Loss: 3658542.0
Iteration: 10, Loss: 2462471.250000
Iteration: 20, Loss: 2168747.250000
Iteration: 30, Loss: 2027108.000000
Iteration: 40, Loss: 1920317.000000
Iteration: 50, Loss: 1820339.375000
Iteration: 60, Loss: 1715066.750000
Iteration: 70, Loss: 1597465.500000
Iteration: 80, Loss: 1458511.500000
Iteration: 90, Loss: 1283235.000000
Iteration: 100, Loss: 1019827.000000
Iteration: 110, Loss: 1338709.875000
Iteration: 120, Loss: 1314226.625000
Iteration: 130, Loss: 1302655.000000
Iteration: 140, Loss: 1298621.000000
Iteration: 150, Loss: 1297661.875000
Iteration: 160, Loss: 1297670.625000
Iteration: 170, Loss: 1297710.250000
Iteration: 180, Loss: 1298473.000000
Iteration: 190, Loss: 1299416.625000
Iteration: 200, Loss: 1299907.250000
Iteration: 210, Loss: 549504.375000
Iteration: 220, Loss: 543379.875000
Iteration: 230, Loss: 538801.687500
Iteration: 240, Loss: 536142.812500
Iteration: 250, Loss: 533946.437500
Iteration: 260, Loss: 532219.437500
Iteration: 270, Loss: 530960.875000
Iteration: 280, Loss: 529975.500000
Iteration: 290, Loss: 529150.562500
Iteration: 300, Loss: 528282.187500
Iteration: 310, Loss: 527335.875000
Iteration: 320, Loss: 526335.562500
Iteration: 330, Loss: 525393.625000
Iteration: 340, Loss: 524565.250000
Iteration: 350, Loss: 523874.250000
Iteration: 360, Loss: 523341.718750
Iteration: 370, Loss: 522950.468750
Iteration: 380, Loss: 522681.093750
Iteration: 390, Loss: 522464.968750
Iteration: 400, Loss: 522235.750000
Iteration: 410, Loss: 521987.750000
Iteration: 420, Loss: 521706.531250
Iteration: 430, Loss: 521428.281250
Iteration: 440, Loss: 521167.281250
Iteration: 450, Loss: 520922.718750
Iteration: 460, Loss: 520701.250000
Iteration: 470, Loss: 520493.500000
Iteration: 480, Loss: 520307.843750
Iteration: 490, Loss: 520134.531250
```

```
Iteration: 500, Loss: 519955.812500
Iteration: 510, Loss: 519771.156250
Iteration: 520, Loss: 519592.312500
Iteration: 530, Loss: 519416.937500
Iteration: 540, Loss: 519239.625000
Iteration: 550, Loss: 519066.750000
Iteration: 560, Loss: 518899.062500
Iteration: 570, Loss: 518758.531250
Iteration: 580, Loss: 518629.656250
Iteration: 590, Loss: 518521.250000
Iteration: 600, Loss: 518437.312500
Iteration: 610, Loss: 518373.375000
Iteration: 620, Loss: 518325.937500
Iteration: 630, Loss: 518289.000000
Iteration: 640, Loss: 518256.312500
Iteration: 650, Loss: 518230.093750
Iteration: 660, Loss: 518207.968750
Iteration: 670, Loss: 518198.218750
Iteration: 680, Loss: 518201.750000
Iteration: 690, Loss: 518203.500000
Iteration: 700, Loss: 518214.750000
Iteration: 710, Loss: 518231.156250
Iteration: 720, Loss: 518261.593750
Iteration: 730, Loss: 518287.906250
Iteration: 740, Loss: 518299.687500
Iteration: 750, Loss: 518308.062500
Iteration: 760, Loss: 518306.593750
Iteration: 770, Loss: 518298.812500
Iteration: 780, Loss: 518291.906250
Iteration: 790, Loss: 518283.406250
Iteration: 800, Loss: 518277.343750
Elapsed time: 182.54s
X is normalized.
X is normalized.
Found nearest neighbor
(740012, 2)
Initial Loss: 1388304.25
Iteration: 10, Loss: 719653.062500
Iteration: 20, Loss: 357892.656250
Iteration: 30, Loss: 257328.578125
Iteration: 40, Loss: 221728.609375
Iteration: 50, Loss: 212586.328125
Iteration: 60, Loss: 209357.421875
Iteration: 70, Loss: 207909.843750
Iteration: 80, Loss: 207547.921875
Iteration: 90, Loss: 207347.859375
Iteration: 100, Loss: 207288.437500
Iteration: 110, Loss: 312695.906250
Iteration: 120, Loss: 311715.656250
Iteration: 130, Loss: 311469.562500
Iteration: 140, Loss: 311377.750000
Iteration: 150, Loss: 311325.468750
Iteration: 160, Loss: 311306.062500
Iteration: 170, Loss: 311308.937500
Iteration: 180, Loss: 311308.406250
Iteration: 190, Loss: 311327.875000
Iteration: 200, Loss: 311359.250000
Iteration: 210, Loss: 103737.781250
Iteration: 220, Loss: 103656.578125
Iteration: 230, Loss: 103639.296875
```

