Use IoT to enable Predictive Maintenance

LUMEN Data Science 2020.

Project Documentation

Contents

1 Business Objectives

Failures of small parts of the machine can lead to expensive breakdowns of the entire machine. For that reason, machine maintenance is very important task in every machine dependant industry. Many companies make the mistake of determining a fixed interval in which they perform their machine maintenance. That leads to two issues:

- If the interval is set to too short, we might replace the parts that work properly.
- If the interval is set to too long, parts may start to malfunction before time for maintenance is reached.

The second issue of fixed time maintenance is that different parts of the machine have different lifetimes, therefore this approach is not usually the best one.

The modern approach to machine maintenance is called predictive maintenance. Our task, as data scientists, is to determine when the machine will malfunction based on historical data. Therefore, we can conduct the maintenance on only the part that is about to fail and just before it fails, thus greatly reducing the cost.

1.1 Business goals

The goal of this predictive maintenance project is to analyze the data from six different sensors that have been installed on seven different machines. Each sensor gathered data several times a day for extended periods of time.

By enhancing the way we conduct the maintenance of the machine through inspection of the gathered data we expect to greatly reduce the cost that is currently spent on machine maintenance.

1.2 Assessing the Current Situation

As we can see from the data frequent maintenances have been preformed and no apparent breakdowns took place. This could be sign that those fixes have been performed just on time thus greatly reducing the cost of machine maintenance, however it could also be the case that those fixes happened too soon and company wasted valuable resources.

1.3 Data

Data was provided by *Atomic Intelligence d.o.o.*, a company with headquarters in Zagreb. Data used for implementation of the project is property of *Atomic Intelligence d.o.o.* and is available for use only during this project. Any unauthorized use of data is illegal and is punishable by Croatian laws.

1.4 Data mining goals

From the data gathered by *Atomic Intelligence d.o.o.* we want to conclude when the specific part of the machine will malfunction and at what moment in time it will do so. The success of the project will be measured by models ability to predict whether a maintenance should be held and by accuracy by which it does so. We will evaluate our data on intervals between two maintenances and expect to find that a maintenance is necessary just before the maintenance that is logged in the data.

1.5 Project plan

The project will be conducted in several parts: First step of the project is understanding the data, finding correlation or dependence between different sensors of the machine, checking whether the same sensors on different machines behave in a similar way. To conclude understanding of the data we will look for evidence that some sort of failure has occurred and which sensors seem to be the best at predicting these failures.

Second step, after gathering enough intuition about the data, is to reorganize, regroup and clean the data gathered by sensors in a way that coincides with the knowledge gained during the first step of the project.

When the data is properly organized, next step to take is feature extraction using several libraries such as *tsfresh* and *XGBoost*. After that, detailed analysis of the features is necessary to conclude which features actually give even more insight into the data and failures that occur. The final part of this step is to select the features that show the deterioration of the machine and thus have predictive qualities.

Fourth step of the project will be creation of the model based on assumptions gathered thus far. For modeling, since the maintenance log is very sparse, the plan is to try a simpler, more intuitive approach like simple linear or logistic regression to predict when maintenance should be preformed. If such approach doesn't work we will try to use more advanced methods such as gradient boosted trees and neural networks. After evaluating the desired regression models by identifying other maintenances, we will try to also build classification models and compare the two.

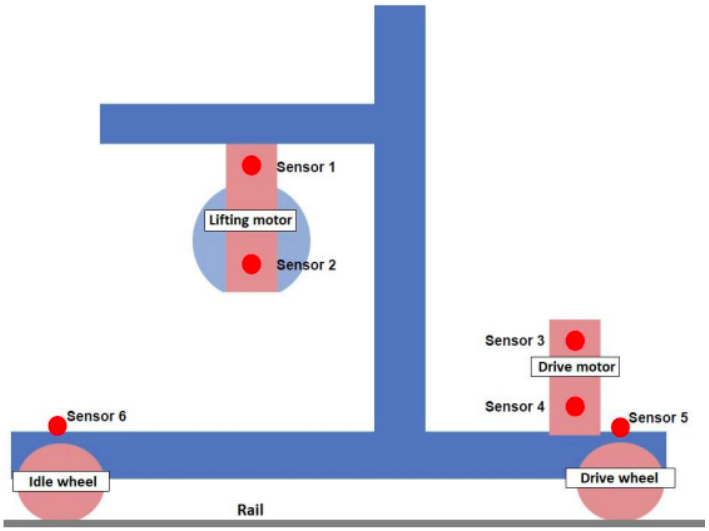
Fifth step of the project is the final evaluation of the model, we will compare our predicted maintenances to maintenances given in the data and see how they compare.

The last step is deployment of the project for that purpose we shall provide all the functioning parts of code as well as user documentation which explains in detail how to reproduce our results and how to tweak our solution so it can be implemented on other systems.

2 Data Understanding

Data gathered by *Atomic Intelligence d.o.o.* consists of signals from sensors of seven storage and retrieval systems. Each system is divided into three subsystems (drive, lifting and idle) and each system consists of one or more parts as described in following list:

- Drive system:
  - Drive wheel
  - Drive gear
  - Drive motor
- Lifting system:
  - Lifting wheel
  - Lifting gear
  - Lifting motor
- Idle system:
  - Idle wheel



For each part there are two sensors. One which measures maximum acceleration (a_{max}) and other one which measures effective velocity (V_{eff}). Furthermore, for each signal measurement timestamp was provided containing information when measurement took place. From now on following convention is to be employed: each sensor is referred to in format *Machine -Part-Sensor-Feature*.

Besides sensor signals, log containing maintenance information was given in format:

Machine	Date	Repair description

Which contains information about when and on which part of which machine maintenance was performed.

