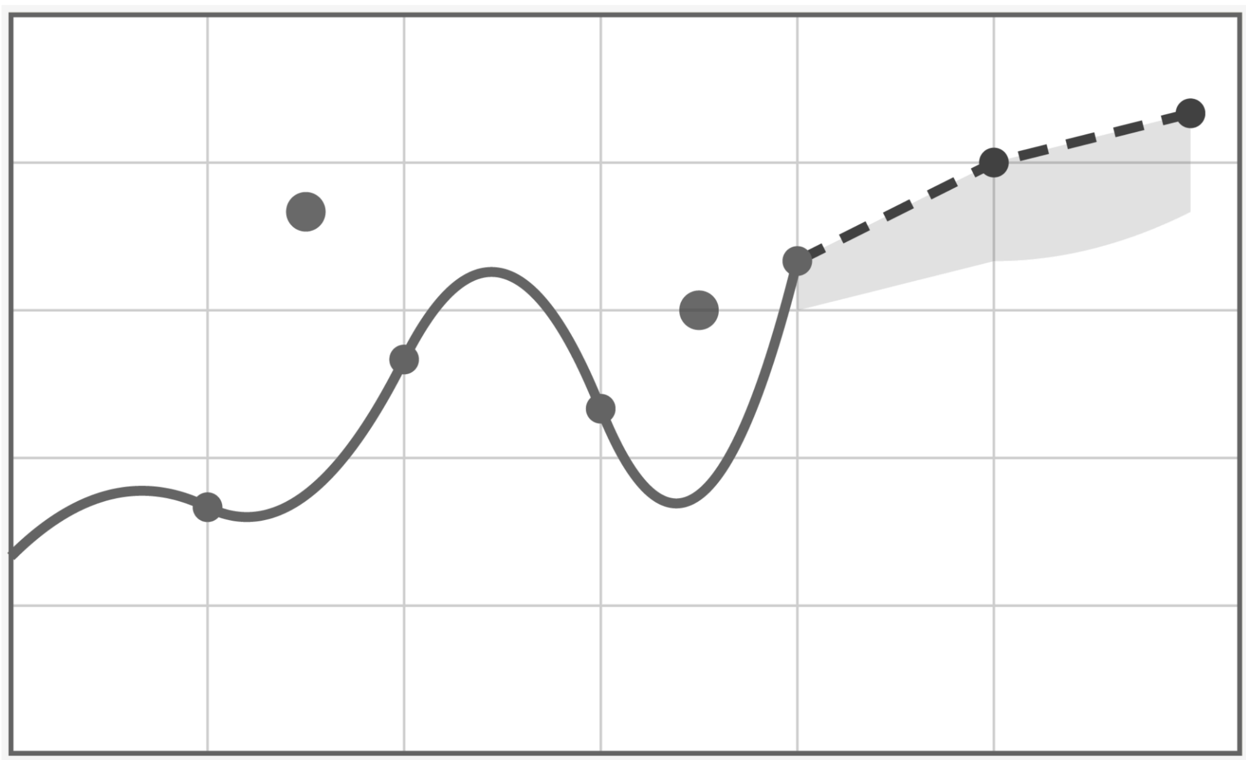
Exercises 3
(Data Preprocessing)

DATA.ML.450 Time Series Analysis using Machine Learning (Autumn 2025)



Anas Uddin

24.09.2025

MSc in Data Science

Exercise 1

How the Fourier Transform can be used for time series data manipulation in machine learning, and especially with neural networks:

The Fourier Transform decomposes a time series into its frequency components, which are sines and cosines. This is useful in machine learning because:

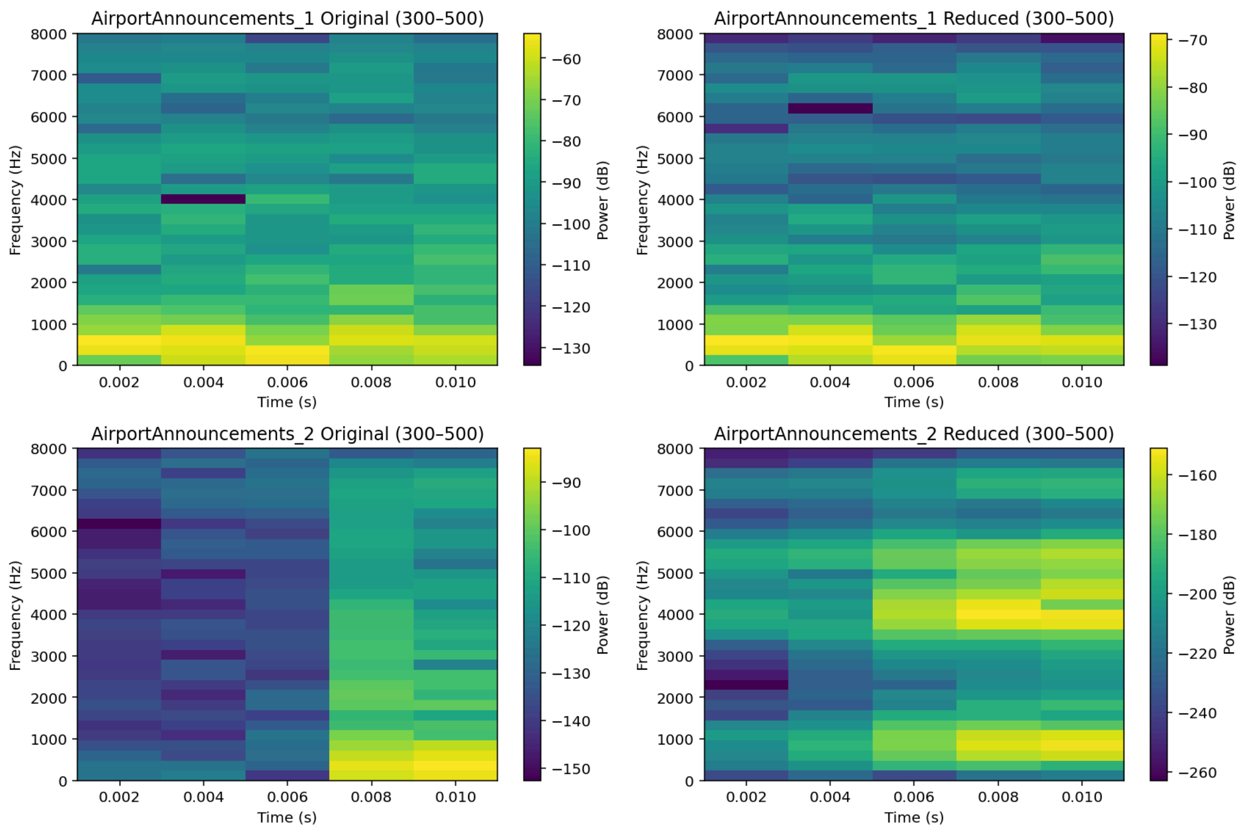
- It shows periodic patterns, for example, seasonality, cycles that may not be apparent in the raw time domain.
- It serves as a preprocessing step, converting signals into the frequency domain for easier visualisation, filtering, or feature extraction.
- Neural networks can then work on features that describe the frequency content of the data, like amplitude, phase, and dominant cycles, thereby improving prediction or classification tasks.
- In the case of non-stationary series, Fourier transforms are applied in windows (spectrograms), so that networks can learn both time and frequency-based patterns.

Why is it not worth training and computing the frequency conversion (Fourier Transform) using neural networks?

- The Fourier transform is a well-defined, exact, and efficient mathematical operation (FFT).
- FFT computes frequency decomposition in O(N log N) time, so it is much faster and more accurate than training a neural network.
- Neural networks would be slow, data-hungry, and imprecise for something that has a precise analytical solution.
- It's more useful to precompute frequency features with FFT and let the neural network focus on higher-level tasks such as pattern recognition, classification, or forecasting.

Exercise 2

I applied noise reduction using the noisereduce library on two speech samples, `AirportAnnouncements_1.wav` and `AirportAnnouncements_2.wav`, found in the repository. From each of these speech samples, a short segment (samples 300-500) was extracted, and spectrograms were generated for both the original and noise-reduced signals. We can see the results show that noise reduction lowered background energy levels and smoothed frequency components, specifically in higher bands; therefore, it makes the speech signals cleaner and more distinct. It clearly shows the effectiveness of the method.

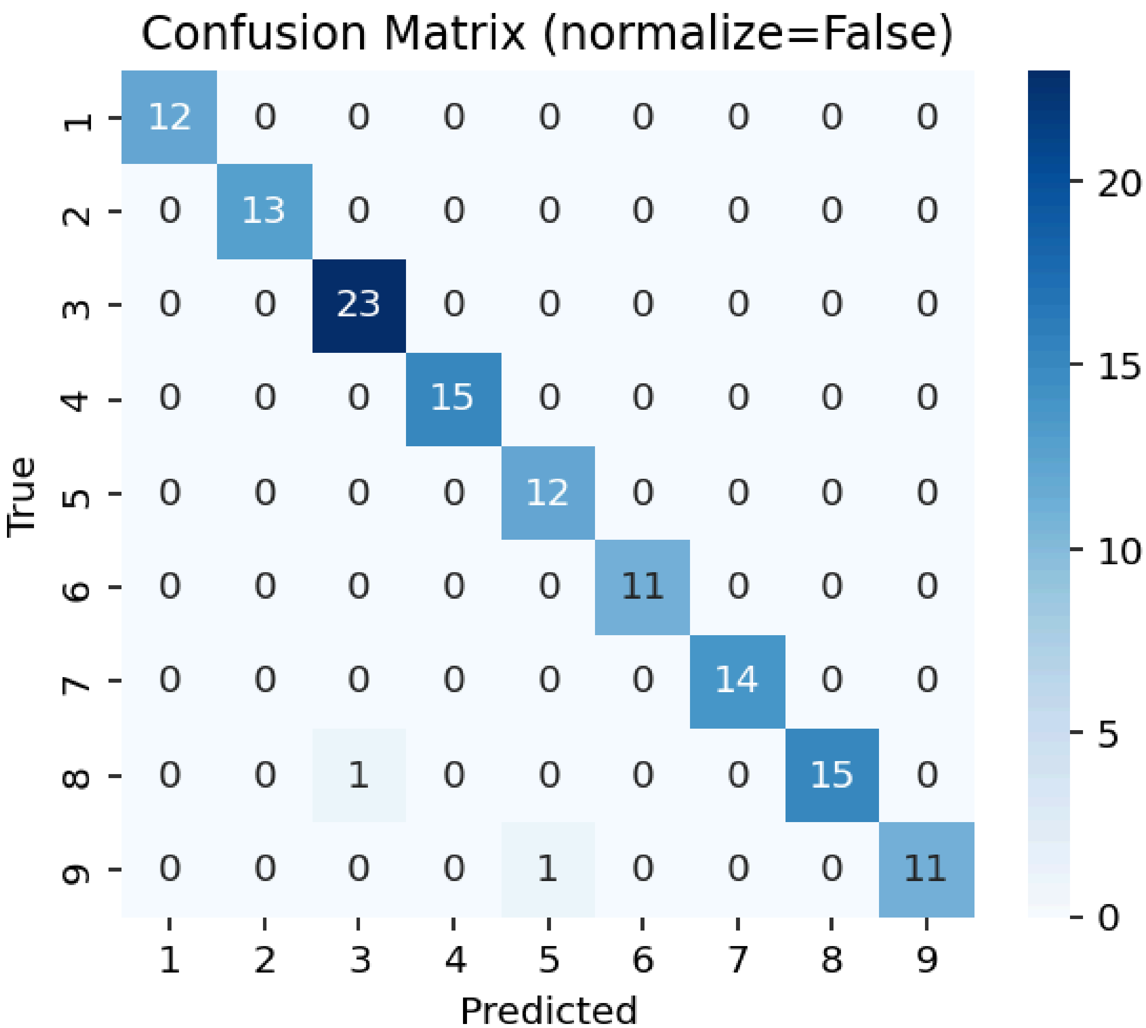


Exercise 3

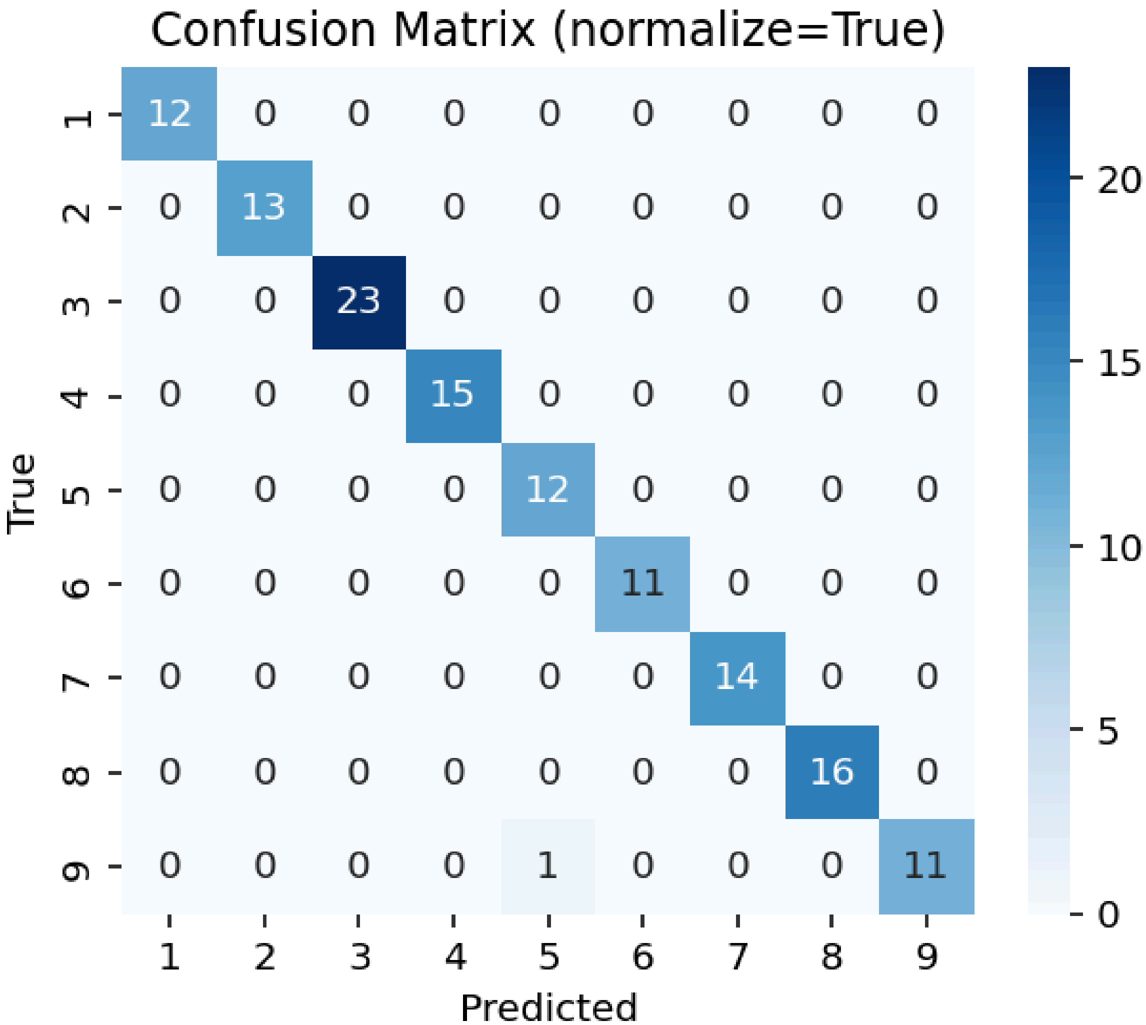
I trained the neural network classifier on the `JapaneseVowels` aeon dataset with and without feature normalization.

- Accuracy without normalization: 98.4375%
- Accuracy with normalization: 99.21875%

Without normalization, the confusion matrix shows a strong diagonal structure, but with a few off-diagonal entries, which indicates minor misclassifications.



But with normalization, we can see that the confusion matrix is perfectly diagonal. It confirms that all samples were classified correctly after normalization.



Normalization improved the performance of the model by removing scale bias across features. This `JapaneseVowels` dataset included recordings with different raw amplitudes, so rescaling the input space allowed the classifier to focus on temporal patterns more than absolute magnitudes. Therefore, there were fewer misclassifications and higher separability of classes after normalization.

Why is the X usually normalized but not the y in prediction?

X (features) is normalized to ensure that all input variables are on a comparable scale. It improves optimization stability and convergence. However, y (labels) represent discrete class indices whose meaning would be lost if normalized. Thus, they remain unscaled.