```
Iteration: 240, Loss: 103632.257812
Iteration: 250, Loss: 103629.937500
Iteration: 260, Loss: 103628.726562
Iteration: 270, Loss: 103628.601562
Iteration: 280, Loss: 103628.570312
Iteration: 290, Loss: 103628.640625
Iteration: 300, Loss: 103628.445312
Iteration: 310, Loss: 103628.406250
Iteration: 320, Loss: 103628.382812
Iteration: 330, Loss: 103628.453125
Iteration: 340, Loss: 103628.570312
Iteration: 350, Loss: 103628.515625
Iteration: 360, Loss: 103628.523438
Iteration: 370, Loss: 103628.531250
Iteration: 380, Loss: 103628.515625
Iteration: 390, Loss: 103628.515625
Iteration: 400, Loss: 103628.515625
Iteration: 410, Loss: 103628.523438
Iteration: 420, Loss: 103628.515625
Iteration: 430, Loss: 103628.523438
Iteration: 440, Loss: 103628.515625
Iteration: 450, Loss: 103628.515625
Iteration: 460, Loss: 103628.515625
Iteration: 470, Loss: 103628.523438
Iteration: 480, Loss: 103628.515625
Iteration: 490, Loss: 103628.515625
Iteration: 500, Loss: 103628.515625
Iteration: 510, Loss: 103628.515625
Iteration: 520, Loss: 103628.515625
Iteration: 530, Loss: 103628.515625
Iteration: 540, Loss: 103628.523438
Iteration: 550, Loss: 103628.515625
Iteration: 560, Loss: 103628.515625
Iteration: 570, Loss: 103628.515625
Iteration: 580, Loss: 103628.515625
Iteration: 590, Loss: 103628.523438
Iteration: 600, Loss: 103628.523438
Iteration: 610, Loss: 103628.523438
Iteration: 620, Loss: 103628.515625
Iteration: 630, Loss: 103628.515625
Iteration: 640, Loss: 103628.515625
Iteration: 650, Loss: 103628.523438
Iteration: 660, Loss: 103628.515625
Iteration: 670, Loss: 103628.507812
Iteration: 680, Loss: 103628.515625
Iteration: 690, Loss: 103628.515625
Iteration: 700, Loss: 103628.515625
Iteration: 710, Loss: 103628.515625
Iteration: 720, Loss: 103628.515625
Iteration: 730, Loss: 103628.523438
Iteration: 740, Loss: 103628.523438
Iteration: 750, Loss: 103628.523438
Iteration: 760, Loss: 103628.515625
Iteration: 770, Loss: 103628.515625
Iteration: 780, Loss: 103628.515625
Iteration: 790, Loss: 103628.515625
Iteration: 800, Loss: 103628.515625
Elapsed time: 0:00:11.773395
X is normalized.
X is normalized.
```