2.1 Data description report

2.1.1 Signals

First step is to load and plot given data using python libraries *pandas* and *matplotlib.pyplot* respectively. This step will give us crude insight in data.

Figure 1 shows FL01-Drive wheel-V_{eff} plotted over time. Upon closer inspection, conclusion is that points (instances of measurement) are grouped in signals (sequence of

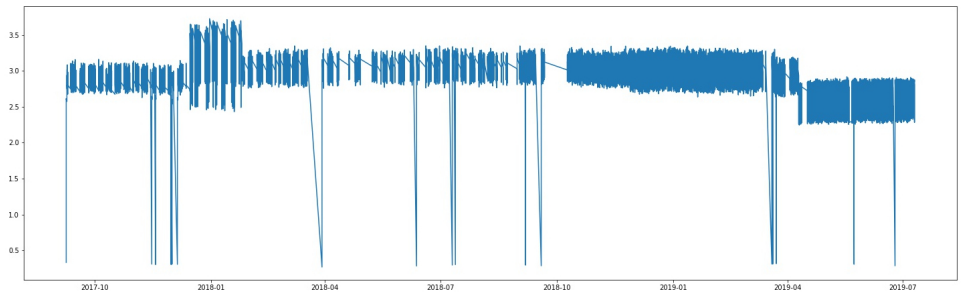


Figure 1: Drive wheel V_{eff} of FL01 machine

consecutive instances of measurements) and that is the reason for large gaps in data.

Secondly, we decided to divide the data into 2 groups, one group of lifting data, and other group of drive and idle data since they perform different tasks (one moves and other lifts).

Thirdly, the sensors usually sent a short sequence of information for about 40-60 seconds then it made a pause of about 30 seconds, and then it made another sequence of information for about 40-60 seconds.

2.1.2 Maintenance log

Alongside sensor data, maintenance log is provided in form of a table. Each row represents one maintenance on one machine. It is important to mention that only large scale maintenance are logged. As we shall see later, there is evidence in data indicating that other procedures have been preformed which have not been logged. First column is machine column. Second column is date column, it provides us with approximate interval during which procedure have been preformed. There is also a description column containing crude description of type of maintenance preformed. It is usually part replacement or grinding of the rails. Further, during each maintenance, smaller repairs have been preformed i.e. screw tightening, lubrication etc.

2.2 Data exploration

In this chapter we will observe differences between machines FL01 and FL06. Those machines were chosen in order to explore differences and similarities in behavior of a machine that required several maintenances (FL01), and machine which worked flawlessly for the entire time (FL06).

Let us begin with brief review of distribution moments. First moment is mean, it is usually denoted by $\mu = \mathbb{E}[X]$, where $\mathbb{E}[X]$ is expected value. Second moment is variance, it is measure of spread of the distribution, given by formula: $\sigma = \sqrt{\mathbb{E}[\mu - X]^2}$. Third moment is skewness, it is a measure of lopsidedness of distribution. Any symmetric

normal distribution is going to have vanishing skewness. It is calculated by formula: $\gamma = \frac{\mathbb{E}[X-\mu]^3}{\sigma^3}$. Forth moment is measure of heaviness of tails of distribution. It is called kurtosis and given by formula: $\delta = \frac{\mathbb{E}[X-\mu]^4}{\sigma^4}$.

