

A Senior Falling Alert System Embedded in Smart Wristband

Chen Boyi 119010010

Li Zihan 119010167

Zeng Zhuoru 119010417



Part 1 Introduction

A Senior Falling Alert System Embedded in Smart Wristband

Target:
Detect falling action of elderly people
and raise an alert.





Part 2 DataSet

DataSet Information

- Dataset:
 - UCI Machine Learning Repository: Simulated Falls and Daily Living Activities Data Set (Ȧzdemir, Barshan, 2014).
 - <http://archive.ics.uci.edu/ml/datasets/Simulated+Falls+and+Daily+Living+Activities+Data+Set>
- Dataset Formation
 - Contains activities performed by 17 volunteers.
 - Each volunteer performs 16 normal activities (NON-FALL) and 20 falls (FALL) with five or six repetitions.
 - Each activity contains a series of data captured by the right wrist sensor in the rate of 25.0Hz.
- Data Used in Project
 - Focus on 6 data fields namely, "Acc_X", "Acc_Y", "Acc_Z", "Gyr_X", "Gyr_Y" & "Gyr_Z", (acceleration and gyroscope information in three dimensions)



Data Selection

Mix data from different action categories to form the two class (FALL & NON-FALL).

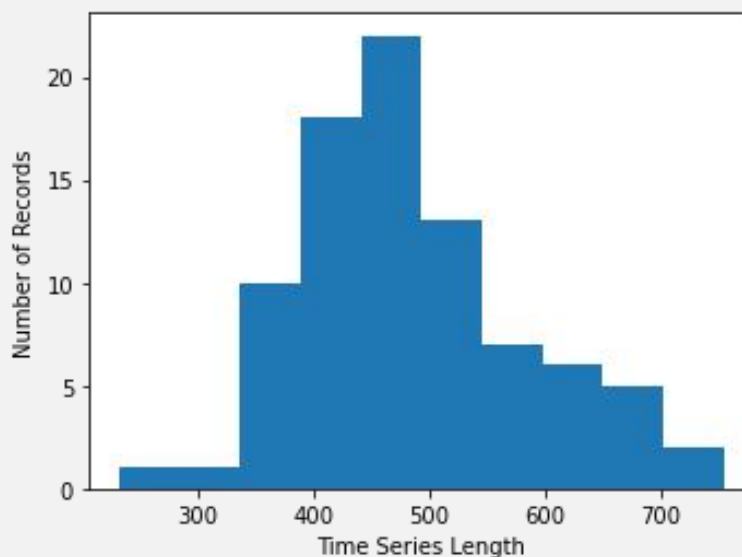
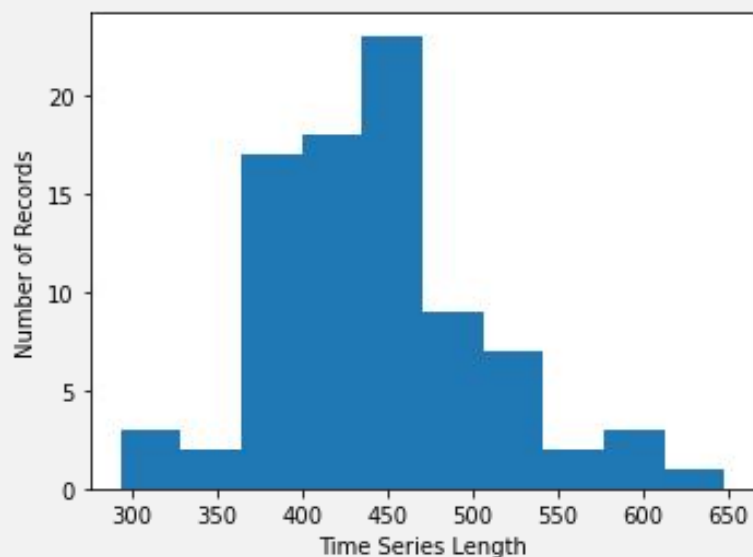
- FALL
 - Vertical falling forward; Falling and recovery; Falling ending sitting; Falling ending lying ...
- NON-FALL
 - Walking forward; Running; Bending to pick up an object; Lying on the bed ...

Data Analysis

Data Feature

- Multivariate (“Acc_X”, “Acc_Y”, “Acc_Z”, “Gyr_X”, “Gyr_Y” & “Gyr_Z”)
- Varying length
- Time Sereis

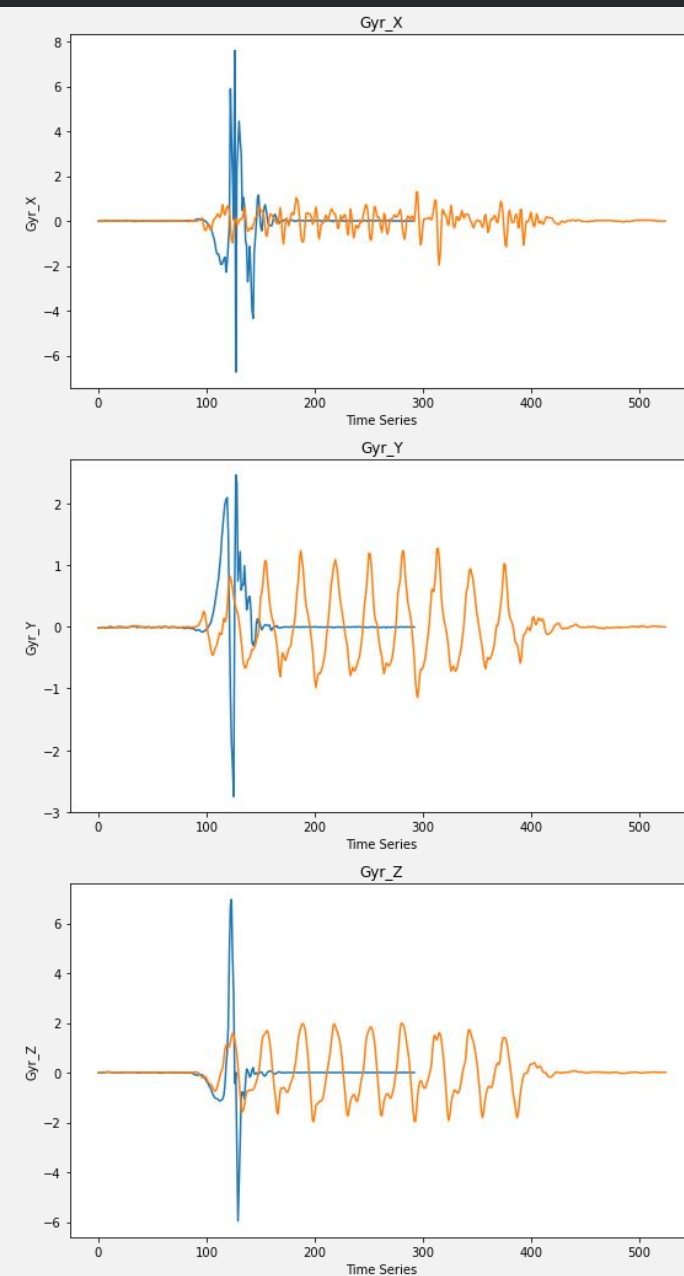
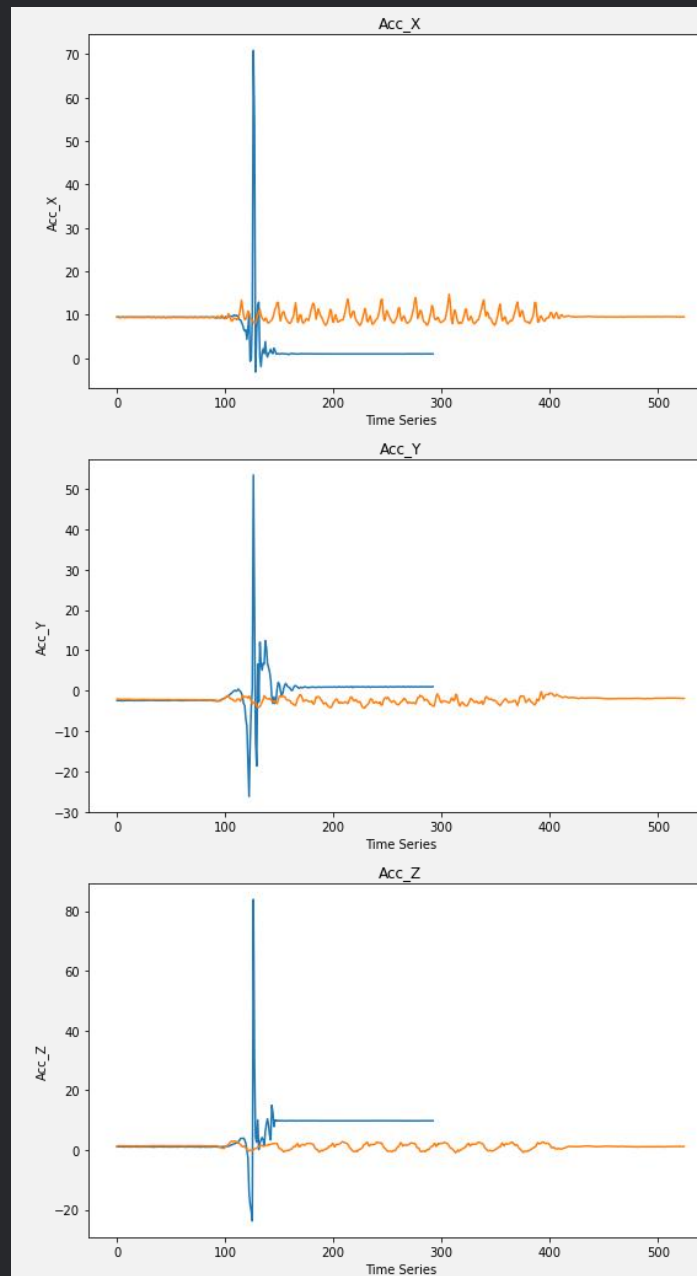
Length Distribution of FALL & NON-FALL Sample