```
Found nearest neighbor  
(3705078, 2)  
Initial Loss: 7107333.5  
Iteration: 10, Loss: 3588380.500000  
Iteration: 20, Loss: 1795610.625000  
Iteration: 30, Loss: 1296906.000000  
Iteration: 40, Loss: 1107943.000000  
Iteration: 50, Loss: 1060182.875000  
Iteration: 60, Loss: 1043774.625000  
Iteration: 70, Loss: 1038894.187500  
Iteration: 80, Loss: 1037617.125000  
Iteration: 90, Loss: 1037004.937500  
Iteration: 100, Loss: 1036823.625000  
Iteration: 110, Loss: 1544797.875000  
Iteration: 120, Loss: 1537778.750000  
Iteration: 130, Loss: 1536001.625000  
Iteration: 140, Loss: 1535614.875000  
Iteration: 150, Loss: 1535297.000000  
Iteration: 160, Loss: 1535143.500000  
Iteration: 170, Loss: 1535239.625000  
Iteration: 180, Loss: 1535370.250000  
Iteration: 190, Loss: 1535358.750000  
Iteration: 200, Loss: 1535523.000000  
Iteration: 210, Loss: 518701.968750  
Iteration: 220, Loss: 518450.875000  
Iteration: 230, Loss: 518393.250000  
Iteration: 240, Loss: 518370.500000  
Iteration: 250, Loss: 518364.375000  
Iteration: 260, Loss: 518362.625000  
Iteration: 270, Loss: 518363.281250  
Iteration: 280, Loss: 518364.781250  
Iteration: 290, Loss: 518364.718750  
Iteration: 300, Loss: 518364.093750  
Iteration: 310, Loss: 518363.906250  
Iteration: 320, Loss: 518364.281250  
Iteration: 330, Loss: 518364.375000  
Iteration: 340, Loss: 518364.000000  
Iteration: 350, Loss: 518364.187500  
Iteration: 360, Loss: 518364.093750  
Iteration: 370, Loss: 518364.125000  
Iteration: 380, Loss: 518364.187500  
Iteration: 390, Loss: 518364.187500  
Iteration: 400, Loss: 518364.156250  
Iteration: 410, Loss: 518364.218750  
Iteration: 420, Loss: 518364.187500  
Iteration: 430, Loss: 518364.187500  
Iteration: 440, Loss: 518364.187500  
Iteration: 450, Loss: 518364.187500  
Iteration: 460, Loss: 518364.187500  
Iteration: 470, Loss: 518364.156250  
Iteration: 480, Loss: 518364.187500  
Iteration: 490, Loss: 518364.187500  
Iteration: 500, Loss: 518364.218750  
Iteration: 510, Loss: 518364.187500  
Iteration: 520, Loss: 518364.187500  
Iteration: 530, Loss: 518364.187500  
Iteration: 540, Loss: 518364.187500  
Iteration: 550, Loss: 518364.187500  
Iteration: 560, Loss: 518364.218750  
Iteration: 570, Loss: 518364.187500
```

```

Iteration: 580, Loss: 518364.187500
Iteration: 590, Loss: 518364.187500
Iteration: 600, Loss: 518364.125000
Iteration: 610, Loss: 518364.187500
Iteration: 620, Loss: 518364.218750
Iteration: 630, Loss: 518364.187500
Iteration: 640, Loss: 518364.156250
Iteration: 650, Loss: 518364.218750
Iteration: 660, Loss: 518364.218750
Iteration: 670, Loss: 518364.156250
Iteration: 680, Loss: 518364.218750
Iteration: 690, Loss: 518364.187500
Iteration: 700, Loss: 518364.156250
Iteration: 710, Loss: 518364.156250
Iteration: 720, Loss: 518364.156250
Iteration: 730, Loss: 518364.187500
Iteration: 740, Loss: 518364.218750
Iteration: 750, Loss: 518364.187500
Iteration: 760, Loss: 518364.125000
Iteration: 770, Loss: 518364.156250
Iteration: 780, Loss: 518364.218750
Iteration: 790, Loss: 518364.187500
Iteration: 800, Loss: 518364.156250
Elapsed time: 0:00:51.773778

```

```
In [21]: ## pacmap과 isolation forest 3차 이용(2)
val_score_3 = f1_score(ori_val_df['Class'], np.round(val_pred_set_3), average='macro')
print(f'Validation F1 Score : [{val_score_3}]')
print(classification_report(ori_val_df['Class'], np.round(val_pred_set_3)))
print(confusion_matrix(ori_val_df['Class'], np.round(val_pred_set_3)))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	28432
1	0.89	0.80	0.84	30
accuracy			1.00	28462
macro avg	0.94	0.90	0.92	28462
weighted avg	1.00	1.00	1.00	28462

```
[[28429      3]
 [     6     24]]
```

```
In [22]: set(test_pred_set_3)
```

```
Out[22]: {np.float64(0.0),
 np.float64(0.14285714285714285),
 np.float64(0.42857142857142855),
 np.float64(1.0)}
```

```
In [23]: ## pacmap과 isolation forest 3차 이용(3)
# 2차 저장
chujung_train_2 = pd.DataFrame({'Class':np.round(train_pred_set_3)})
chujung_val_2 = pd.DataFrame({'Class':np.round(val_pred_set_3)})
chujung_test_2 = pd.DataFrame({'Class':np.round(test_pred_set_3)})

result_train_2 = pd.concat([train_df,chujung_train_2], axis=1)
result_val_2 = pd.concat([val_df,chujung_val_2], axis=1)
```

```
result_test_2 = pd.concat([test_df, chujung_test_2], axis=1)

result_train_2.to_csv('result_train_2.csv', index=False)
result_val_2.to_csv('result_val_2.csv', index=False)
result_test_2.to_csv('result_test_2.csv', index=False)
```

In [25]:

```
## pacmap과 isolation forest 3차 이용(4)
# 2차 불러오기 및 outlier된 dataset 모음
train_pred_set_3 = np.array(pd.read_csv('result_train_2.csv')['Class'])
val_pred_set_3 = np.array(pd.read_csv('result_val_2.csv')['Class'])
test_pred_set_3 = np.array(pd.read_csv('result_test_2.csv')['Class'])

one_train_df = train_df.iloc[np.where(np.round(train_pred_set_3) == 1)[0]]
one_val_df = ori_val_df.iloc[np.where(np.round(val_pred_set_3) == 1)[0]]
one_test_df = test_df.iloc[np.where(np.round(test_pred_set_3) == 1)[0]]
```

저는 이에 더해 판별된 outlier들 중에 가짜를 도출하기로 생각했습니다. 역시 저는 변수를 선택하였고 validation dataset의 통계정보를 이용하였습니다. 이번엔 다른 방식으로 판별하고자 합니다. 개인적으로 저는 isolation forest를 사용함에도 표출이 안 된것은 특정 변수하에서는 가짜 outlier들이 진짜 outlier안에 숨어있다고 생각하였습니다. 그에 따라 KernelPCA를 이용해서 양쪽에 있는 가짜 outlier를 빼내고, pacmap을 사용한 다음, pca를 통해 좌표를 바꾸고, 극단값 중 몇몇개를 뽑아서 가짜 outlier라고 생각하였습니다. 우선, Validation dataset 내 ouliter로 판별된 것 중에서 실제 outlier들의 중위값과 가짜 outlier들의 중위값 검정 비교하였고, Validation dataset 내 ouliter로 판별된 것 중에서 실제 outlier들의 분산과 가짜 outlier 비율 체크해서 변수들을 확인하였습니다.

In [26]:

```
## outlier 중 가짜 판별(1)
# 변수 선택

rrr_df = []
for what_val in range(30):
    rrr_df.append(ranksums(one_val_df.iloc[np.where(one_val_df['Class'] == 1)[0]],
                           one_val_df.iloc[np.where(one_val_df['Class'] == 0)[0]]))

rrr_idx = np.where(np.array(rrr_df) > 0.05)[0]

qqq_df = []
for what_val in rrr_idx:
    qqq_df.append(one_val_df.iloc[np.where(one_val_df['Class'] == 0)[0], what_val])

new_born_idx = np.argsort(qqq_df)[:5]
```

In [27]:

```
print(new_born_idx)
```

[20 23 14 3 18]

다음으로 가짜 판별하는데 있어서 위에서 이야기한 그대로 바탕으로 진행하였습니다. 가짜 outlier인지 아닌지 판별하는데 있어서 Validation set의 통계정보와 wilcoxon rank sum test를 이용하였습니다.