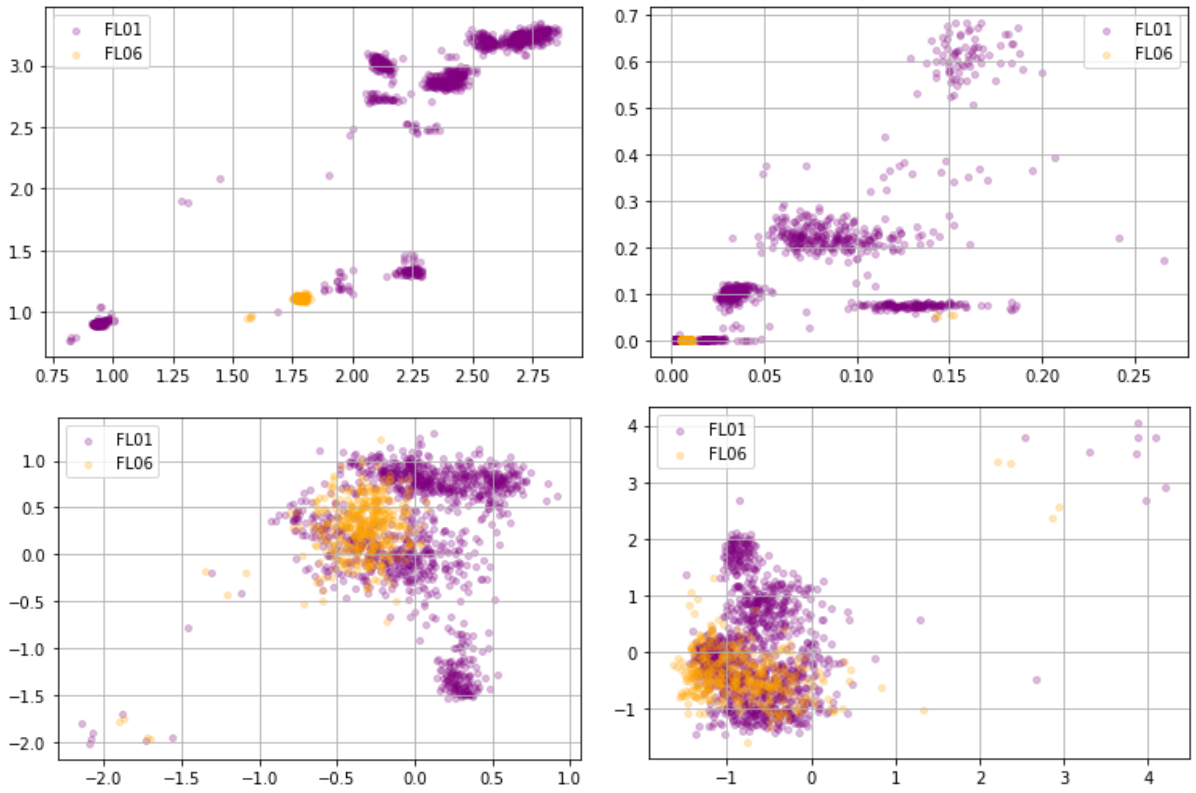


Figure 2: Comparison of dependence of moments of v_{eff} of drive-wheel to those of idle-wheel for machines FL01 and FL06. Top left is mean, top right is variance, bottom left is skewness and bottom right is kurtosis

For starters, we will examine how different parts of same machine behave with respect to each other. For that purpose scatter plot of first four moments of distributions of idle wheel V_{eff} with respect to same moments of drive wheel V_{eff} for FL01 and FL06 is shown in Figure 2.

Next, we will repeat the same thing except now we will observe V_{eff} - a_{max} plot for the same part (drive-wheel). That is shown in Figure 3.

We conclude that in both cases, the values of FL06 machine are much more grouped when compared to those of FL01. But as we increase order of moment, those differences diminish. As we can see, data is clustered. Every interval between maintenances behaves very differently from the others. There is no clear way how to compare them.

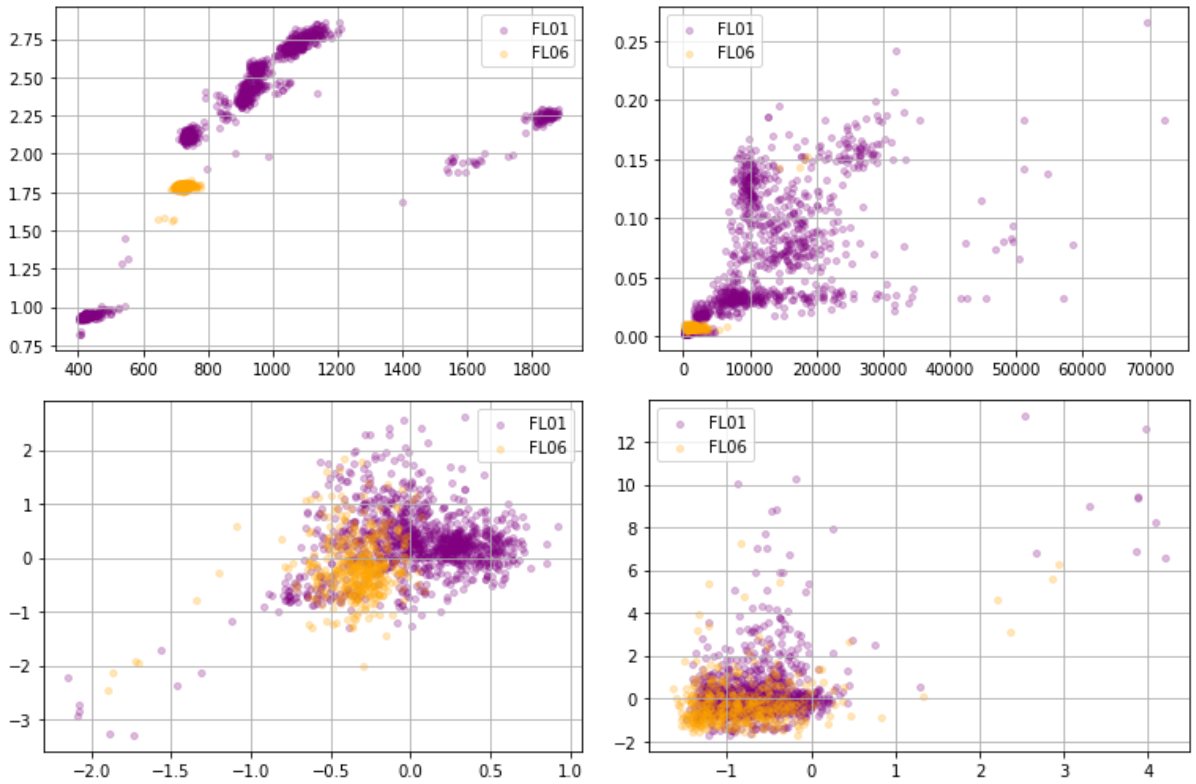


Figure 3: Comparison of dependence of moments of v_{eff} to a_{max} of drive-wheel for machines FL01 and FL06. Top left is mean, top right is variance, bottom left is skewness and bottom right is kurtosis

Next what we can observe from signals is how they were measured. They are grouped in pairs, one was measured as machines was accelerated down the rails, then there was pause and machine was accelerated backwards. This can best be seen on Figure 4. We can see a positive trend in one direction and negative in other.