The Problem

Problem: Multivariate Variable-length Time Series Classification (TSC)

- Classify a time series sample with multiple variables and varying length to FALL & NON-FALL Class





Three Approach to Solve the Problem

Approach 1:

Reform Problem to Non Time Series Classification - SVM

Approach 2:

Time Series Classification Algorithms on Fixed Length Time Series
- RocketClassifier & HiveCoteV2

Approach 3:

Time Series Classification Algorithms on Varying Length Time
Series - KNN & DTW



Part 3 Model Implementation

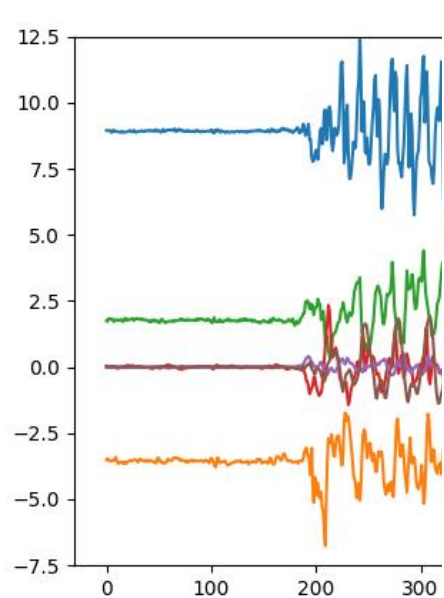
TSC Approach 1: Reform Problem to Non Time Series Classification - SVM

Data Processing – Add label to every time stamp :

- Too large (10 cases have over 250,000 rows)
- Not accurate (non-falling wrongly labeled as falling)

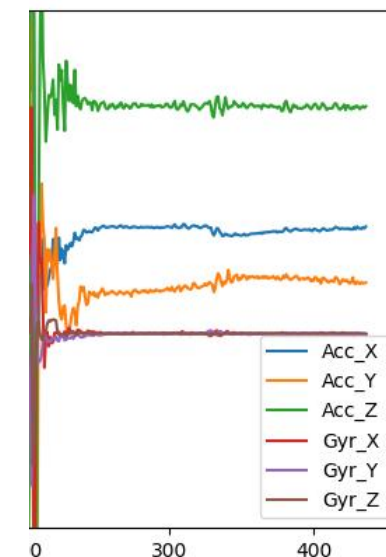
index	0	1	2	3	4	5	6
0	8.949280	-3.509521	1.754761	0.040436	-0.011587	0.009727	0
1	8.933473	-3.459883	1.722622	0.010109	-0.007057	0.014496	0
2	8.935547	-3.492737	1.828003	0.004196	-0.000143	0.015450	0
3	8.948612	-3.551388	1.830053	0.012732	-0.003529	0.005150	0
4	8.926392	-3.558350	1.800537	0.021935	-0.006437	0.005722	0
...
259494	-0.594378	-2.276421	9.757018	-0.000191	-0.001240	0.007391	1
259495	-0.617981	-2.253723	9.780884	0.004196	-0.004721	-0.001144	1
259496	-0.609922	-2.256703	9.747791	0.001049	-0.002527	-0.000954	1
259497	-0.592041	-2.278137	9.800720	0.005531	-0.007772	0.002098	1
259498	-0.586128	-2.268314	9.776974	0.004005	0.002289	0.004721	1

Not the whole time series, but the max absolute value matters



	0	1	2	3	4	5	6
0	12.921762	6.753969	4.403806	2.336028	0.509660	2.480800	0
1	19.526458	33.610535	9.181213	3.071411	2.766372	4.233532	0
2	10.003662	9.175110	4.983521	1.060008	0.810435	1.284449	0
3	9.934306	10.493469	8.091044	3.016995	0.836159	1.061924	0
4	17.669678	6.201172	11.788940	2.314435	3.727930	2.752292	0
...
165	12.376404	5.217624	43.204403	1.878733	4.945130	1.215626	1
166	11.665344	16.033936	38.397217	1.616141	4.258256	1.782907	1
167	17.840648	34.515810	13.679004	4.879740	2.089246	4.068235	1
168	18.516541	14.079285	20.065689	6.456118	5.152426	4.764977	1
169	16.297913	14.173889	33.699036	6.876566	4.811998	5.622021	1

[170 rows x 7 columns]




