Machine learning for predictive maintenance

Alessandro Gaudenzi

Abstract

This relation discusses the use of machine learning techniques for predictive maintenance. The study begins by introducing the tools used in the research and the dataset used. In the first part there are the analysis of the dataset to identify patterns and correlations that indicate equipment failure. The second part presents two forecasting method that uses machine learning model to predict sensors state and consequentially indicators of component failures.

1 Introduction

Predictive maintenance is a proactive maintenance strategy that relies on data analytics and machine learning algorithms to predict equipment failures before they occur. This approach aims to maximize equipment uptime and reduce maintenance costs by identifying potential issues early on and scheduling maintenance tasks accordingly.

In order to do so there is a need to examine and understand the data provided by the machine's sensors (and first of all collect and categorize them so that they are relevant) and then develop machine learning models to forecast the possible state of sensors that indicates an imminent failure.

2 Tool and dataset used

For the experimentation I used an example dataset from Microsoft Azure and two different neural networks models for forecasting.

2.1 Long short-term memory

A Long Short-Term Memory (LSTM) neural network is a type of recurrent neural network (RNN) architecture that is designed to address the vanishing gradient problem that can arise in traditional RNNs.

At its core, the LSTM network is composed of memory cells, input gates, output gates, and forget gates. The memory cells are responsible for storing information over long periods of time, while the input gates control the flow of new information into the memory cells. The output gates determine which information from the memory cells is outputted by the network, and the forget gates determine which information is discarded.

The LSTM network is trained using backpropagation through time, which involves recursively applying the chain rule of calculus to compute gradients of the loss function with respect to the network parameters over a sequence of inputs. The use of the forget gates in the LSTM architecture allows gradients to be propagated over longer time intervals, which can help to address the vanishing gradient problem.

One of the key benefits of the LSTM network is its ability to capture long-term dependencies in sequential data, such as language or speech. This has led to its widespread adoption in a variety of applications, including natural language processing, speech recognition, and time series analysis. Overall, the LSTM network has proven to be a powerful tool for modeling complex sequential data, and its continued development and refinement will likely lead to further advances in a wide range of fields.

2.2 Convolutional neural networks

A Convolutional Neural Network (CNN) is a type of deep neural network architecture that is commonly used in image and video processing applications. The CNN was first introduced in the 1980s, but it wasn't until the advent of large datasets and powerful computing hardware that the CNN became widely used.

At its core, the CNN is composed of one or more convolutional layers, followed by one or more pooling layers, and then one or more fully connected layers. The convolutional layers are responsible for extracting features from the input data, while the pooling layers downsample the feature maps to reduce computational complexity. The fully connected layers perform classification or regression on the extracted features.

During training, the CNN is optimized using backpropagation, which involves computing gradients of the loss function with respect to the network parameters and updating the parameters using gradient descent or a related optimization algorithm.

One of the key benefits of the CNN is its ability to automatically learn spatial hierarchies of features from raw input data. This makes the CNN well-suited for image classification, object detection, and other computer vision tasks, where spatial relationships among pixels or image regions are important.

2.3 Microsoft Azure Predictive Maintenance dataset

The dataset used for the experimentation is the dataset that was made available from Microsoft in the context of "Azure AI Notebooks for Predictive Maintenance".

It consists in temporal data of about 100 machines in time series format, registration of the hourly recording of 4 different sensors if the machines (voltage, rotation, pressure and vibration) and the recording of 4 components failure, error and scheduled maintenance, with the reading recorded every hour for an year (8750 measurements for every machine).

3 Data analysis

3.1 Plotting features

The first thing to do to explore the data is to try plotting the features to see if you can find any useful information.

From the boxplot of the feature values divided by component failure in 1 it can be noted that every component failure is associated with an average increment or decrement of only one sensors value.