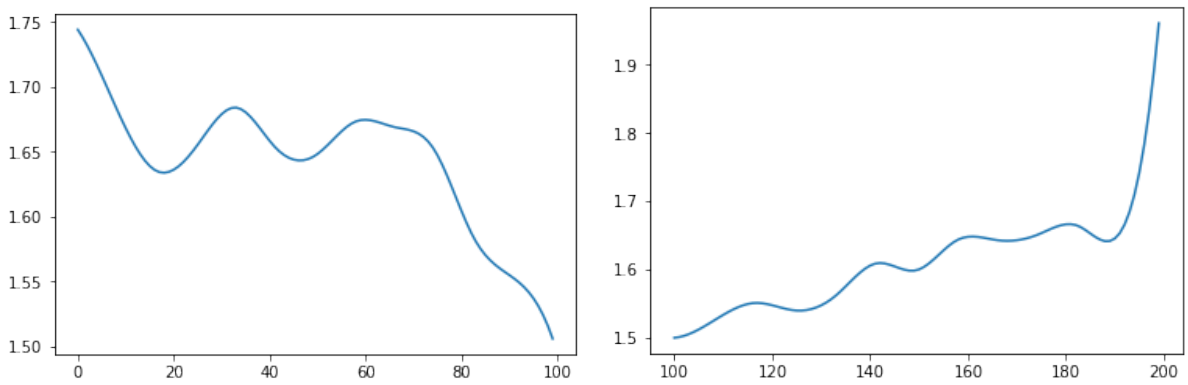


Figure 4: Plot of pair of signals of FL01-drive wheel V_{eff}

2.3 Data quality report

Several problems were noticed in the gathered data First problem we encountered was that some sensors have sent much more information then other sensors thus having signals with not equal amount of data points. One possible solution for this issue is to interpolate through our signal points and evaluate a given function in equal amount of points, each point measured at the same value.

Second problem we encountered was that not all sensors started measuring at the same time. Usually, both drive sensors and lift sensors started sending data at approximately the same moment in time (up to a few seconds sooner or later).

Third problem we encountered was that some signals on one sensors had were completely missing on other sensors. Since those occurrences were very rare, one solution could be to simply drop those signals so that at a given time all sensors have a signal starting at the same time. Fourth problem we encountered was signals with too few data points, thus we set a lower bound on how many points a signal must have.

The last problem we noticed was that some smaller repairs such as tightening of the screws and lubrication of the machine were not recorded in maintenance log and that posed a problem because after those small maintenances a small shift in the sensors were noticed.

3 Data preparation

3.1 Rationale for inclusion/exclusion

As problem creators have experimented with and excluded sensors which they deemed unnecessary, we decided to keep all of them. It may seem redundant to keep both velocity and acceleration as on is the function of the other, but relationship between them gives as useful insight in state of part. That said, some points are outliers and they will be excluded as described in following subchapter.

3.2 Data cleaning

After closer examination of the data and few ideas how to approach the problem some flaws in the data for such approach were noticed.

First thing noticed was that data was organized in *signals*, clusters of dense measurements sent by sensors that last approximately 120 seconds with a 30 second break somewhere in between. For that reason, we decided that if there is a break of more then 35 seconds between 2 measurements then those measurements belong to different signals.

The first step in cleaning the data was the problem of great discrepancy in number of measurements that a_{max} sensors send in comparison to V_{eff} sensors. As we decided

to turn clusters of dense measurements into signals, one property that seemed to be desirable was that all signals have an equal number of measurements which just wasn't possible with data as it was. For that reason we interpolated through the measurements of each signal and evaluated the given interpolant function in 100 equidistant points. The interpolation was conducted using cubic polynomials.

When working with cubic interpolation with non-equidistant points the problem that often occurs are great deviations from expected values on parts of the function with adjacent points that are very close or adjacent points that are far away. After the first three months of collecting the data on machines *FL01* and *FL07* the way data was collected from the sensors has changed from having one long cluster of data to having two subsclusters inside a single cluster. For that reason the data from the first three months was dropped as it would only cause confusion in further work. As we noticed that the gap between two subcluster inside a cluster can last up to 30 seconds, the dense measurements that are no longer then 5 seconds apart were put in the same subsignal so that each subsignal has 100 interpolated measurements.

Also, we noticed that there were measurements that were less that 0.1 seconds apart on the same sensor which might cause problems with interpolation. Since measurements with less then 0.5 seconds apart have almost the same value, those measurements were just dropped from the data. With this procedure we achieved subsignals where each pair of adjacent measurements was no less then 0.5 seconds apart and no more then 5 seconds apart, thus guaranteeing expected output of interpolation. Also, as we used interpolation with cubic polynomials we had to put a lower bound on number of measurements a cluster must have to be called a signal and that lower bound was set to 4.

The second step in cleaning the data was pairing up signals with the same timestamp of the signal. Timestamp represents time when sensors started sending measurements. The first thing noticed after the data was turned into signals is that almost all drive and idle signals start sending measurements at approximately the same time with deviation up to three seconds. The same was noticed for lift signals. Since one part of the device is responsible for movement and other for lifting (the two perform completely different operations), we decided to split the sensors into two groups. First group was named *drive* and in it were all drive and idle sensors. Second group was named *lift* and in it were all lift sensors. After closer examination, it was noticed that there exist some signals which were measured on one sensor and not the others. For that reason, if a timestamp exists (with deviation up to 3 seconds) for which a signal on any of the sensors is missing, all signals with that timestamp are dropped. Since those signals were very rare, the quality of the data wasn't distorted. All the remaining signals which were measured with the approximately same timestamp were put under the same timestamp.

In the end we got the following organization of data:

timestamp	countdown	x value	LG V_{eff}	LG a_{max}	LM V_{eff}	LM a_{max}
2017-12-12 08:47:20.663	28998759.337	8706039.246	1.306	1622.376	2.021	1026.371
2017-12-12 08:47:21.033	28998758.966	8706039.616	1.352	1573.148	2.036	1040.966
2017-12-12 08:47:21.404	28998758.595	8706039.987	1.393	1495.546	2.047	984.551
2017-12-12 08:47:21.774	28998758.225	8706040.357	1.427	1431.893	2.056	968.298
2017-12-12 08:47:22.145	28998757.854	8706040.728	1.456	1413.246	2.062	1073.795

Table 1: First five measurements of the first signal in lift data for machine FL01.

timestamp	countdown	x value	LG V_{eff}	LG a_{max}	LM V_{eff}	LM a_{max}
2017-12-12 08:47:20.663	28998759.337	8706039.246	1.306	1622.376	2.021	1026.371
2017-12-12 08:48:25.573	28998694.426	8706104.156	1.172	159.399	1.659	258.734
2017-12-12 09:15:36.257	28997063.743	8707734.840	1.515	585.155	2.208	692.923
2017-12-12 09:16:41.807	28996998.193	8707800.390	1.147	149.8	1.747	239.468
2017-12-12 09:42:51.940	28995428.060	8709370.523	1.488	633.131	2.285	664.046

Table 2: First (out of a 100) measurement of the first five signals in lift data for machine FL01.

For *drive* and *lift* groups of every of the seven machine tables are made analogous to the table showed above. Lift tables consist of 7 columns: *timestamp* column (time when measurement occurred), *countdown* column (time until next maintenance), *x value* column (value got by difference of timestamp of current measurement and timestamp of first measurement in seconds), *LG* V_{eff} column (lifting_gear_V_eff), *LG* a_{max} column (lifting_gear_a_max), *LM* V_{eff} column (lifting_motor_V_eff), *LM* a_{max} column (lifting_motor_a_max).

Drive tables consist of 11 columns each: *timestamp* column (time when measurement occurred), *countdown* column (time until next maintenance), *x value* column (value got by difference of timestamp of current measurement and timestamp of first measurement in seconds), *DG* V_{eff} column (drive_gear_V_eff), *DG* a_{max} column (drive_gear_a_max), *DM* V_{eff} column (drive_motor_V_eff), *DM* a_{max} column (drive_motor_a_max), *DW* V_{eff} column (drive_wheel_V_eff), *DW* a_{max} column (drive_wheel_a_max), *IW* V_{eff} column (idle_wheel_V_eff), *IW* a_{max} column (idle_wheel_a_max).

When organized in this way it is much easier to observe connections between different sensors of the same machine and also to use some subset of sensors to predict the behaviour of other sensors.

3.3 Fourier analysis

To begin with, let us do a brief rundown of Fourier analysis. The key concept behind Fourier analysis is that some functions are vectors forming a vector space. Just like

"ordinary" vectors, functions can be decomposed into components that we call basis. To be more specific, we define space of square-integrable functions on interval $f : [a, b] \in \mathbb{R} \to \mathbb{C}$ as set of all functions with following property:

$$\int_a^b |f(x)|^2 dx < \infty$$

Let us denote such space as $\mathcal{L}^2[a, b]$. We choose that space so that we can define a scalar product analogous to one introduced in linear algebra:

$$\langle g|f\rangle = \int_a^b g(x)^* f(x) dx$$

Where a^* denotes complex conjugate of a. It can be shown that such space ($\mathcal{L}^2[-\pi, \pi]$) has orthonormal basis consisted of sines and cosinies:

$$\int_{-\pi}^{\pi} cos(mx)cos(nx)dx = \pi\delta_{m,n}$$
$$\int_{-\pi}^{\pi} sin(mx)sin(nx)dx = \pi\delta_{m,n}$$
$$\int_{-\pi}^{\pi} sin(mx)cos(nx)dx = 0$$

Where $\delta_{m,n} = 0 \iff m \neq n$. Thus, every function can be decomposed in Fourier series:

$$\tilde{f}(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos nx + b_n \cos nx\right)$$

With coefficients:

$$a_n = 2\int_{-\pi}^{\pi} f(x)cos(nx)dx$$
$$b_n = 2\int_{-\pi}^{\pi} f(x)sin(nx)dx$$

Where $f(x)$ is original function. It can be shown that $f(x) = \tilde{f}(x)$ everywhere except at most countably many points. This procedure can be generalized for any symmetric interval where the only difference will be in constants, this one was chosen for the sake of simplicity.

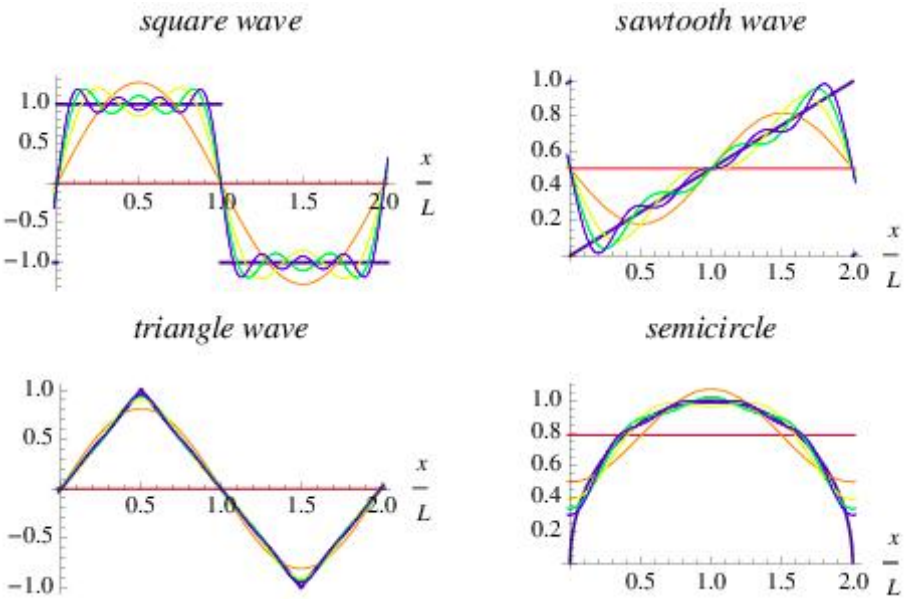


Figure 5: Illustration of Fourier decomposition of various signals.

Now we shall exploit the fact that sines and cosines are connected via Euler's identity:

$$e^{i\omega t} = cos(\omega t) + isin(\omega t)$$

And rewrite Fourier series as:

$$\tilde{f}(t) = \sum_{n=-\infty}^{\infty} e^{i\omega_n t \cdot P}$$

Where P depends on size of the interval.
Since computers cannot calculate infinite sums we will use partial Fourier series with $2N$ components:

$$\tilde{f}(t) = \sum_{n=-N}^{N} e^{i\omega_n t \cdot P}$$

And this is spectral decomposition of signal. Plots of several Fourier series of several functions is given in Figure 5.

Next, we can further generalize Fourier series on infinite domain and we are left with Fourier transform:

$$\tilde{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) \exp(-i\omega t) dt.$$

where $\tilde{f}(\omega)$ is function $f(x)$ in the frequency domain. This is very useful for our problem as it turns out that most of information about malfunctioning is hidden inside changes in power contained in frequency spectrum. For that purpose we introduce power

spectral density:

$$S(\omega)_{f,f} = \lim_{T \to 0} \mathbb{E}[|f(\omega)|^2]$$

Where $\mathbb{E}[X]$ is expected value of variable X.

Intuitively it is a measure how power is distributed across the frequency spectrum. Scipy signal implemetation of Welch method will be used for this problem as it was proven experimentally to correlate highly with maintenances and gives usable information about machine health.

3.3.1 Tsfresh

Tsfresh is a module with the purpose to extract features from time series data. For a given time series *tsfresh* outputs over 1000 features. Features include mean, variance, higher-order moments, autocorrelation for different lags, Fourier coefficients, change in quantiles, energy, entropy, different measures of non-linearity, etc.

3.3.2 PPS

During the data analysis we used another advanced property. That property is *PPS* (predictive power score). It can be perceived as a generalization of correlation which improves numerous shortcomings of correlation.

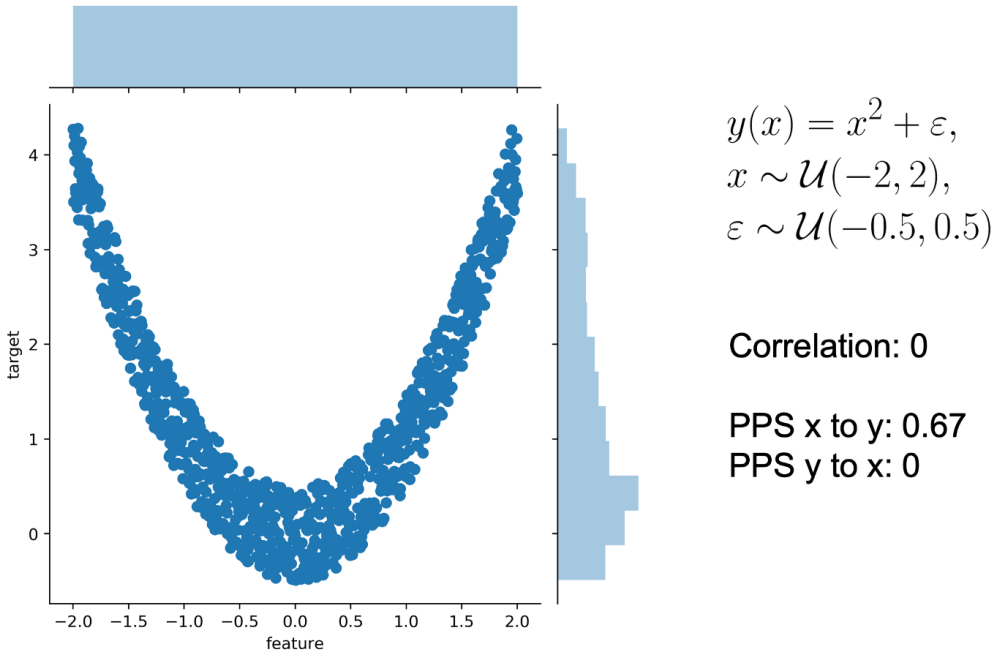


Figure 6: Comparison of correlation and PPS-a

After observing Figure 6 we deduce that correlation of that graph is 0 because correlation measures linear dependence. However, by knowing value x we can approximate what

value y is. The value of PPS of $x \to y$ amounts to 0.67, while value of $y \to x$ amounts to 0. Besides being able to recognize non-linear dependence, PPS isn't symmetrical as are not relations in nature. To illustrate this fact, suppose I have a dog. What is the probability that it is an animal with 4 legs? Almost 1. Except for deformity, all dogs have 4 legs. However, if we ask the opposite question and know that it is a 4-legged animal, what is the probability that it is a dog? Very low. Indeed, there are a lot of animals with 4 legs and that is not a sufficient predictor to be a dog.

To better understand the properties of PPS, let's ask ourselves what it means for us to have PPS 0.9. A PPS of 1 exhibits perfect predictability, but a PPS of 0.9 can be very poor or extremely favorable, depending on the choice of reference value. Namely, let's imagine an algorithm that recognizes a probability of 0.9 benign tumors, but this is not necessarily an achievement if indeed 90 % of all tumors are benign. Then that model is just as effective as the "naive" model that only chooses the category to which most members belong. In the implementation of PPS that we used, this problem was solved so that PPS was compared with the "naive" model, if it is equal or worse then PPS is 0. It is useful to note that PPS is implemented using cross-validation trees with mean square error metric.

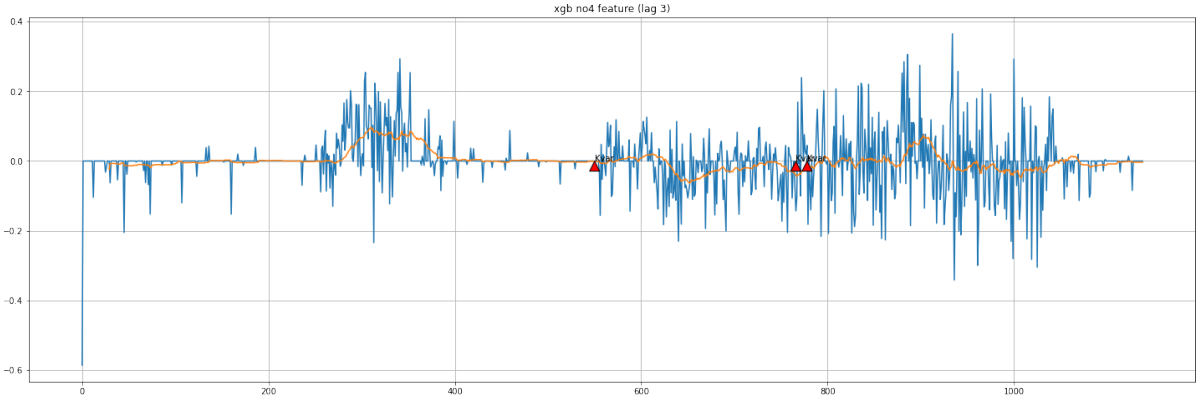


Figure 7: Asymmetry of PPS for FL01 - Idle wheel, red triangles denote logged fixes, it should be mentioned that at there were several more fixes and that when values is near 0 (begining and the end) is where machine behaved well.

We constructed following feature. For each signal we computed PPS $V_{eff} \to a_{max}$ and vice versa and than subtracted one from another. Let's call that feature asymmetry of PPS. It can be used to identify malfunctioning as for perfectly working machine this properly should tend to 0. Plot of this feature for FL01-idle-wheel is given in figure 7 above.

3.3.3 Derived properties

Our goal was to find properties that are relevant to recognizing when the repair occurred. So we first extracted all the properties offered by *tsfresh* and then made a classification

using *XGBoost*. The classification was made so that we wondered if the machine had been repaired in a period of a month. Two properties proved to be extremely useful, namely the c3 coefficient and the change in quantiles. The c3 coefficient is given by the formula:

$$c_3(x, lag) = \frac{1}{n - 2lag} \sum_{n=1}^{n-2lag} x_{i+2lag} x_{i+lag} x_i$$

What is actually:

$$c_3(x, lag) = \mathbb{E}[L^2(X) \cdot L(X) \cdot X]$$

Where $\mathbb{E}$ is the expected value, and L is the delay operator of the random variable X. More intuitively, it is a measure of the non-linearity of a time series. The obtained delay parameter is equal to three.

Another property that has proven useful is $change_{quantiles}(x, q_l, q_h, isabs, f_{agg})$. With this property, we fix a corridor in which we observe the average change in the absolute values of the required properties. Specifically: we obtained the parameters $(q_l = 0.2, q_h = 0.6, isabs = True, f_{agg} = var)$. In words, it is a change of variance in the corridor from 20 % of the lowest values to 60 % of the lowest values of variance.

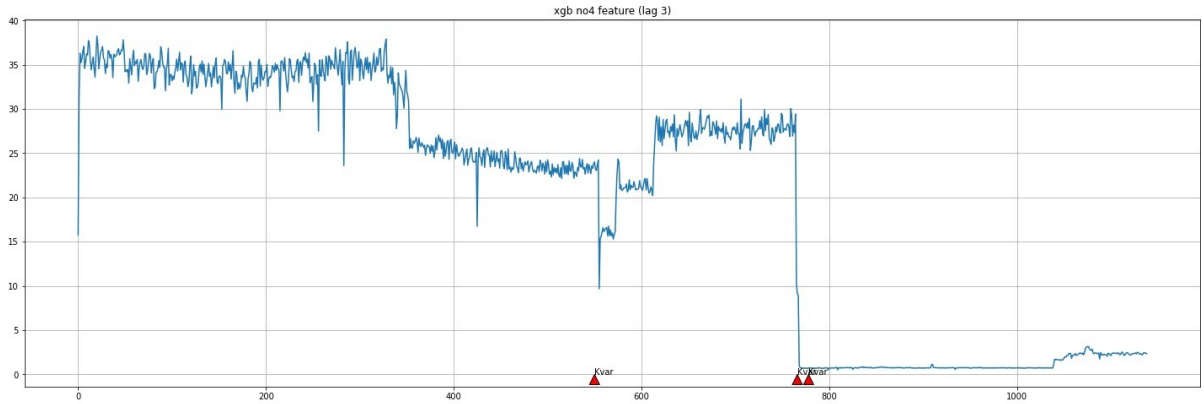


Figure 8: xgboost no4 feature (lag3) - Feature that captures maintiances well - red triangles denote logged maintenances. It should be mentioned that there were several more smaller maintiances on the device as are visible in the picture.

While feature plotted on figure 8 shows when the maintenances occurred, it has no predictive value because it is almost constant on parts before the maintenances. This motivates us to search for a feature which decreases as the machine deteriorates, and increases after a maintenance is conducted.

4 Modelling

4.1 Hypothesis

Our hypothesis is that machine changes it's spectral power density as it begins to malfunction. It will be best observed in high frequency spectrum, which is intuitive because of fatigue materials lose it's elastic properties. We shall define machine health function with following properties:

- as the machine breaks down the value of the function increases monotonically
- after repair we expect a sudden decrease in function
- if the machine is working properly the function is approximately constant

Our model takes interval of power density found via Welch method of highest frequency and than it performs Dickey-Fuller test so we can see weather or not time series is stationary. A stationary process is a process whose unconditional joint probability distribution does not change when shifted in time. In other words we can deduce whether fluctuations are normal and whether trend is just a consequence of random behaviour. If times series is not stationary that is a clear indication that machine started to malfunction and maintenance should be performed in near future.

4.2 Modeling technique

4.2.1 Assumptions

The assumption that our model uses is that the machine has enough logged data so that our model doesn't fluctuate. Then our model should yield stable predictions.

4.2.2 Method

Our philosophy is that common sense should always guide us rather then using some neural network without any intuition on the problem. Advanced methods should be used after the intuitive methods give some results and therefore give us a way to approach the problem. If such advanced methods outperform those baseline models, then those intuitive models should be replaced. Firstly, we loaded parsed data from algorithm described in previous chapters. Then, we introduce new function *getInt(start,end,index1,index2,part)* which takes as arguments start of the interval, end of the interval, machine index, second index which is 0 or 1, 0 belongs to drive and idle mechanisms and part u lifting mechanism, lastly it takes sensor name. This function first demeans and detrends series. Reason for that is that we observed from the data that machine was moved in one direction while data was recorded, that there was brief pause and machine was moved backwards. This

way we can only get vibrational degrees of freedom and not worry about translational ones. Next, function performes Welch algortihm for spectral density in highest frequency and appends that value into list. Process is repeated for all signals in interval and time series is returned. Next, we test that interval using statsmodels.tsa.stattools implementation of Dickey-Fuller test to see if time series is stationary. That is done in *testInt(signal)* function. Since after maintenance, value of our feature decreases and *testInt* would recognize that as non-stationarity we developed a helper function *getSlope(t,signal)* which returns slope of linear regression over that interval if it is negative, *testInt* would still return 'True' even though series could be non-stationary. We wrap it all up with function *findMaintenance(machine,interval)* which automatizes the whole process. For given machines and interval length it finds all instances in time when faulty behaviour was observed and returns date as well as on which part anomaly was spotted. To find date function *dateIndexParser(data,index)* parses index of a signal where maintenance should be performed with a date.

5 Evaluation

There is no objective way to evaluate our data as logged maintenances have been preformed long before machine was about to break. In spite of that there should still be general agreement in fix times. There is however one problem. We can't test whether our model finds a need for maintenance if maintenance hadn't occurred. Since maintenance process disrupts data to great extent it is easy to see when maintenance has been performed. Our model detects anomalies in data but we cannot claim that those are necessary signs of machine malfunctioning. It could simply be consequence of repair process. That being said, we detected some anomalies up to 10 days before logged maintenance was performed and we are confident in our method as a reliable way to detect machine malfunctioning. Crucial thing to keep in mind is that the biggest cost is human work, so that if grinding of rails is preformed on which more machines operate, small maintenance can be preformed on all those machines as that can increase their lifespan.

6 Deployment

6.1 Final report

After comprehensive data analysis and evaluation of many features and methods, we found that the best feature is spectral power density of low frequency for effective velocity sensors. More advanced methods such as sensor prediction from other sensors via neural networks and searching where the real and predicted signals differ were very unstable and didn't perform as well as our common sense approach.

Velocity sensors turned out to be more useful in general as they are less noisy then their acceleration counterparts. Nevertheless, acceleration sensors proved useful for data clustering as shown in Figures 2 and 3.

Our model predicted a need for maintenance on FL01 that was performed on 08.02.2019 already on 30.11.2018. Next, we found anomaly on FL03 on 03.04.2019., while maintenance was performed 2 days later. Next, we found anomalies on FL02 and FL07 in range when maintenance was performed. We also found evidence of other maintenances:

- FL01
  - 19.03.2019.
  - 12.05.2019.
- FL02
  - 5.06.2019.
- FL03
  - 01.05.2019.
  - 04.06.2019.
  - 09.07.2019.
- FL04
  - 05.04.2019.
  - 19.05.2019.
  - 23.07.2019.
- FL05
  - 08.04.2019.
  - 05.05.2019.
  - 18.06.2019.
- FL06
  - 23.05.2019.
- FL06
  - 27.12.2017.
  - 06.03.2018.

- 10.09.2018.
- 11.06.2019.

Consecutive anomalies have been omitted because they just represent early detection and kept repeating until problem had been fixed.

6.2 Monitoring

Our model should run once every few days on latest data. Output of the model should be checked for a date when the first problem on sensors was encountered. If such a date should appear, maintenance should be scheduled. Model will always evaluate current state of machine.

6.3 Project review

During this project we had a lot of fun as it was really hard to pinpoint which features correlate with maintenances. Part of the reason was very sparse maintenance log with almost zero information. A more detailed log would be of great use, for instance logging of evaluation of machine health, which could be used for survival regression analysis. The other reason is very noisy data.

Besides vastly increasing our signal processing knowledge we also learned a valuable lesson - problem should first be approached with simpler solutions until we familiarize ourselves with the project and gain intuition on the problem. Only afterwards should more advanced methods be used. Sometimes simpler and more intuitive solutions do outperform the more advanced methods.

Some methods such as PPS have not been part of final solution but they have potential applications in other systems.