Final Report: Time Series Assignment

The approach used to implement PAA on this data set:

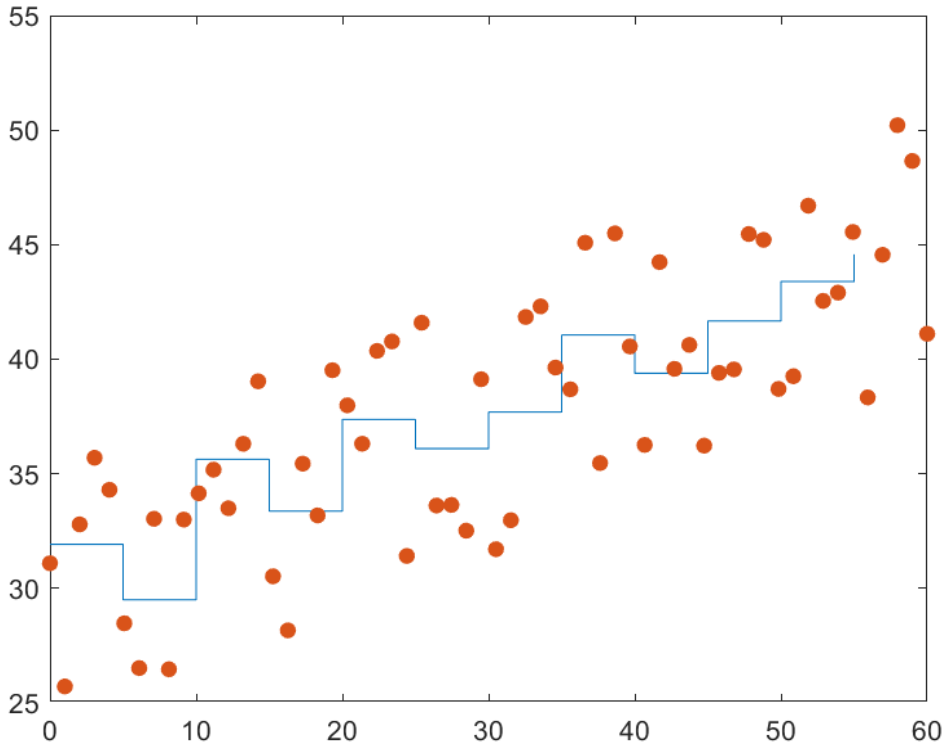
PAA is the Piecewise Aggregate Approximation of the data. In other words, it takes segments of size N of the data and uses the average of those N values to reduce the amount of data, while accurately representing it.

To implement the PAA, I take in the data as a matrix and the int c as inputs. Data is the data set, where each row represents a time series. The int c is the size of the desired segments.

I start off by initializing a matrix of size [600, 60/c] to store the PAA. Since each segment is of size c, there will be 60/c segments. Then I iterate through all the times series in data. For each time series, I get the average of every segment of size c and store that value into the PAA.

The plot of a time series and the PAA representation:

Here is an example of a time series that is classified as increasing to show the PAA represents the data accurately.



The approach to creating the SAX implementation (including the alphabet used):

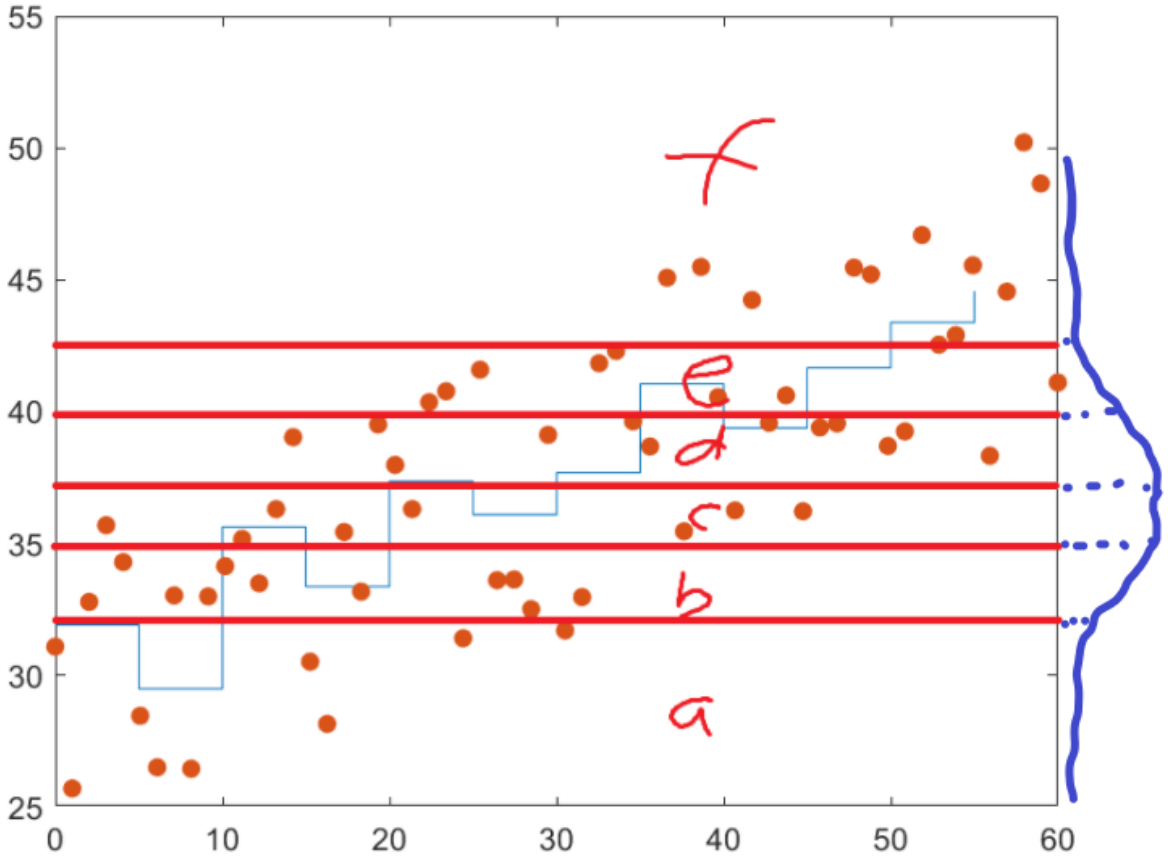
SAX is the Symbolic Aggregate Approximation of the data. In other words, it is the PAA of the data, but then each value of the PAA is given a letter representing the region it falls into. The regions are determined through the normal distribution of the data.

To implement the SAX, I take in the data as a matrix the int L and the int c as inputs. Data is the data set, where each row represents a time series. The int c is the size of the desired segments for the PAA. The int L is the number of letters to be used to represent the data.

First I create a PAA of the data. Then I find the standard deviation and mean of the data. I use the inverse norm of (%, mean, std) to find which region each value in the PAA falls into, then I assign a letter representing that region.

The plot of a time series and the SAX representation:

Here is an example of a time series that is classified as increasing to show the PAA represents the data accurately.



The SAX representation of the data becomes: “aacbdcdedeff”. By looking at the letters representing the data it is easy to see that this time series is increasing.

The processing process to establish your training and testing data generation:

It is generally good practice to take 10% of the data and put it aside for testing. My “splitData” function takes the first 90 data points of each set of data and puts it in DataTrain,

while the last 10 data points are put in DataTest. Therefore DataTrain has 540 rows of data, and DataTest has 60 rows of data. Also for the Labels, they correspond to:

1. Normal
2. Cyclic
3. Increasing trend
4. Decreasing trend
5. Upward shift
6. Downward shift

Explanation of my classification process and results:

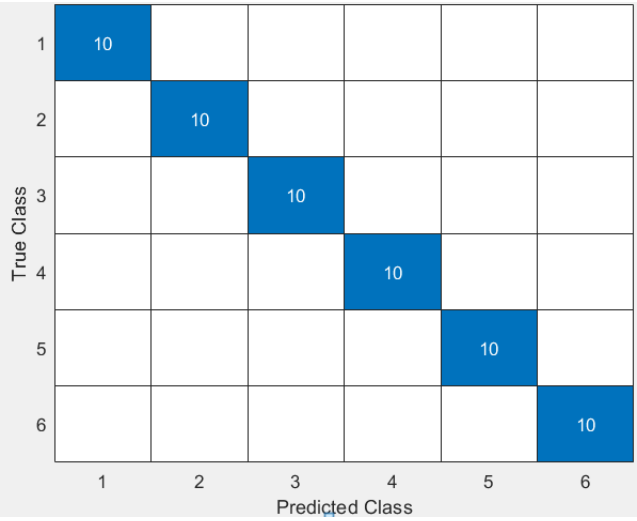
There are 4 different classification scenarios I had to consider.

1. The normal data set trained using euclidian distance.
2. The normal data set trained using manhattan distance.
3. The PAA of the data trained using euclidian distance.
4. The PAA of the data trained using manhattan distance.

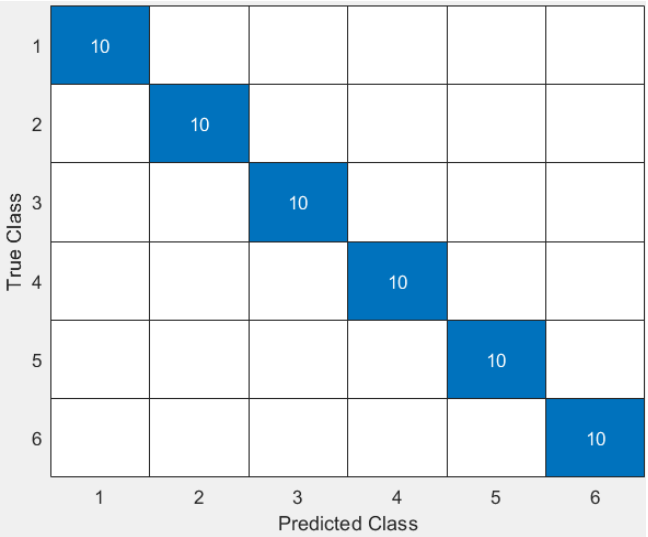
For each classification, the method to train the data is almost identical. For the data set in question (original data or PAA of it), and the choice of distance, I train the data using:
“mdl=fitcknn(dataTrain, trainLabel, 'distance', @(x,y)dist_func(x,y), 'NumNeighbors', 1);”
This trains the data through an “expert” approach. mdl makes a prediction using the nearest neighbor via the current distance function and chooses the class label of that neighbor as the prediction for the current test time series.

Here are the results of the classification process. A confusion matrix will be displayed for each.

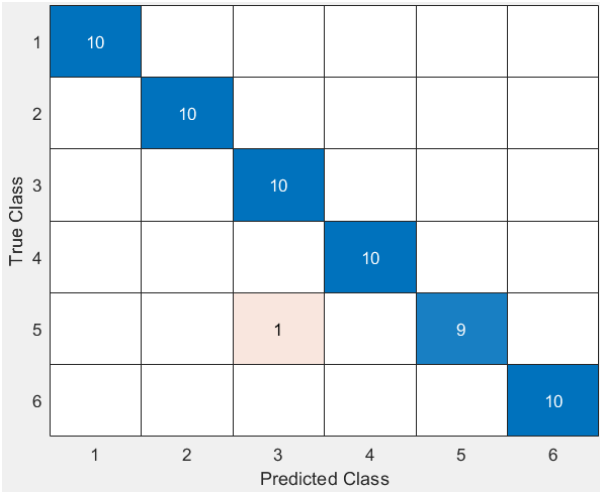
1. The normal data set trained using euclidian distance.



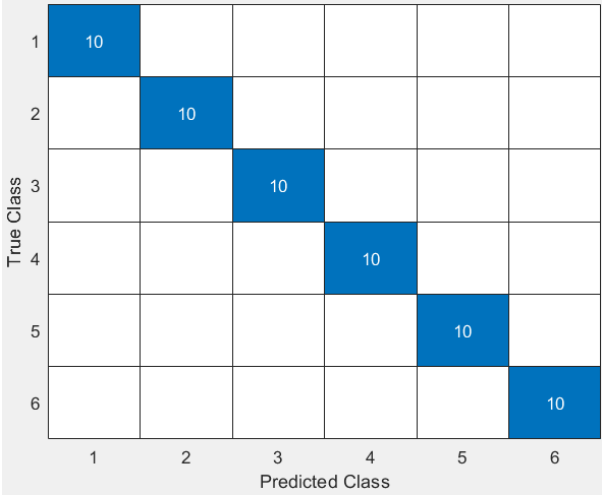
2. The normal data set trained using manhattan distance.



3. The PAA of the data trained using euclidian distance.



4. The PAA of the data trained using manhattan distance.



Comparisons and conclusion of the classification:

- There is no difference between the accuracy of the model on the original data between using euclidian distance and manhattan distance. The reason being that both models have 100% accuracy on the test data.
- The model for using manhattan distance on the PAA still have 100% accuracy, but the euclidian distance on the PAA has only a 98% accuracy. It mistook a time series that was categorized as an upward shift as an increasing time series, which is a fairly reasonable mistake. Therefore we can say that using the manhattan distance is more accurate for the PAA set.