In [28]:

```
## outlier 중 가짜 판별(2)
# 판별
transformer = KernelPCA(n_components=5, kernel='rbf')
hhhhh = new_born_idx
can_sepearate = np.argsort(rrr_df)[0]

X_transformed = transformer.fit_transform(one_train_df.iloc[:, hhhh])
```

```

X1_transformed = transformer.transform(one_val_df.iloc[:,hhhhh])
X2_transformed = transformer.transform(one_test_df.iloc[:,hhhhh])

# kernel pca 후 1에서 0 찾기 by pacmap 그후 판별
hhmm = 201
skip_count = 0

contamin = 0.0725
making_set = 0

for num in range(hhmm):
    embedding1010 = pacmap.PaCMAP(n_components=3, n_neighbors=None, MN_ratio=0.5)
    pacmac_train1010 = embedding1010.fit_transform(X_transformed, init="pca")
    pacmac_val1010 = embedding1010.transform(X1_transformed, basis=X_transformed)
    pacmac_test1010 = embedding1010.transform(X2_transformed, basis=X_transformed)

    # 그냥 pca
    pcawow = PCA(n_components = 3) # 주성분을 몇개로 할지 결정
    pca_train_wow = pcawow.fit_transform(pacmac_train1010)
    pca_train_wow = pd.DataFrame(data=pca_train_wow)

    pca_val_wow = pcawow.transform(pacmac_val1010)
    pca_val_wow = pd.DataFrame(data=pca_val_wow)

    pca_test_wow = pcawow.transform(pacmac_test1010)
    pca_test_wow = pd.DataFrame(data=pca_test_wow)

    rlwns = int(contamin*len(pca_train_wow))
    thtn = contamin*len(pca_train_wow) - rlwns

    if making_set == 0:
        max_part_zero = 0
        max_part_one = 0
        max_what = 100

        # PC 축을 설정하고 좌 또는 우 극단값을 찾고 해당하는 중위값이 실제 outlier
        for jrj in range(6):
            val_pred_set_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_val_wow))})

            if jrj % 2 == 0:
                rlwns_val = pca_train_wow.iloc[:,(jrj//2)].nlargest(rlwns).iloc[0]
                rlwns_val_Qkd = pca_train_wow.iloc[:,(jrj//2)].nlargest((rlwns+1)).iloc[0]

                val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] >= rlwns_val)] = 0
                val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] <= rlwns_val_Qkd)] = 0

            else:
                rlwns_val = pca_train_wow.iloc[:,(jrj//2)].nsmallest(rlwns).iloc[0]
                rlwns_val_Qkd = pca_train_wow.iloc[:,(jrj//2)].nsmallest((rlwns+1)).iloc[0]

                val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] <= rlwns_val)] = 0
                val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] >= rlwns_val_Qkd)] = 0

            ranksum_pval_ed_one = ranksums(one_val_df.iloc[np.where(one_val_df['Class'] == 1)], axis=0)
            ranksum_pval_ed_zero = ranksums(one_val_df.iloc[np.where(one_val_df['Class'] == 0)], axis=0)

            if np.isnan(ranksum_pval_ed_zero):
                continue

            if max_part_zero <= ranksum_pval_ed_zero:
                max_part_zero = ranksum_pval_ed_zero
            if max_part_one <= ranksum_pval_ed_one:
                max_part_one = ranksum_pval_ed_one