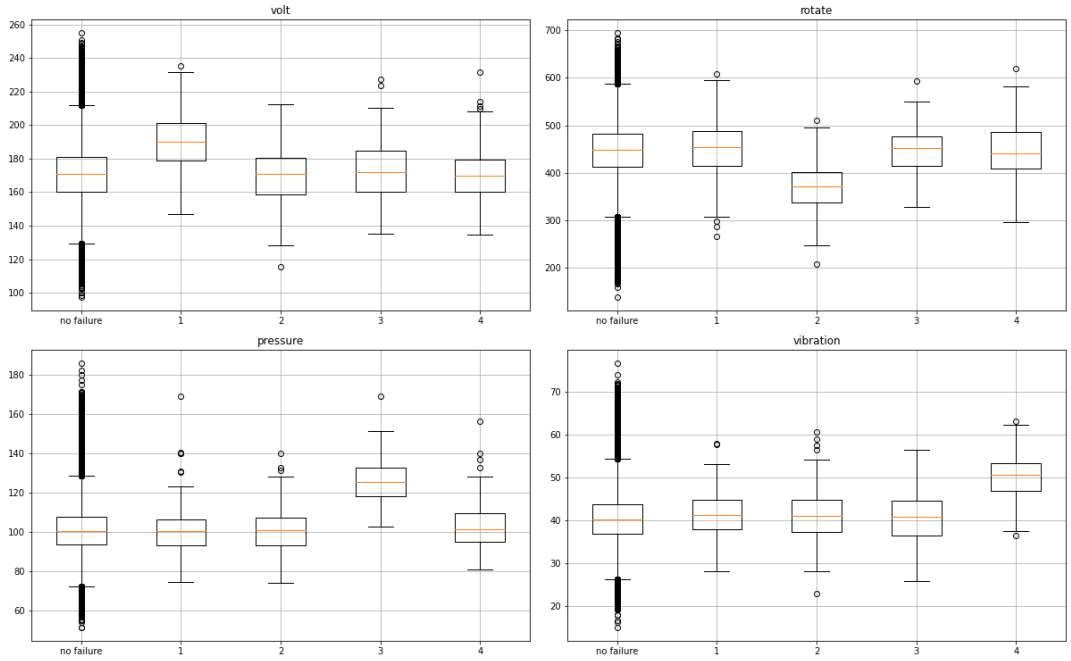

Figure 1: Value of sensors divided by component failures

In order to confirm that the sudden increment or decrement of the sensor's reading is really associated with a failure it can be plotted the sensor value graph of two machine that had multiple failure of a component. In figure 2 it can be seen the plot of the vibration sensor of machine 1, in figure 3 the plot of the rotation sensors of machine 10. It can be seen a spike in the sensor's reading in the proximity of a failure (marked with a red dot) but also other spikes where no failure occour, probably due to an hidden cross-dependence of other feature values in the failure of a component.

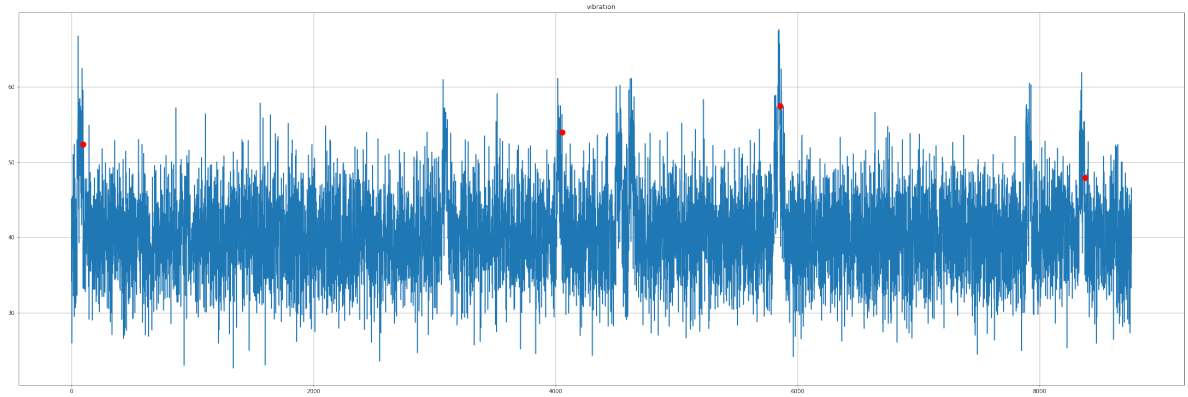

Figure 2: Graph of the vibration sensor reading of the machine 1 with failures marked red

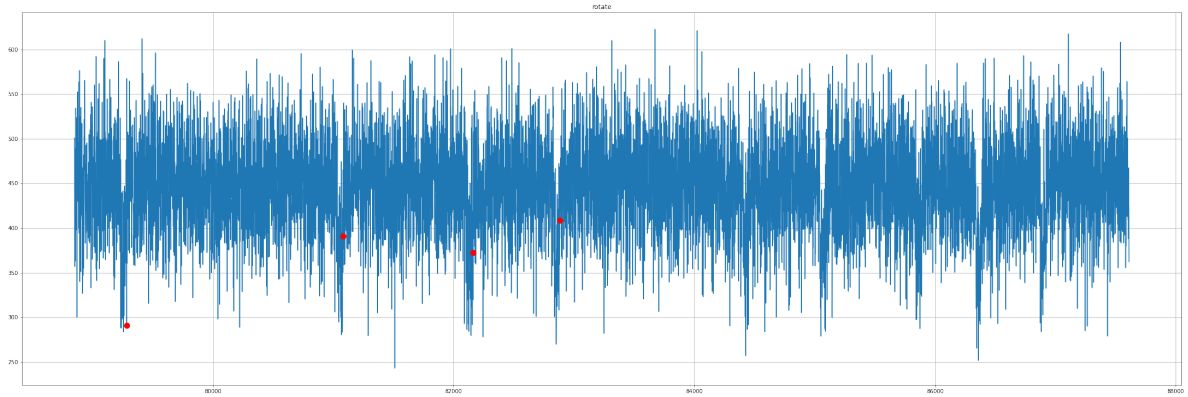

Figure 3: Graph of the rotation sensor reading of the machine 10 with failures marked red

3.2 Moving average

In order to spot better the spikes in reading the moving average of sensor's reading can be used instead of the punctual ones, but a good window for the moving average it has to be individuate first.

To estimate a good window length of the moving average the autocorrelation and partial autocorrelation of the feature can be plotted, so that is provided an indication of after how many step of the timeframe the future reading of the sensors are no more correlate to the previous one. In figure 4 there is the plot of autocorrelation, while in figure 5 there is the plot of the partial one: can be seen that the autocorrelation tend to become irrelevant after the 40 hours mark and the partial autocorrelation after 20

hours, so it was chosen a windows of 24 hours to be pair to a single day.

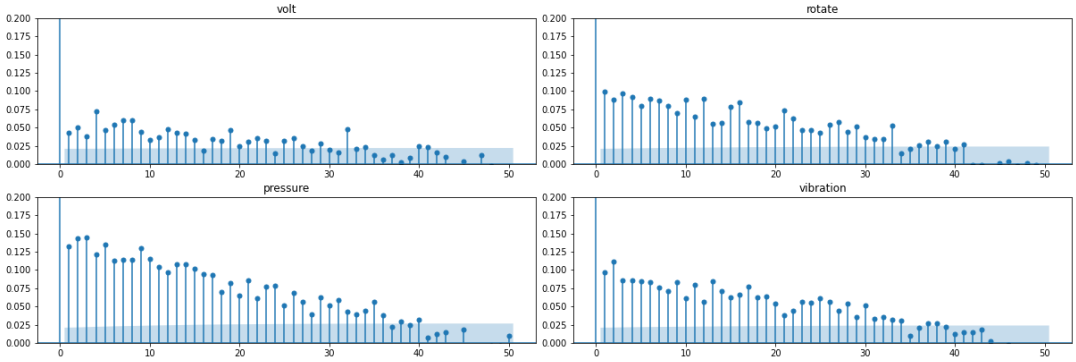

Figure 4: Autocorrelation of the feature correspondent to the sensor's readings

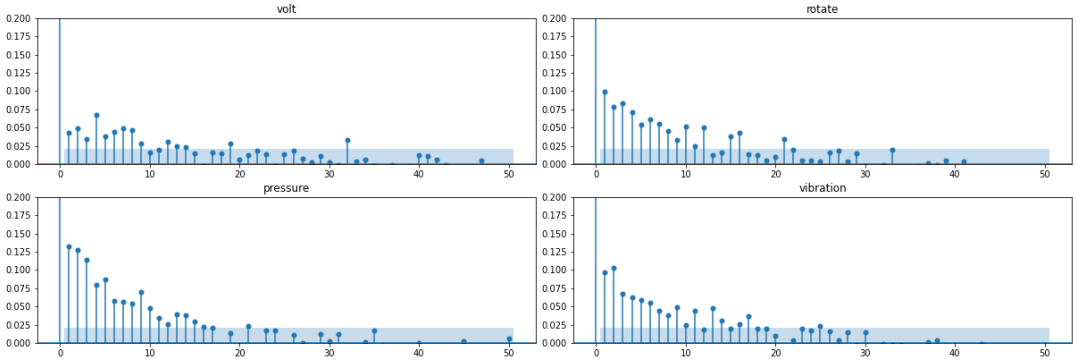

Figure 5: Partial autocorrelation of the feature correspondent to the sensor's readings

After converting the features value to their moving average with a 24 hours moving window, it can be seen in figures 6 and 7 that show the graph for the same machines and sensors of figures 2 and 3. The spikes in proximity of the failures are highlighted and can be clearly seen, but it highlight other spikes too, where a failure don't happen. This fact has to be considered when evaluating the results of the forecasting experiment, because according to maintenance cost and stop cost can be profitable or not to do the predictive maintenance for every spike predicted since the rate of false positive can be relevant.

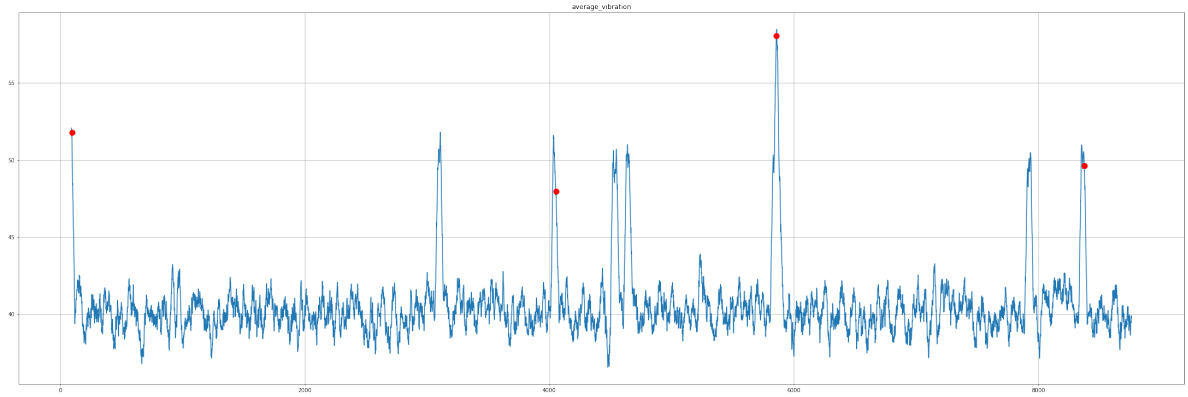

Figure 6: Graph of the vibration sensor reading of the machine 1 with failures marked removing average with 24 hours window

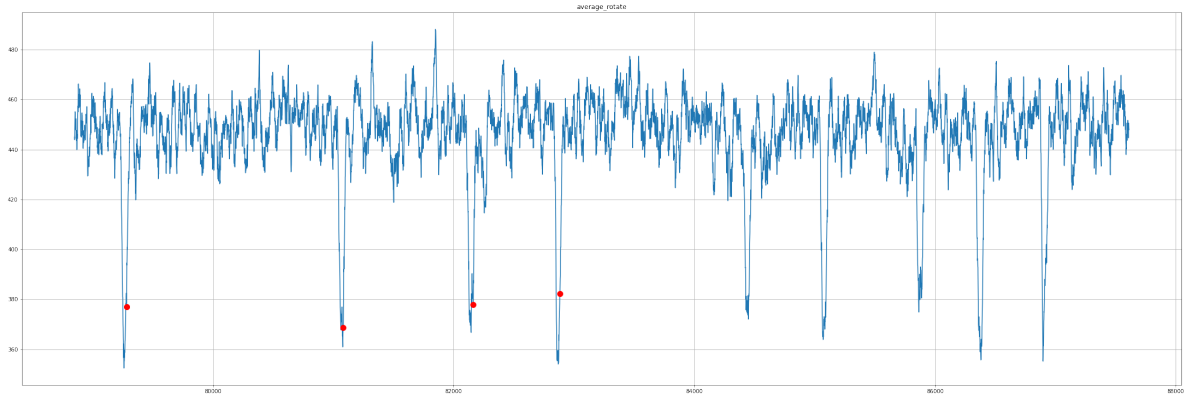

Figure 7: Graph of the rotation sensor reading of the machine 10 with failures marked red, moving average with 24 hours window

4 Forecasting

After having analyzed the correlation between sensors reading and component failures there are forecasting models that can be used in order to try to anticipate the condition that more probably lead to a failure. They have been trained two different type of forecasting models, based on LSTM and CNN explained in the second chapter.

The values of the four features have been normalized for the purpose of prevent the model from preferring one over another.

To test the model the dataset have been splitted in 80% training (7000 instances, equal to 7000 hours) and 20% test (1750 instances/hours). The models uses all features at once for the training and return a model that predict the result of a temporal moment based on the n precedent value. It was chosen $n = 24$ for the same reason that it was chosen for the moving average plotting.

4.1 LSTM

In the figure 8 there is the scheme of the recurrent neural network of type LSTM that present a fairly standard approach of initial expansion in node number and then a contraction.

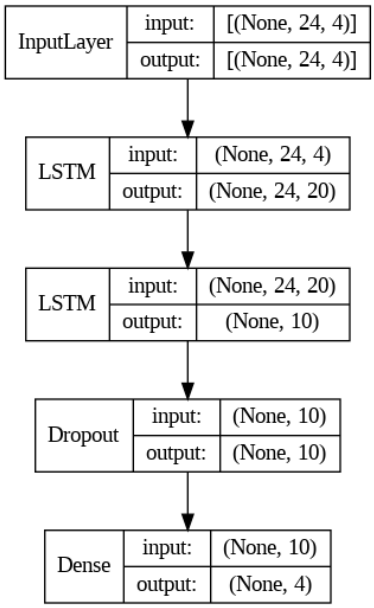

Figure 8: Scheme of the LSTM model utilized

The model's forecasting result on the testing part of the dataset can be seen in figure 9. With the data normalized it has reached a mean squared error of 0.01438, used as a score. The two spikes presents in the original data can be seen clearly, and it becomes even more clearly with the moving average plotted in figure 10.

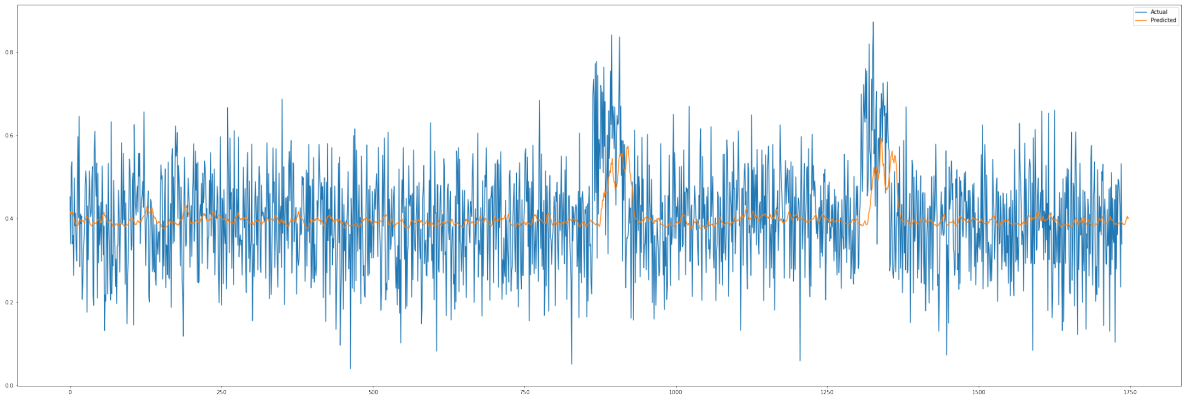

Figure 9: Graph of the forecasting done for the testing instances with the LSTM model (orange) overlying the true graph (blue)

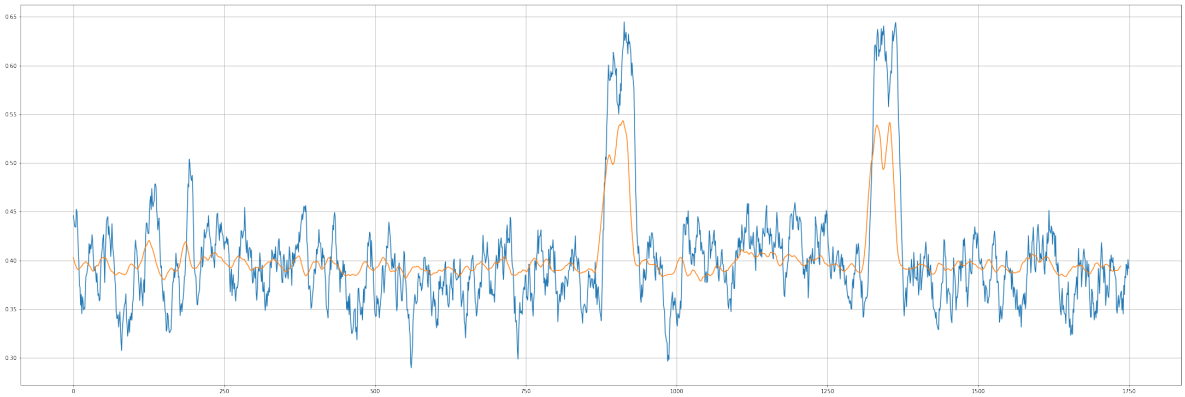

Figure 10: Graph of the moving average of the forecasting done for the testing instances with the LSTM model (orange) overlying the true graph (blue)

4.2 CNN

In the figure 8 there is the scheme of the concurrent neural network that present a fairly standard approach of initial expansion in node number and then a contraction.

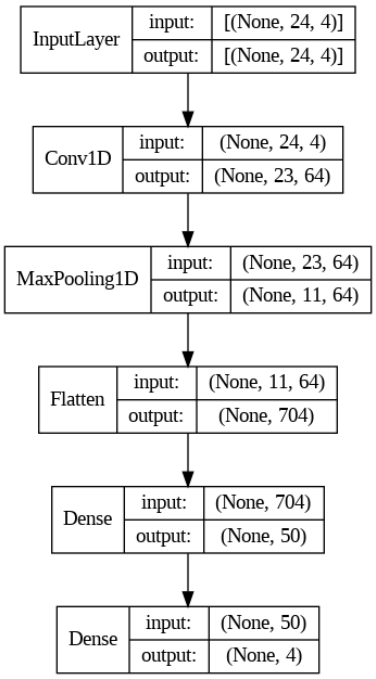

Figure 11: Scheme of the CNN model utilized

The model's forecasting result on the testing part of the dataset can be seen in figure 12. With the data normalized it has reached a mean squared error of 0.01478, a similar value to the previous LSTM model. The two spikes presents in the original data can be seen clearly, and it becomes even more clearly with the moving average plotted in figure 13.

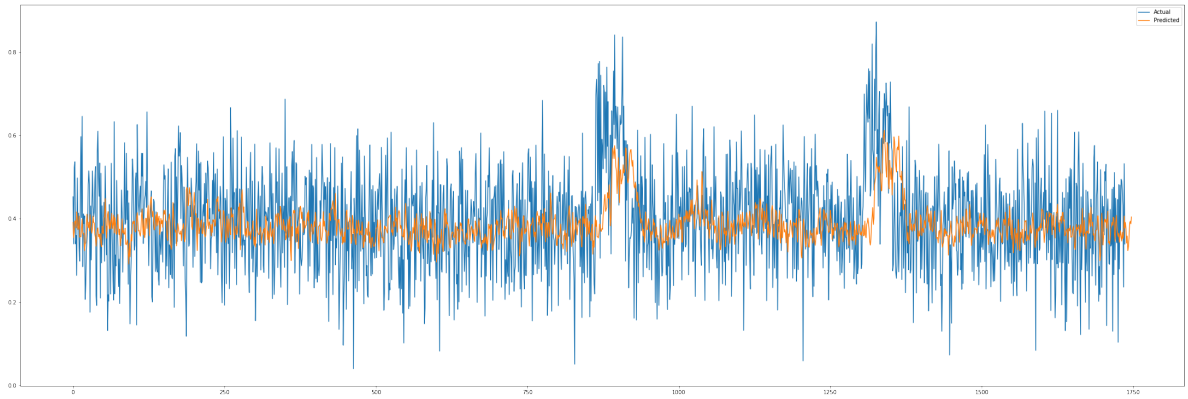

Figure 12: Graph of the forecasting done for the testing instances with the CNN model (orange) overlying the true graph (blue)

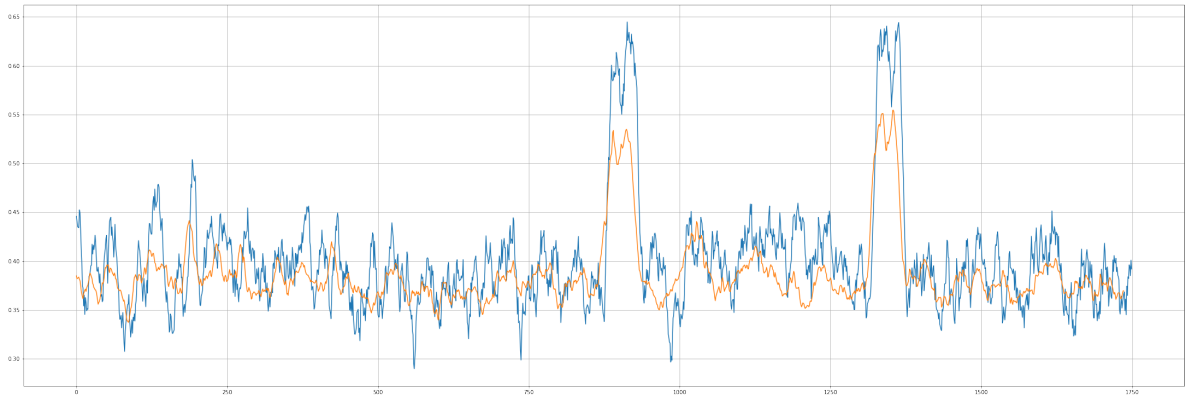

Figure 13: Graph of the moving average of the forecasting done for the testing instances with the CNN model (orange) overlying the true graph (blue)

5 Conclusion

The data examination and the results of the model's test show that there is a way to forecast a sudden spike in sensors reading that can be tracked down to a failure with at least a day in advance.

In an attempt to translate those results on another business case some things are to be considered: there may be a significant variance on the needs of a particular machine or robot, for example the forecast window may have to be longer for the needs of the production cycle, but also an economic factor have to be considered, it can be disadvantageous to perform predictive maintenance every time a possible failure is forecasted, even because the probability that the failure happen is not 1, even if the spike in the sensors readings happen as forecasted. For this reason these conclusions cannot be drawn for each production cycle but the same process can be followed by adapting it.

Usage report

The project is developed with those tool and languages:
Google Colab: `www.colab.research.google.com/`
Python: `www.python.org/`
Numpy: `www.numpy.org/`
Keras: `www.keras.io/`
Tensorflow: `www.tensorflow.org/`
Sklearn: `www.scikit-learn.org`

Repository of the project:
`www.github.com/ale-gaudenzi/predictive-maintenance`

Repository of the dataset:
`www.kaggle.com/datasets/arnabbiswas1/microsoft-azure-predictive-maintenance`

References

[1] R. C. Staudemeyer, E. R. Morris, "Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks"

[2] S. Hochreiter, J. Schmidhuber, "Long short-term memory"

[3] K. O'Shea, R. Nash, "An Introduction to Convolutional Neural Networks"