Time series of non-falling

Time series of falling

Data after processing

```
temp = []
for i in range(len(vals)):
    Acc_Xs.append(abs(vals[i][0]))
    Acc_Ys.append(abs(vals[i][1]))
    Acc_Zs.append(abs(vals[i][2]))
    Gyr_Xs.append(abs(vals[i][3]))
    Gyr_Ys.append(abs(vals[i][4]))
    Gyr_Zs.append(abs(vals[i][5]))
temp.append(max(Acc_Xs))
temp.append(max(Acc_Ys))
temp.append(max(Acc_Zs))
temp.append(max(Gyr_Xs))
temp.append(max(Gyr_Ys))
temp.append(max(Gyr_Zs))
if (kind in noFalls):
    temp.append(0)
elif(kind in falls):
    temp.append(1)
else: continue
x.append(temp)
```

```
clf = svm.SVC(kernel="linear")
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
print('split done')
clf.fit(x_train, y_train)
print('train done')
```



TSC Approach 2: Time Series Classification Algorithms on Fixed Length Time Series - RocketClassifier & HiveCoteV2

Data Preprocessing: 5 Approach

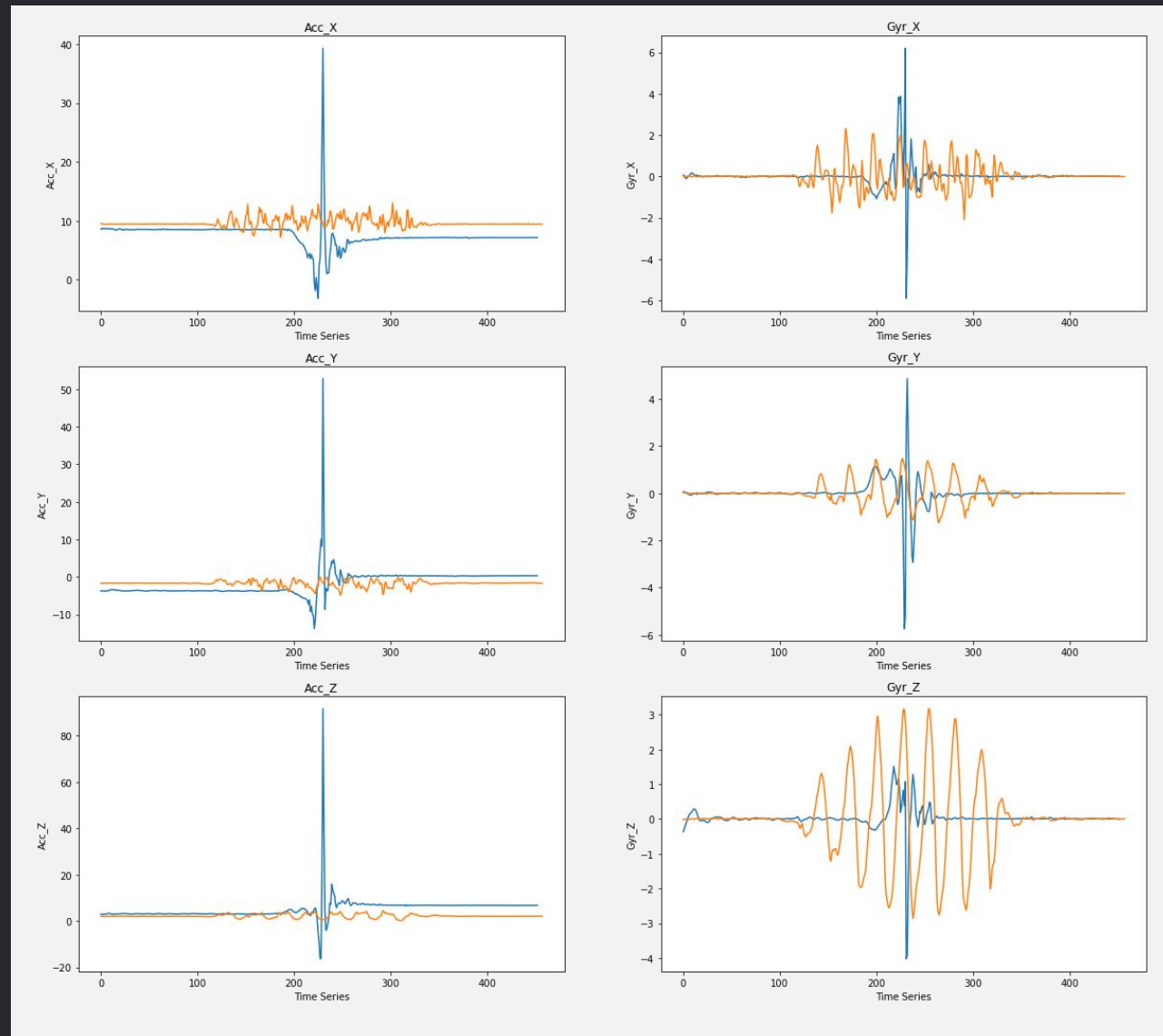
- Truncate Data to Fixed Length
- Suffix Sample Padding with Mean
- Prefix & Suffix Sample Padding with Mean
- Uniform Scaling
- Sample Padding with Regression Prediction

Training Model: 2 Approach

- RocketClassifier
- HiveCoteV2

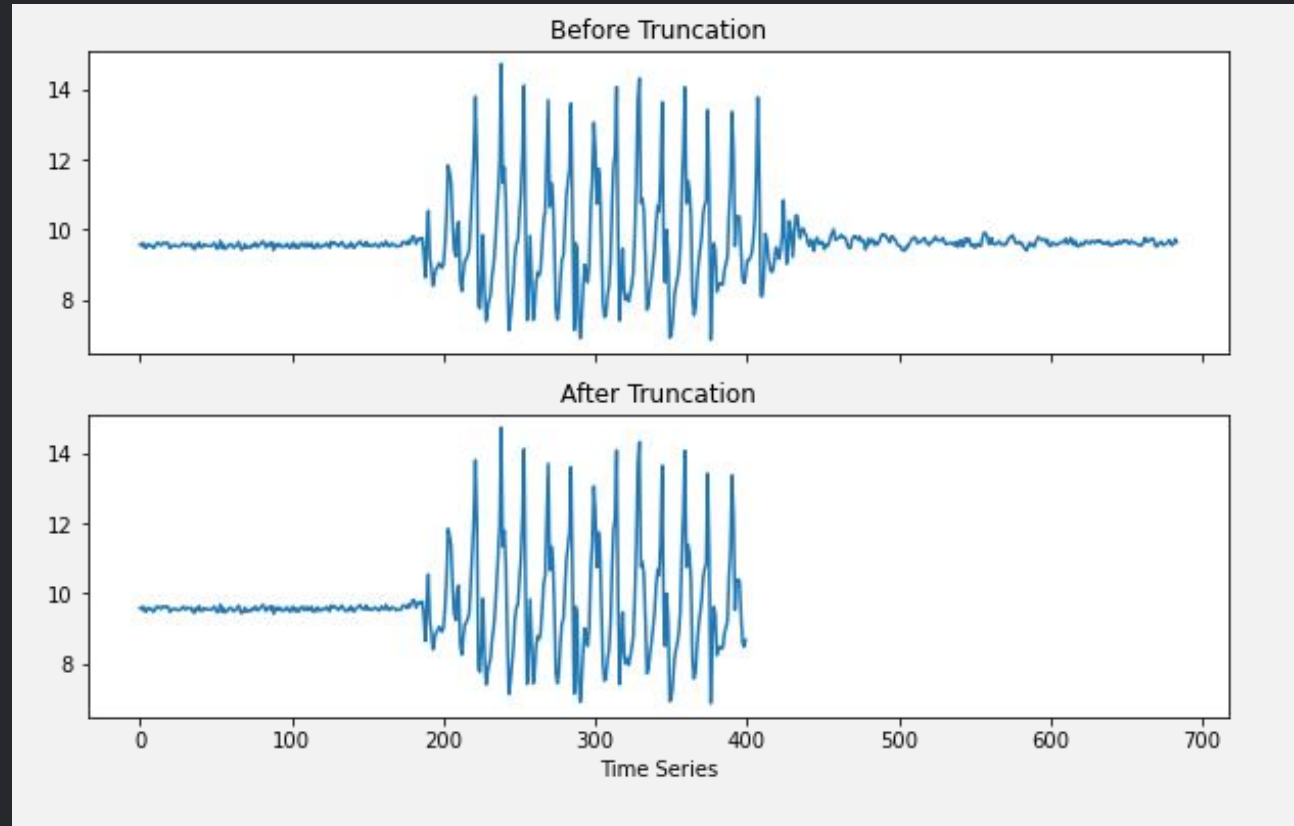
Data Preprocessing - Ideal Data

Ideal Data:
Time Series of the
same length



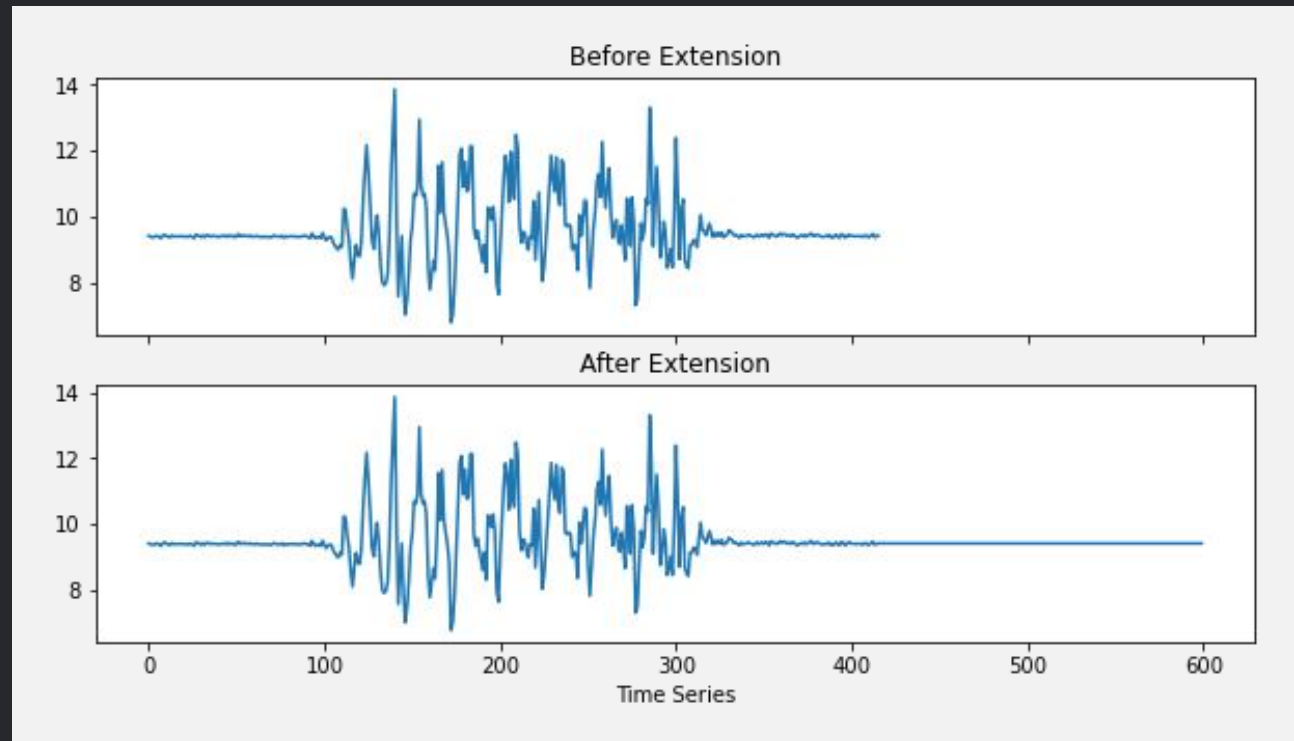
Data Preprocessing - Approach 1

Approach 1:
Truncate Data to
Fixed Length



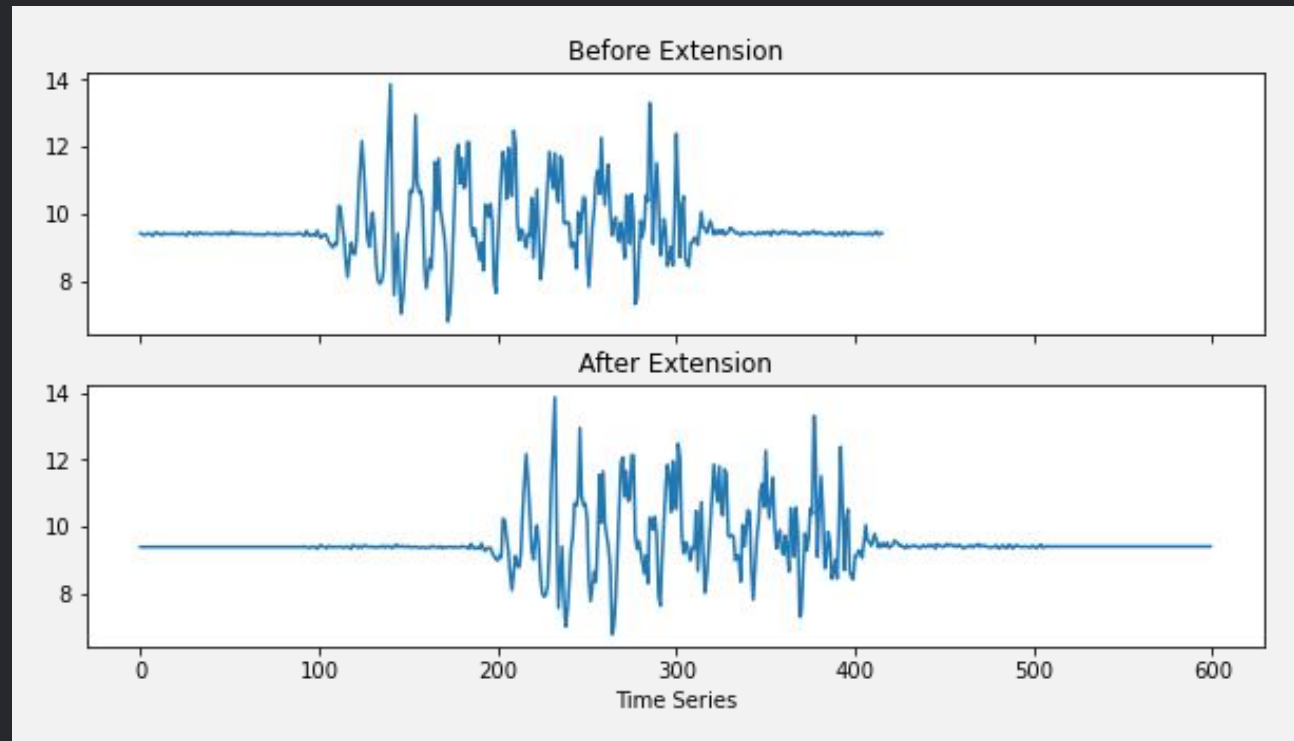
Data Preprocessing - Approach 2

Approach 2:
Suffix Sample
Padding with Mean



Data Preprocessing - Approach 3

Approach 3:
Suffix & Prefix
Sample Padding with
Mean



Data Preprocessing - Approach 4

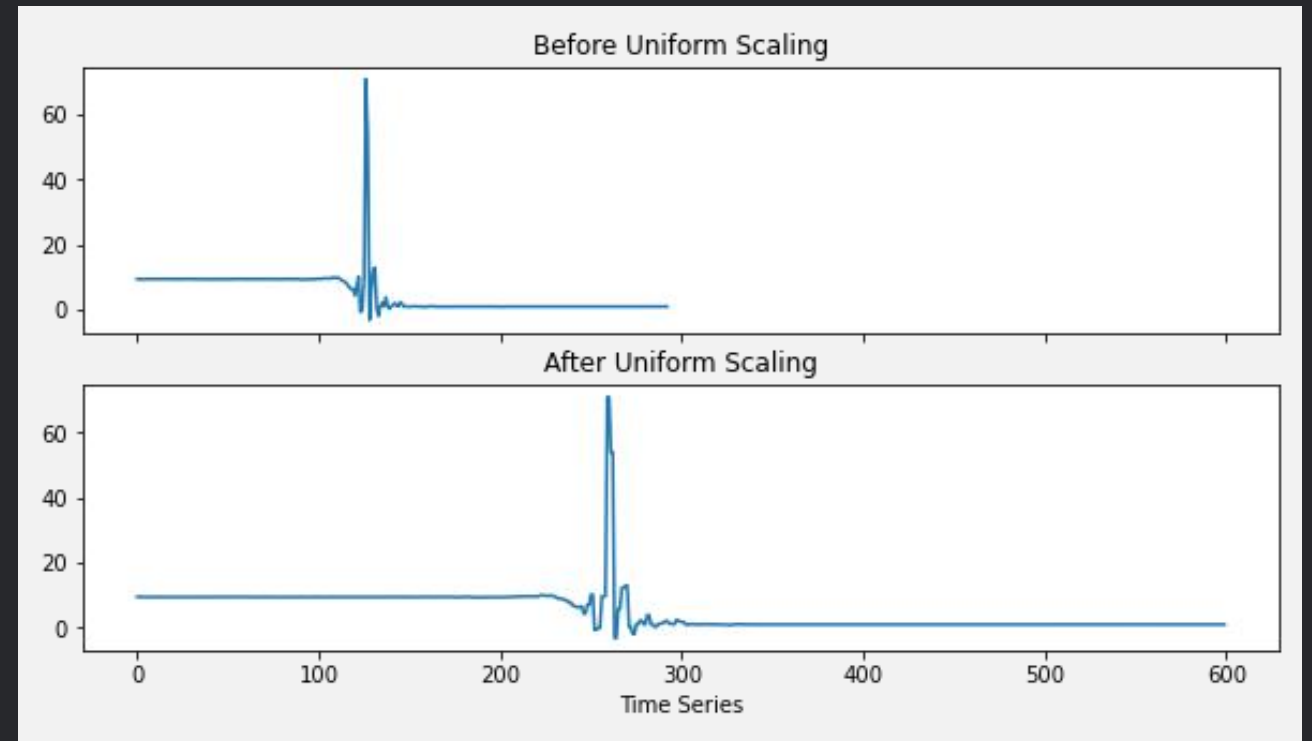
Approach 4: Uniform Scaling (Keogh, E., 2003)

Q is a time series with length n

$$Q = q_1, q_1, \dots, q_i, \dots, q_n$$

To scale time series Q to produce a new time series QP of length p, the formula is:

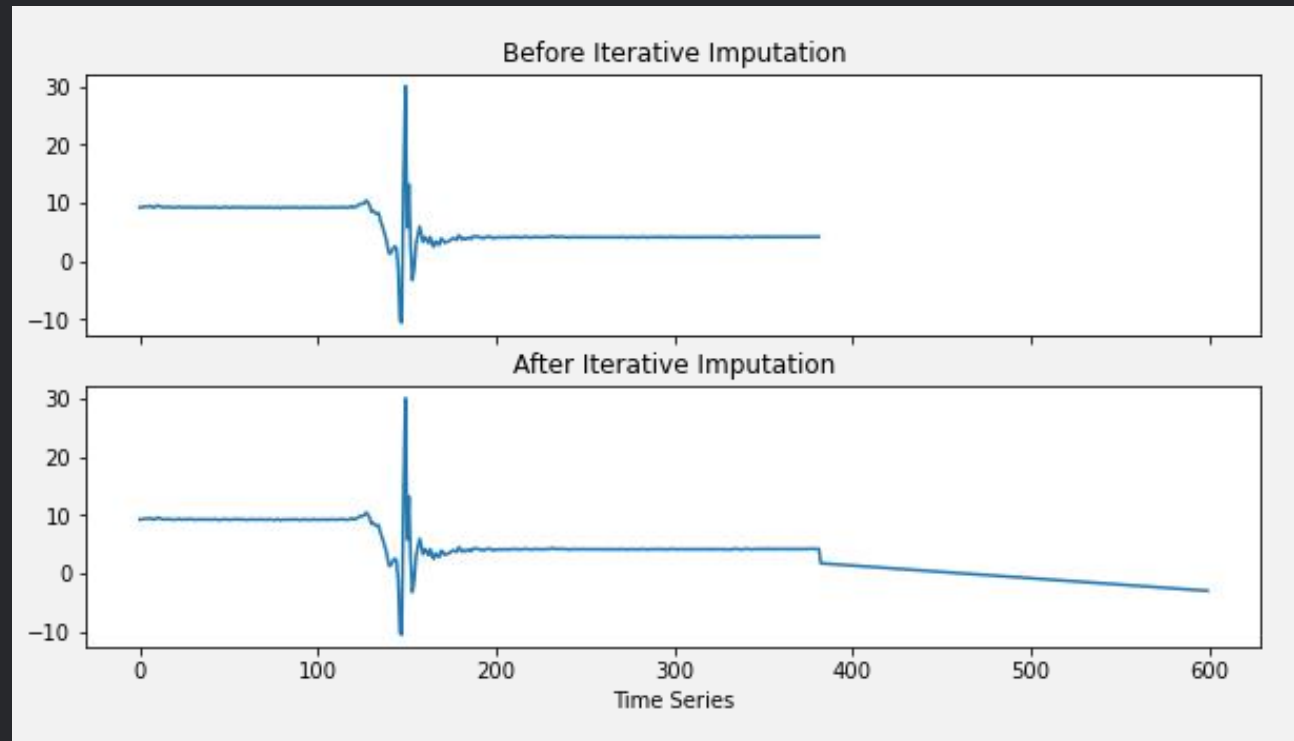
$$QP_i = Q_{\left\lceil j \cdot \frac{n}{p} \right\rceil}, 1 \leq j \leq p$$



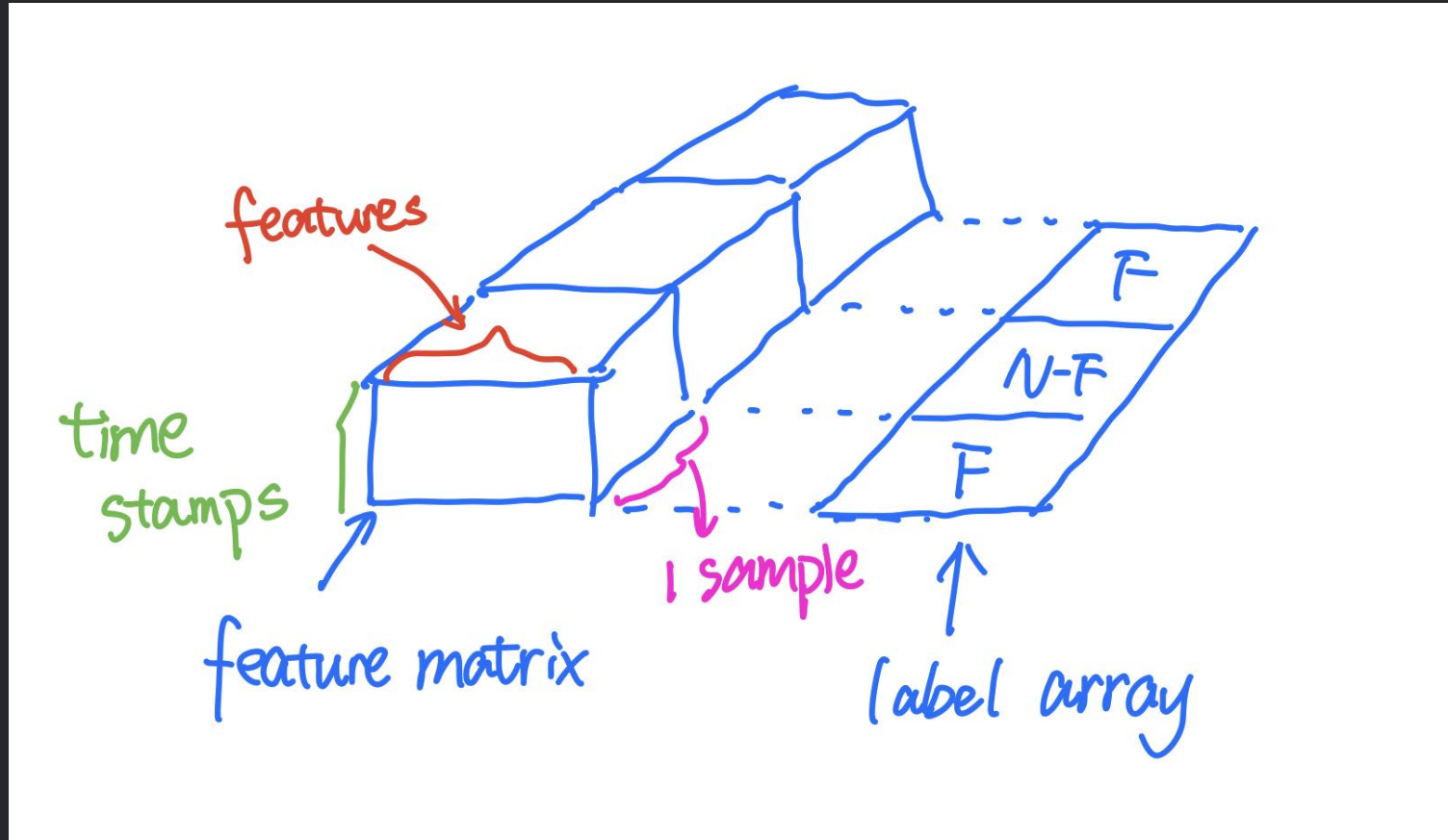
Data Preprocessing - Approach 5

Approach 5: Sample Padding with Regression Prediction

- Scikit-Learn
package Iterative
Imputer



Data Preprocessing - Model Input



Model Training

RocketClassifier

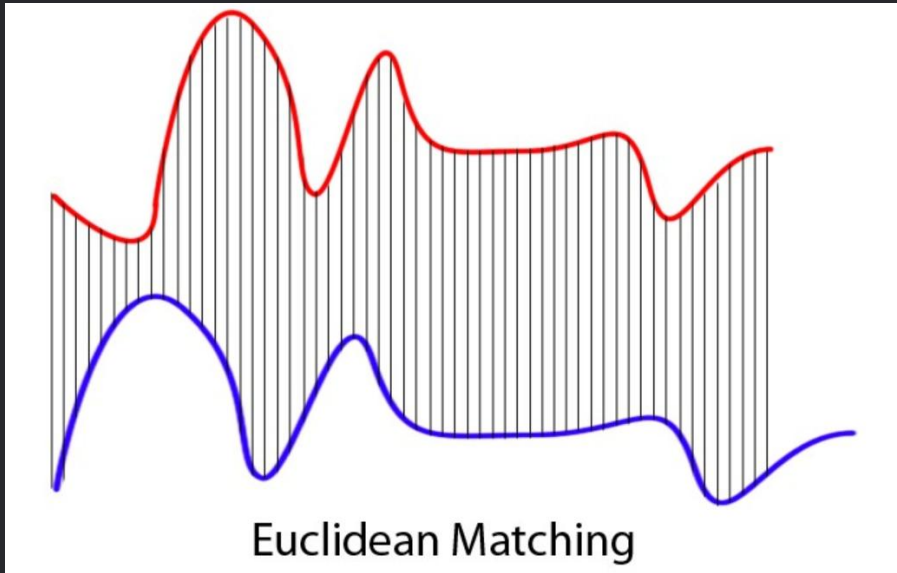
- Classifier wrapped for the ROCKET transformer using RidgeClassifierCV (Dempster et al., 2019)
- Sktime RocketClassifier Package

HiveCoteV2

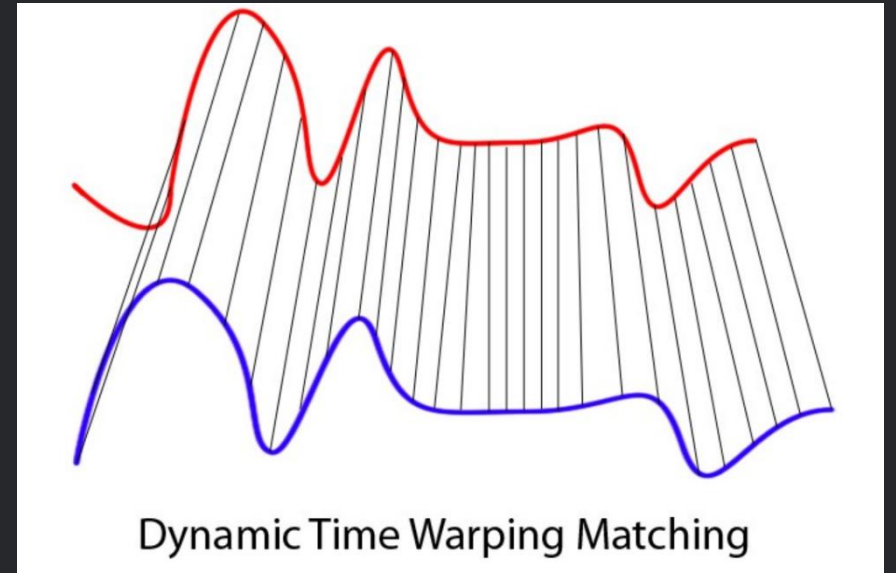
- An ensemble of the STC, DrCIF, Arsenal and TDE classifiers from different feature representations using the CAWPE structure (Middlehurst et al., 2021)
- Sktime HiveCoteV2 Package

TSC Approach 3: Time Series Classification Algorithms on Varying Length Time Series

- KNN & DTW



Traditional KNN



KNN & DTW

TSC Approach 3: Time Series Classification Algorithms on Varying Length Time Series

- KNN & DTW

Single feature:

$$D[i][j] = d(y_i, x_j) + \min(D[i-1][j-1], D[i-1][j], D[i][j-1]), \quad d(y_i, x_j) = |y_i - x_j|$$

Multiple feature:

$$D[i][j] = d(y_i, x_j) + \min(D[i-1][j-1], D[i-1][j], D[i][j-1]), \quad d(y_i, x_j) = \sum (y_{in} - x_{jn})^2$$

Matrix D (m*n):

$$\text{distance} = D[m-1][n-1]$$




Part 4 Results

TSC Approach 1: Reform Problem to Non Time Series Classification - SVM

	$Y' = 1$	$Y' = 0$
$Y = 1$	100%	0
$Y = 0$	15.79%	84.21%

Accuracy: 93.02%



TSC Approach 2: Time Series Classification Algorithms on Fixed Length Time Series - RocketClassifier & HiveCoteV2

Model Evaluation

Representation Shorthand

Preprocess Method	Shorthand Notation
Truncate Data to Fixed Length	P1
Suffix Padding	P2
Suffix and Prefix Padding	P3
Uniform Scaling	P4
Regression Prediction Padding	P5

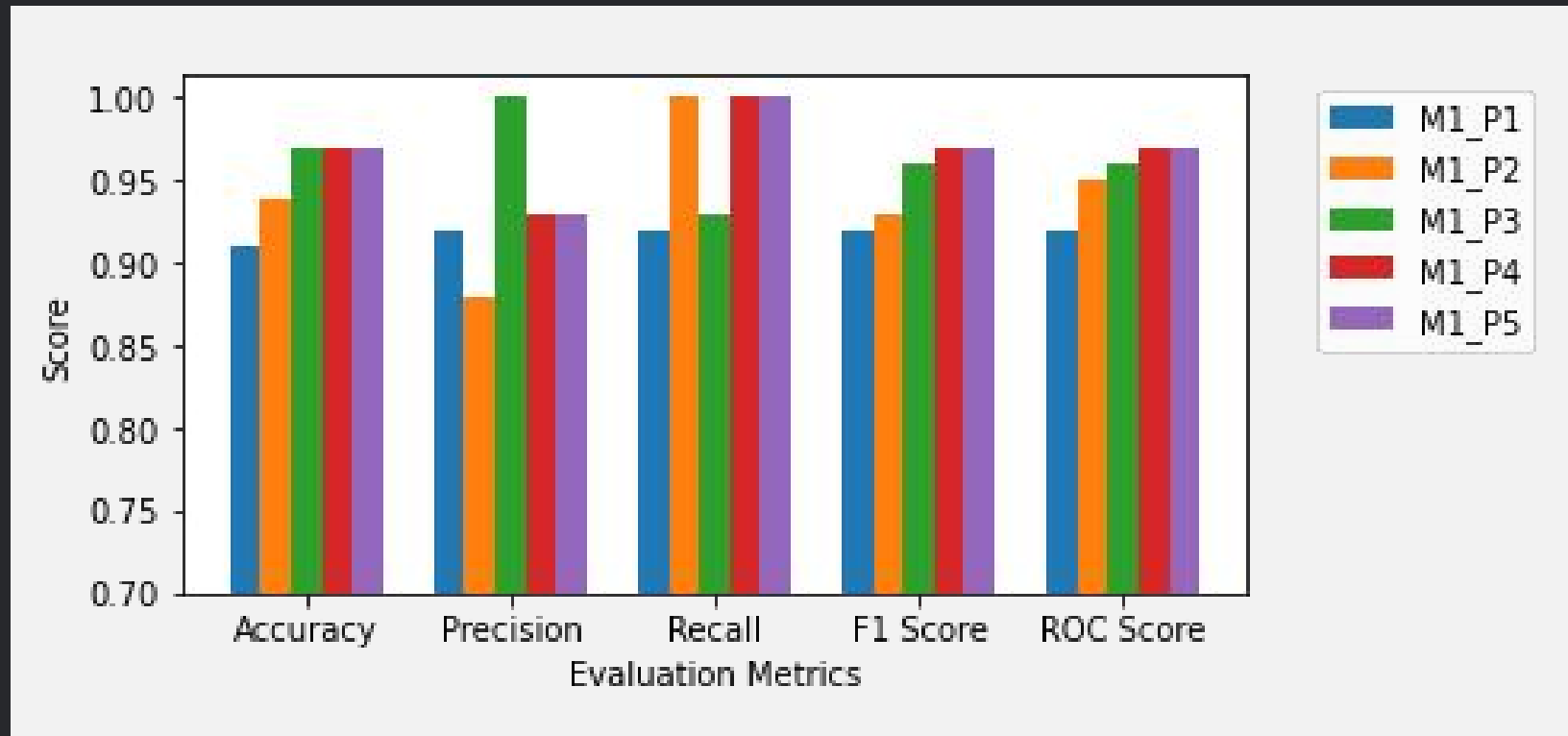
Machine Learning Model	Shorthand Notation
RocketClassifier	M1
HiveCoteV2	M2

Model Evaluation - RocketClassifier

Performance Comparison

Evaluation Matrix	M1-P1	M1-P2	M1-P3	M1-P4	M1-P5
Training Time (Wall time)	11.3 s	25.8 s	24.9 s	24.5 s	24.4 s
Prediction Time (Wall time)	2.97 s	6.49 s	6.28 s	6.18 s	6.1 s
Accuracy Score	0.91	0.94	0.97	0.97	0.97
Precision Score	0.92	0.88	1.00	0.93	0.93
Recall Score	0.92	1.00	0.93	1.00	1.00
F1 Score	0.92	0.93	0.96	0.97	0.97
ROC AUC Score	0.92	0.95	0.96	0.97	0.97

Model Evaluation - RocketClassifier

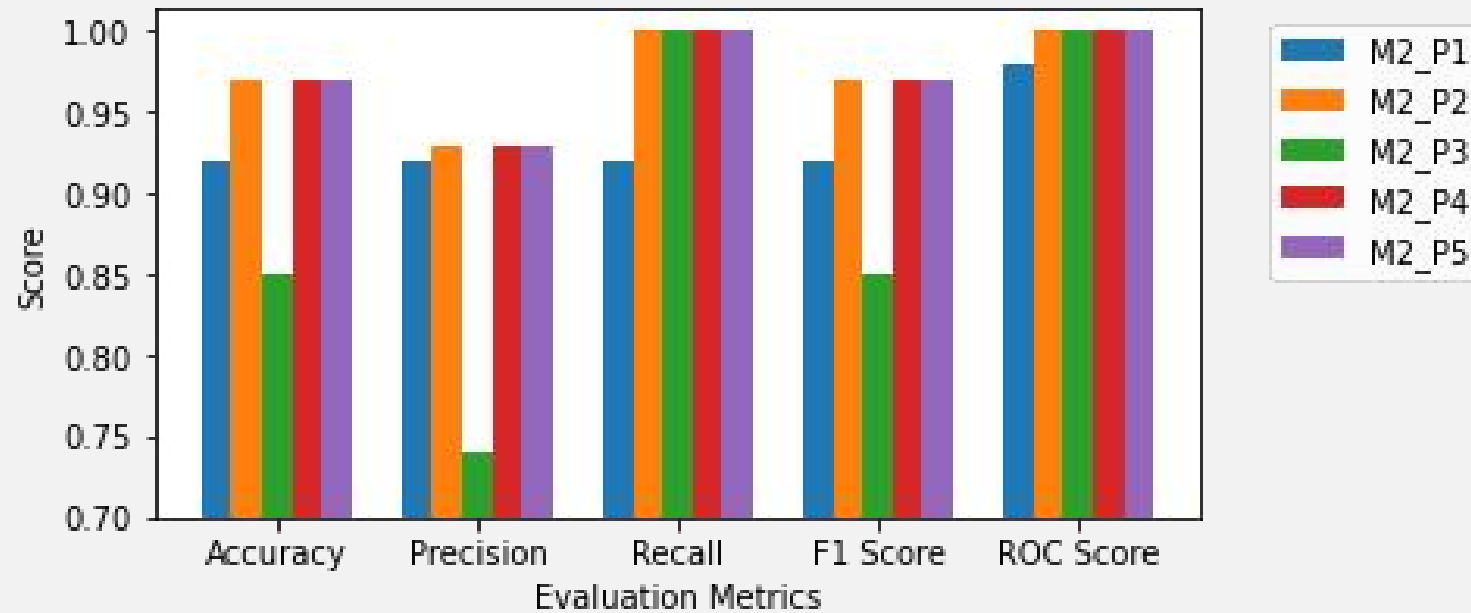


Model Evaluation - HiveCoteV2

Performance Comparison

Evaluation Matrix	M2-P1	M2-P2	M2-P3	M2-P4	M2-P5
Training Time (Wall time)	1min 2s	1min 32s	1min 26s	1min 31s	1min 25s
Prediction Time (Wall time)	9.87 s	14 s	14.4 s	13.8 s	12.7 s
Accuracy Score	0.92	0.97	0.85	0.97	0.97
Precision Score	0.92	0.93	0.74	0.93	0.93
Recall Score	0.92	1.00	1.00	1.00	1.00
F1 Score	0.92	0.97	0.85	0.97	0.97
ROC AUC Score	0.98	1.00	1.00	1.00	1.00

Model Evaluation - HiveCoteV2





Observation

- RocketClassifier requires less training time and predicting time in general comparing to HiveCoteV2.
- Model's performance is highly related to preprocessing methods.
- Truncate Data to Fixed Length
 - Has the poorest performance among the 5 methods.
- Sample Padding with Regression Prediction and Uniform Scaling
 - Have the best performance among the 5 methods.
- Suffix Padding with Mean
 - Has a satisfying performance with both of the two model.
- Suffix and Prefix Padding with Mean
 - Has good performance with RocketClassifier
 - Has poor performance with HiveCoteV2.

TSC Approach 3: Time Series Classification Algorithms on Varying Length Time Series

- KNN & DTW

	$\hat{Y} = 1$	$\hat{Y} = 0$
$Y = 1$	75.86%	24.14%
$Y = 0$	0	100%

Accuracy: 79.41%

TSC Approach 3: Time Series Classification Algorithms on Varying Length Time Series - KNN & DTW

L1 Normalization:

$$X_i = \frac{X_i}{\sum |X_i|}.$$

Accuracy: 94.12%

L2 Normalization:

$$X_i = \frac{X_i}{\sqrt{\sum X_i^2}}.$$

Accuracy: 91.18%



Part 5 Conclusions



Three Approaches to solve Multivariate Variable-length Time Series Classification (TSC)

- Approach 1: Reform Problem to Non Time Series Classification - SVM
- Approach 2: Time Series Classification Algorithms on Fixed Length Time Series - RocketClassifier & HiveCoteV2
- Approach 3: Time Series Classification Algorithms on Varying Length Time Series - KNN & DTW



Future Works

- Enlarging and adding further diversity to the dataset.
- Hyper-parameter tuning to improve performance.
- More data preprocessing methods and time series classification algorithms can be taken into the study.
- Study why certain data preprocessing methods performs poorly with some models but performs satisfyingly with some other models

References

Ã–zdemir, A.T., Barshan, B. (2014). Simulated Falls and Daily Living Activities Data Set [Data set]. UCI Machine Learning Repository.

<https://archive.ics.uci.edu/ml/datasets/Simulated+Falls+and+Daily+Living+Activities+Data+Set#>

Dempster, Angus and Petitjean, Francois and Webb, Geoffrey I, ROCKET: Exceptionally fast and accurate time series classification using random convolutional kernels, 2019, arXiv:1910.13051 <https://doi.org/10.48550/arXiv.1910.13051>

Dynamic time warping. (2022, April 11). In Wikipedia.
https://en.wikipedia.org/wiki/Dynamic_time_warping

Jeremy, Z. (2020, February 1). Dynamic time warping. Medium.
<https://towardsdatascience.com/dynamic-time-warping-3933f25fcdd>

Keogh, E. (2003). Efficiently Finding Arbitrarily Scaled Patterns in Massive Time Series Databases. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds) Knowledge Discovery in Databases: PKDD 2003. PKDD 2003. Lecture Notes in Computer Science(), vol 2838. Springer, Berlin, Heidelberg.
https://doi.org/10.1007/978-3-540-39804-2_24



References

Middlehurst, Matthew, James Large, Michael Flynn, Jason Lines, Aaron Bostrom, and Anthony Bagnall. "HIVE-COTE 2.0: a new meta ensemble for time series classification." Machine Learning (2021). <https://doi.org/10.48550/arXiv.2104.07551>

Shokoohi-Yekta, M., Hu, B., Jin, H., Wang, J., & Keogh, E. (2017). Generalizing DTW to the multi-dimensional case requires an adaptive approach. *Data mining and knowledge discovery*, *31*(1), 1–31. <https://doi.org/10.1007/s10618-016-0455-0>

Tan, Chang Wei & Petitjean, François & Keogh, Eamonn & Webb, Geoffrey. (2019). Time series classification for varying length series. <https://doi.org/10.48550/arXiv.1910.04341>



Thanks