```

```

if max_part_zero >= 0.5:
    if max_part_one >= 0.5:
        max_what = jrj
        break

if max_what == 100:
    skip_count += 1
    continue
making_set += 10

# 체크 후 해당하는 축으로 가짜 outlier 판별
val_pred_set_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_val_wow))}

if max_what % 2 == 0:
    rlwns_val = pca_train_wow.iloc[:,(max_what//2)].nlargest(rlwns).iloc[0]
    rlwns_val_Qkd = pca_train_wow.iloc[:,(max_what//2)].nlargest((rlwns+1)).iloc[0]
    val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(max_what//2)] >= rlwns_val)] = 1
    val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(max_what//2)] < rlwns_val_Qkd)] = 0
else:
    rlwns_val = pca_train_wow.iloc[:,(max_what//2)].nsmallest(rlwns).iloc[0]
    rlwns_val_Qkd = pca_train_wow.iloc[:,(max_what//2)].nsmallest((rlwns+1)).iloc[0]
    val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(max_what//2)] <= rlwns_val)] = 1
    val_pred_set_1010.iloc[np.where(pca_val_wow.iloc[:,(max_what//2)] > rlwns_val_Qkd)] = 0

train_pred_set_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_train_wow))}

if max_what % 2 == 0:
    train_pred_set_1010.iloc[np.where(pca_train_wow.iloc[:,(max_what//2)] >= rlwns_val)] = 1
    train_pred_set_1010.iloc[np.where(pca_train_wow.iloc[:,(max_what//2)] < rlwns_val_Qkd)] = 0
else:
    train_pred_set_1010.iloc[np.where(pca_train_wow.iloc[:,(max_what//2)] <= rlwns_val)] = 1
    train_pred_set_1010.iloc[np.where(pca_train_wow.iloc[:,(max_what//2)] > rlwns_val_Qkd)] = 0

test_pred_set_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_test_wow))}

if max_what % 2 == 0:
    test_pred_set_1010.iloc[np.where(pca_test_wow.iloc[:,(max_what//2)] >= rlwns_val)] = 1
    test_pred_set_1010.iloc[np.where(pca_test_wow.iloc[:,(max_what//2)] < rlwns_val_Qkd)] = 0
else:
    test_pred_set_1010.iloc[np.where(pca_test_wow.iloc[:,(max_what//2)] <= rlwns_val)] = 1
    test_pred_set_1010.iloc[np.where(pca_test_wow.iloc[:,(max_what//2)] > rlwns_val_Qkd)] = 0

else:
    max_part_zero = 0
    max_part_one = 0
    max_what = 100

# PC 축을 설정하고 좌 또는 우 극단값을 찾고 해당하는 중위값이 실제 outlier인지를 판별
for jrj in range(6):
    val_pred_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_val_wow))}

    if jrj % 2 == 0:
        rlwns_val = pca_train_wow.iloc[:,(jrj//2)].nlargest(rlwns).iloc[0]
        rlwns_val_Qkd = pca_train_wow.iloc[:,(jrj//2)].nlargest((rlwns+1)).iloc[0]
        val_pred_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] >= rlwns_val)] = 1
        val_pred_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] < rlwns_val_Qkd)] = 0
    else:
        rlwns_val = pca_train_wow.iloc[:,(jrj//2)].nsmallest(rlwns).iloc[0]
        rlwns_val_Qkd = pca_train_wow.iloc[:,(jrj//2)].nsmallest((rlwns+1)).iloc[0]
        val_pred_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] <= rlwns_val)] = 1
        val_pred_1010.iloc[np.where(pca_val_wow.iloc[:,(jrj//2)] > rlwns_val_Qkd)] = 0

```

```

ranksum_pval_ed_one = ranksums(one_val_df.iloc[np.where(one_val_df['
ranksum_pval_ed_zero = ranksums(one_val_df.iloc[np.where(one_val_df['

if np.isnan(ranksum_pval_ed_zero):
    continue

if max_part_zero <= ranksum_pval_ed_zero:
    max_part_zero = ranksum_pval_ed_zero
if max_part_one <= ranksum_pval_ed_one:
    max_part_one = ranksum_pval_ed_one

if max_part_zero >= 0.5:
    if max_part_one >= 0.5:
        max_what = jrz
        break

if max_what == 100:
    skip_count += 1
    continue

# 체크 후 해당하는 축으로 가짜 outlier 판별
val_pred_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_val_wow))})

if max_what % 2 == 0:
    rlwns_val = pca_train_wow.iloc[:,(max_what//2)].nlargest(rlwns).iloc
    rlwns_val_Qkd = pca_train_wow.iloc[:,(max_what//2)].nlargest((rlwns+

    val_pred_1010.iloc[np.where(pca_val_wow.iloc[:,(max_what//2)] >= (rl
else:
    rlwns_val = pca_train_wow.iloc[:,(max_what//2)].nsmallest(rlwns).iloc
    rlwns_val_Qkd = pca_train_wow.iloc[:,(max_what//2)].nsmallest((rlwns

    val_pred_1010.iloc[np.where(pca_val_wow.iloc[:,(max_what//2)] <= (rl

ranksum_pval_ed_one = ranksums(one_val_df.iloc[np.where(one_val_df['Clas
ranksum_pval_ed_zero = ranksums(one_val_df.iloc[np.where(one_val_df['Clas

val_pred_set_1010 = val_pred_set_1010 + val_pred_1010

train_pred_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_train_wow))}

if max_what % 2 == 0:
    train_pred_1010.iloc[np.where(pca_train_wow.iloc[:,(max_what//2)] >
else:
    train_pred_1010.iloc[np.where(pca_train_wow.iloc[:,(max_what//2)] <=

train_pred_set_1010 = train_pred_set_1010 + train_pred_1010

test_pred_1010 = pd.DataFrame({'Class':np.repeat(1,len(pca_test_wow))})

if max_what % 2 == 0:
    test_pred_1010.iloc[np.where(pca_test_wow.iloc[:,(max_what//2)] >=
else:
    test_pred_1010.iloc[np.where(pca_test_wow.iloc[:,(max_what//2)] <=

test_pred_set_1010 = test_pred_set_1010 + test_pred_1010

train_pred_set_1010 = train_pred_set_1010/(hhmm- skip_count)
val_pred_set_1010 = val_pred_set_1010/(hhmm-skip_count)
test_pred_set_1010 = test_pred_set_1010/(hhmm-skip_count)

```



```
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
Note: `n_components != 2` have not been thoroughly tested.
```

```
In [32]: ## 최종 확인
# voting해서 50% 이상 1이면 1로, 반대면 0으로
final_one_train_df = one_train_df.iloc[np.where(np.round(train_pred_set_1010['Class']) >= 0.5)]
final_one_val_df = one_val_df.iloc[np.where(np.round(val_pred_set_1010['Class']) >= 0.5)]
final_one_test_df = one_test_df.iloc[np.where(np.round(test_pred_set_1010['Class']) >= 0.5)]

# validation score 확인
chujung_train2 = pd.DataFrame({'Class':np.repeat(0,len(train_df))})
chujung_val2 = pd.DataFrame({'pre_Class':np.repeat(0,len(val_df))})
chujung_test2 = pd.DataFrame({'Class':np.repeat(0,len(test_df))})

chujung_train2.iloc[final_one_train_df.index,0] = 1
chujung_val2.iloc[final_one_val_df.index,0] = 1
chujung_test2.iloc[final_one_test_df.index,0] = 1

sec_result_train = pd.concat([train_df,chujung_train2], axis=1)
sec_result_val = pd.concat([val_df,chujung_val2], axis=1)
sec_result_test = pd.concat([test_df,chujung_test2], axis=1)

print('F1-score',f1_score(sec_result_val['pre_Class'], val_class, average='macro'))
print(confusion_matrix(sec_result_val['pre_Class'], val_class))
print(classification_report(sec_result_val['pre_Class'], val_class))

# 저장
submit = pd.read_csv('sample_submission.csv')
submit['Class'] = 0

submit.iloc[final_one_test_df.index,1] = 1
submit.iloc[final_one_test_df.index,1]

submit.to_csv('result_submit.csv', index=False)
```

```
F1-score 0.9443916925541683
[[28432      6]
 [    0     24]]
precision    recall   f1-score   support
          0       1.00      1.00      1.00      28438
          1       0.80      1.00      0.89       24
accuracy                           1.00      28462
macro avg       0.90      1.00      0.94      28462
weighted avg    1.00      1.00      1.00      28462
```

In []: