EVAD_CWRU.nbconvert

November 16, 2023

[38]:
```
from tqdm import tqdm
import os
import pandas as pd
import numpy as np
from sklearn.svm import OneClassSVM
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import plotly.express as px
from sklearn.preprocessing import RobustScaler
from collections import Counter
from matplotlib import pyplot as plt
plt.rcParams["figure.figsize"] = (10,10)


from sklearn.decomposition import PCA

from he_svm import preprocess_a_sample, he_svm, preprocess_a_sample_encrypted
import glob
```

[39]:
```
healthy_csvs = ['data/CWRU/1_csv/Nominal/MotorLoad0.csv',
                'data/CWRU/1_csv/Nominal/MotorLoad1.csv',
                'data/CWRU/1_csv/Nominal/MotorLoad2.csv',
                'data/CWRU/1_csv/Nominal/MotorLoad3.csv']

LEN_SAMPLES = 500

train_samples = []
for f in healthy_csvs:
    df = pd.read_csv(f)
    df = df.iloc[:, :]
    dfs = df.groupby(np.arange(len(df))//LEN_SAMPLES)
    [train_samples.append(t[1]) for t in list(dfs)[:-1]]
```

[40]:
```
len(train_samples)
```

[40]: 3395

[41]:
```
len(train_samples[0])
```

[41]: 500

1 Train a SVM

[42]:
```
def preprocess_a_sample(df, windows):
    final_sample = []

    for column in df.columns:
        signal = df.loc[:, column]

        signal_fft = np.abs(np.fft.rfft(signal))**2
        len_windows = int(len(signal_fft) / windows) - 1

        for i in range(windows):
            if i == windows-1:
                final_sample.append(np.mean(signal_fft[i*len_windows:]))
            else:
                final_sample.append(np.mean(signal_fft[i*len_windows:
 ↪(i+1)*len_windows]))

    return np.array(final_sample)
```

[44]:
```
windows = 10  # As suggested by the Elbow method in Data_Exploration_CWRU.ipynb!
```

[45]:
```
preprocessed_samples_nominal = np.array([preprocess_a_sample(sample, windows)␣
 ↪for sample in train_samples])
np.random.seed(42)
np.random.shuffle(preprocessed_samples_nominal)

n = int(len(preprocessed_samples_nominal) * 0.8)
preprocessed_samples_train = preprocessed_samples_nominal[:n]
preprocessed_samples_test = preprocessed_samples_nominal[n:]

svm = OneClassSVM(nu=0.05, kernel='poly', gamma='scale', degree=2)
svm.fit(preprocessed_samples_train)
svm.gamma_value = 1 / ((windows*len(df.columns)) * preprocessed_samples_train.
 ↪var())  # to put gamma value in svm
```

2 Test

2.1 Test on nominal data

[46]:
```
x_predicted = svm.predict(preprocessed_samples_train)
print(f"################")
print(f"Training samples: {len(x_predicted)}")
print(f"Found nominal: {len([x for x in x_predicted if x == 1])}")
```

```
print(f"Found anomalous: {len([x for x in x_predicted if x == -1])}")
print(f"Accuracy nominal training: {len([x for x in x_predicted if x == 1]) /␣
 ↪len(x_predicted)}")

x_predicted = svm.predict(preprocessed_samples_test)
print(f"Test samples (nominal): {len(x_predicted)}")
print(f"Found nominal: {len([x for x in x_predicted if x == 1])}")
print(f"Found anomalous: {len([x for x in x_predicted if x == -1])}")
print(f"Accuracy nominal testing: {len([x for x in x_predicted if x == 1]) /␣
 ↪len(x_predicted)}")
```

```
###############
Training samples: 2716
Found nominal: 2580
Found anomalous: 136
Accuracy nominal training: 0.9499263622974963
Test samples (nominal): 679
Found nominal: 645
Found anomalous: 34
Accuracy nominal testing: 0.9499263622974963
```

[47]:
```
print(f'Max distance: {max(svm.decision_function(preprocessed_samples_train))}')
print(f'Min distance: {min(svm.decision_function(preprocessed_samples_train))}')
```

```
Max distance: 176.3770333425292
Min distance: -16.336863124723052
```

3 Test on anomalous data

[48]:
```
# Define the directory
dir = 'data/CWRU/1_csv/12k_Drive/'

# Get the list of files in the directory and its subdirectories
files = []
for f in glob.glob(dir + '**/*.csv', recursive=True):
    files.append(f)
```

[49]:
```
files
```

[49]:
```
['data/CWRU/1_csv/12k_Drive/0_014_outer/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_outer/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_outer/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_outer/Motor2.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_inner/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_inner/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_inner/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_inner/Motor2.csv',
```

```
 'data/CWRU/1_csv/12k_Drive/0_007_outer/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_outer/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_outer/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_outer/Motor2.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_ball/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_ball/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_ball/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_ball/Motor2.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_ball/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_ball/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_ball/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_ball/Motor2.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_inner/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_inner/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_inner/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_inner/Motor2.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_ball/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_ball/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_ball/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_014_ball/Motor2.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_inner/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_inner/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_inner/Motor0.csv',
 'data/CWRU/1_csv/12k_Drive/0_007_inner/Motor2.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_outer/Motor1.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_outer/Motor3.csv',
 'data/CWRU/1_csv/12k_Drive/0_021_outer/Motor2.csv']
```

[50]:
```
sides = ['inner', 'ball', 'outer']
powers = ['0', '1', '2', '3']

multi_index = pd.MultiIndex.from_product([sides, powers], names=['Sides',␣
 ↪'Powers'])
table_accuracy = pd.DataFrame(columns = multi_index)

for f in sorted(files):
    print(f"File: {f}")

    if 'ball' in f:
        side = f[32:36]
        power = f[-5]
        depth = f[26:31]
    else:
        side = f[32:37]
        power = f[-5]
        depth = f[26:31]
```

```
    df = pd.read_csv(f)
    df = df.iloc[:, :]
    dfs = df.groupby(np.arange(len(df))//LEN_SAMPLES)
    anomalous_samples = [t[1] for t in list(dfs)[:-1]]
    anomaly_samples = np.array([preprocess_a_sample(df, windows) for df in␣
 ↪anomalous_samples])

    x_predicted = svm.predict(anomaly_samples)

    n_nom = len([x for x in x_predicted if x == 1])
    n_ano = len([x for x in x_predicted if x == -1])
    accuracy = round((n_ano / len(x_predicted)) * 100, 2)
    table_accuracy.loc[depth, (str(side), str(power))] = accuracy

    print(f"Test samples (anomalous): {len(x_predicted)}")
    print(f"Found nominal: {n_nom}")
    print(f"Found anomalous: {n_ano}")
    print(f"Accuracy: {accuracy}%")
    print(f"################")

display(table_accuracy)
```

```
File: data/CWRU/1_csv/12k_Drive/0_007_ball/Motor0.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
################
File: data/CWRU/1_csv/12k_Drive/0_007_ball/Motor1.csv
Test samples (anomalous): 241
Found nominal: 9
Found anomalous: 232
Accuracy: 96.27%
################
File: data/CWRU/1_csv/12k_Drive/0_007_ball/Motor2.csv
Test samples (anomalous): 243
Found nominal: 0
Found anomalous: 243
Accuracy: 100.0%
################
File: data/CWRU/1_csv/12k_Drive/0_007_ball/Motor3.csv
Test samples (anomalous): 242
Found nominal: 1
Found anomalous: 241
Accuracy: 99.59%
################
File: data/CWRU/1_csv/12k_Drive/0_007_inner/Motor0.csv
Test samples (anomalous): 243
```

```
Found nominal: 241
Found anomalous: 2
Accuracy: 0.82%
###############
File: data/CWRU/1_csv/12k_Drive/0_007_inner/Motor1.csv
Test samples (anomalous): 242
Found nominal: 235
Found anomalous: 7
Accuracy: 2.89%
###############
File: data/CWRU/1_csv/12k_Drive/0_007_inner/Motor2.csv
Test samples (anomalous): 242
Found nominal: 242
Found anomalous: 0
Accuracy: 0.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_007_inner/Motor3.csv
Test samples (anomalous): 243
Found nominal: 243
Found anomalous: 0
Accuracy: 0.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_007_outer/Motor0.csv
Test samples (anomalous): 241
Found nominal: 0
Found anomalous: 241
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_007_outer/Motor1.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_007_outer/Motor2.csv
Test samples (anomalous): 243
Found nominal: 0
Found anomalous: 243
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_007_outer/Motor3.csv
Test samples (anomalous): 243
Found nominal: 0
Found anomalous: 243
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_ball/Motor0.csv
Test samples (anomalous): 244
```

```
Found nominal: 5
Found anomalous: 239
Accuracy: 97.95%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_ball/Motor1.csv
Test samples (anomalous): 243
Found nominal: 0
Found anomalous: 243
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_ball/Motor2.csv
Test samples (anomalous): 244
Found nominal: 0
Found anomalous: 244
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_ball/Motor3.csv
Test samples (anomalous): 242
Found nominal: 1
Found anomalous: 241
Accuracy: 99.59%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_inner/Motor0.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_inner/Motor1.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_inner/Motor2.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_inner/Motor3.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_outer/Motor0.csv
Test samples (anomalous): 243
```

```
Found nominal: 0
Found anomalous: 243
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_outer/Motor1.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_outer/Motor2.csv
Test samples (anomalous): 241
Found nominal: 63
Found anomalous: 178
Accuracy: 73.86%
###############
File: data/CWRU/1_csv/12k_Drive/0_014_outer/Motor3.csv
Test samples (anomalous): 241
Found nominal: 233
Found anomalous: 8
Accuracy: 3.32%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_ball/Motor0.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_ball/Motor1.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_ball/Motor2.csv
Test samples (anomalous): 243
Found nominal: 0
Found anomalous: 243
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_ball/Motor3.csv
Test samples (anomalous): 241
Found nominal: 0
Found anomalous: 241
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_inner/Motor0.csv
Test samples (anomalous): 242
```

```
Found nominal: 56
Found anomalous: 186
Accuracy: 76.86%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_inner/Motor1.csv
Test samples (anomalous): 242
Found nominal: 14
Found anomalous: 228
Accuracy: 94.21%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_inner/Motor2.csv
Test samples (anomalous): 241
Found nominal: 27
Found anomalous: 214
Accuracy: 88.8%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_inner/Motor3.csv
Test samples (anomalous): 242
Found nominal: 116
Found anomalous: 126
Accuracy: 52.07%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_outer/Motor1.csv
Test samples (anomalous): 241
Found nominal: 0
Found anomalous: 241
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_outer/Motor2.csv
Test samples (anomalous): 241
Found nominal: 0
Found anomalous: 241
Accuracy: 100.0%
###############
File: data/CWRU/1_csv/12k_Drive/0_021_outer/Motor3.csv
Test samples (anomalous): 242
Found nominal: 0
Found anomalous: 242
Accuracy: 100.0%
###############
```

Sides	inner				ball				outer		\
Powers	0	1	2	3	0	1	2	3	0	1	
0_007	0.82	2.89	0.0	0.0	100.0	96.27	100.0	99.59	100.0	100.0	
0_014	100.0	100.0	100.0	100.0	97.95	100.0	100.0	99.59	100.0	100.0	
0_021	76.86	94.21	88.8	52.07	100.0	100.0	100.0	100.0	NaN	100.0	

Sides

```
Powers      2      3
0_007   100.0  100.0
0_014   73.86   3.32
0_021   100.0  100.0
```

4 Test on encrypted data

[51]:
```
windows
```

[51]:
```
16
```

[52]:
```
from linetimer import CodeTimer
import tenseal as ts
np.set_printoptions(precision=3, suppress=True)

poly_modulus_degree=2**14
coeff_mod_bit_sizes=[60] + [50]*6 + [60]

# Setup TenSEAL context
context = ts.context(
            ts.SCHEME_TYPE.CKKS,
            poly_modulus_degree=poly_modulus_degree,
            coeff_mod_bit_sizes=coeff_mod_bit_sizes
          )
context.generate_galois_keys()
context.global_scale = 2**50

errors = {}

for f in files:
    error_df = pd.DataFrame(columns=['ID', 'Expected', 'Predicted (enc)', '%␣
 ↪Error', 'Correct?', 'Time enc (s)'])
    df = pd.read_csv(f)
    dfs = df.groupby(np.arange(len(df))//LEN_SAMPLES)
    anomalous_samples = [t[1] for t in list(dfs)[:-1]]

    for sample in anomalous_samples[:]:
        anomaly_sample = preprocess_a_sample(sample, windows).reshape(1, -1)
        x_expected = svm.decision_function(anomaly_sample)[0]

        enc_time = CodeTimer(silent=True, unit='s')
        with enc_time:
            x_enc_preprocessed = preprocess_a_sample_encrypted(sample, context,␣
 ↪windows, None)
            x_enc_predicted = he_svm(x_enc_preprocessed, svm, windows)
            x_predicted = x_enc_predicted[0].decrypt()[0]
```

```
        error_df.loc[len(error_df)] = [sample.index.name, x_expected,␣
↪x_predicted,
                                       (x_expected-x_predicted)/x_expected,
                                       np.sign(x_expected) == np.
↪sign(x_predicted),
                                       enc_time.took]


    if 'ball' in f:
        side = f[32:36]
        power = f[-5]
        depth = f[26:31]
    else:
        side = f[32:37]
        power = f[-5]
        depth = f[26:31]

    f = f"{depth}_{side}_{power}"
    errors[f] = error_df
    error_df.to_csv(f"results/CWRU/Errors_{f}.csv")

    print(f"Sensor: {f}")
    print(error_df)
```

```
[[  0.943   8.571   1.615   5.248   2.147   0.447   2.901   3.227  16.988
   46.814  13.865   0.514   2.09    2.228   0.49    0.517   4.278   2.507
    4.334  29.385   3.494   3.948   3.961   7.846 286.172 371.983  78.914
   10.209   9.464   0.544   2.67    3.834]]
[0.9427639695714443, 8.571251607913307, 1.6147657209378194, 5.248438862518514,
2.146516415356513, 0.4470566966000583, 2.901215371899691, 3.2273812677090548,
16.988421839018095, 46.81356703935209, 13.865154157612773, 0.5142814642628125,
2.0904673549587147, 2.22835318441847, 0.4903127110818944, 0.5170279636183698,
4.278070417827061, 2.5066864072943016, 4.333760821482279, 29.385313130770022,
3.4938935614974436, 3.9480989330930516, 3.9606041818767923, 7.846011739183354,
286.172367076659, 371.98346223122894, 78.91387924662119, 10.209199924183423,
9.464350207120205, 0.5440768757573048, 2.6697313729954737, 3.8340173169181506]
Sensor: Motor1
     ID   Expected  Predicted (enc)       % Error  Correct?  Time enc (s)
0  None -35.971521       -35.971521  1.893220e-08      True     11.636309
```

[53]:

[ ]:

[ ]:

[17]: