European Rotorcraft Forum

HELICOPTER DATA SCIENCE - FROM CONCEPTS TO APPLICATIONS

Ammar MECHOUCHE, Ammar.Mechouche@Airbus.com, Airbus Helicopters (France)

Abstract
This paper presents three application use cases of Machine Learning (ML) - under different forms: classification, regression and clustering - in order to improve the exploitation of helicopter data in different business problems. In each use case a real data is used and promising results are highlighted. After presenting the use cases and obtained results, conclusions address the perspectives as well as the challenges that ML and data science in general open in terms of learning from experience for helicopter continuous improvement.

1. ABBREVIATIONS

AI – Artificial Intelligence
FDCR – Flight Data Continuous Recorder
FH – Flight Hours
FT – Flight Test
FTI – Flight Test Instrumentation
H/C – Helicopter
HMI – Human Machine Interface
HUMS – Health and Usage Monitoring System
IDF – Inverse Document Frequency
ML – Machine Learning
PSI – Population Stability Index
RUL – Remaining Useful Life
SAR – Search And Rescue
TBO – Time Between Overhaul
TF – Term Frequency
TRP – Tail Rotor Power
UC – Use Case
VS – Virtual Sensor

2. INTRODUCTION

Data science allows exploiting data in a creative way to generate business value. This implies a good understanding of business problems and the underlying data, but also good technical skills regarding data visualization, scripting, statistics and AI. Applied ML, which is part of AI, is one lever of data science. It consists of the application of classification, regression or clustering algorithms to a specific domain, in order to extract hidden patterns / rules from past experience conveyed by data collected in that domain.

Within the helicopter industry, a huge volume of time series data is collected during flight tests through Flight Test Instrumentations (FTI), and also from the variety of customer helicopters through the Health and Usage Monitoring Systems (HUMS), in addition to maintenance data and the Flight Data Continuous Recorder (FDCR). The (joint) analysis of all this data brings considerable benefits in terms e.g. of safety, maintenance optimization, system design improvement and in-service incidents support [1].

This paper presents three original use cases of a successful application of ML with helicopter data at Airbus.

- The first use case is a classification problem. It shows how ML can be used to automatically detect measurement errors in vibration data in order to avoid a wrong interpretation of the signal, which can then lead to inappropriate decisions,
- The second use case is a regression problem. ML is used to build Virtual Sensors (VS) in general, with a focus on the Tail Rotor Power (TRP) VS developed using flight test data and deployed on ~100 000 Flight Hours (FH) customer data which do not include the TRP parameter; since it is not measured for the studied helicopter type. The use case shows also the benefits of such a VS for the maintenance of helicopters,
- The third use case is a clustering problem. It shows how flight data can be used to automatically group together helicopters having similar flight profiles and conditions, which can be used to anticipate failures or avoid operational incidents.

The paper, first, provides an overview of ML, including elementary and high level definitions of supervised / unsupervised ML, classification, regression, clustering, linear models and non-lineal models as well as metrics for evaluating and comparing models. Then, each use case is detailed: methodology, results and discussion. Finally, conclusions address the perspectives as well as the challenges that ML and data science in general open in terms of learning from experience for helicopter continuous improvement.

3. ELEMENTARY DEFINITIONS

This section provides brief elementary definitions regarding ML that allows the understanding of the rest of the paper. More detailed definitions are widely available on Internet.

Machine Learning

ML algorithms consist of detecting patterns and learning how to make predictions by processing data and experiences, rather than by receiving explicit programming instructions. Assume we seek to estimate a variable *y* using a set of other variables $X = (x_1, x_2, \ldots, x_n)$: $y = f(X)$. Supervised ML consists of the estimation of the mapping function *f* using learning and labelled data. When *y* is a continuous (numerical) variable we speak about regression. If it is categorical then we are in a classification case. If only *X* is given and no examples for the corresponding output *y*, then we are in the case of unsupervised learning where the purpose is to model the underlying structure / distribution of the data to learn more about it.

Linear vs. Nonlinear Models

A linear model is characterized by the equation 1:

$$y = a_0 + a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \quad (1)$$

Linear models are linear in the parameters, where the output can be represented as a linear combination of functions of the independent variables. E.g. $y = ax^2$ is a linear model, and $y = cos(ax)$ is a nonlinear model.

K-fold Cross-validation: A ML model testing method, where the dataset is divided into *k* subsets. Iterations are then made on the subsets, and each time a model is trained on k-1 subsets put together, and tested on the remaining subset. Finally, the average error across all k tests is computed. In the following, this is referred to as cross-validation.

Precision: a test metric that quantifies the number of positive class predictions that actually belong to the positive class.

Recall: a test metric that quantifies the number of positive class predictions made out of all positive examples in the dataset.

F-Measure (f1): a metric that combines precision and recall metrics as follows: *f1 = (2 * Precision * Recall) / (Precision + Recall)*. It provides a single score that balances both the concerns of precision and recall in one number[1].

Coefficient of determination (R2): represents the proportion of the variance in the dependent variable *y* that is predictable from the other *x-variables*. It ranges from 0 to 1 (i.e. 0% to 100% of the variation in *y* can be explained by the *x-variables*).

Mean Absolute Error (MAE): measures errors between observations expressing the same phenomenon. Adapt to compare different estimators.

[1] https://en.wikipedia.org/wiki/F-score

4. APPLICATION USE CASES

4.1. UC#1 – Defect Detection in Vibration Data

The measurement systems allow to transcribe a mechanical physical quantity into an electrical signal that can be interpreted by humans [2]. These systems often consist of an acquisition chain including sensor, wiring, acquisition card, converter, and recorder. Each of these components is likely to fail, temporarily or permanently, and generate faults in the measured signal. It is imperative for the analyst to detect these faults upstream so as not to distort the interpretation of the signal, which could lead to erroneous decision. The generalization of measurement systems and the volume of data recorded on helicopters lead us to think about the use of algorithms capable of automatically detecting such defects in signals.

Expert rule-based methods which are mostly used in this context are clearly not scalable – in terms of lead time development - regarding for example taking into account of news faults and dynamic thresholding.

We introduce here the use of ML to show its potential for automatic detection and classification of defects that might be present in vibration signals acquired on helicopters. For that, we are considering signals without defects (Reference) as well as signals with the following defects: Bearings fault, Cabling issue, Saturation, Step Change and Non stationarity. Figure 1 depicts some of them.

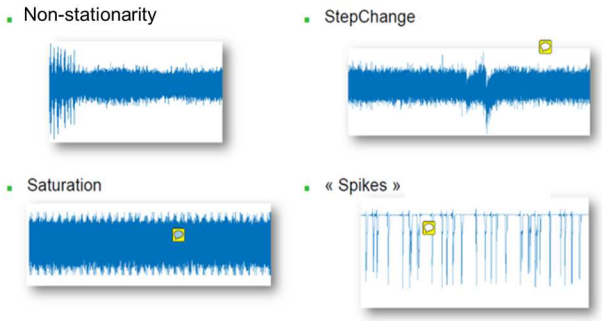

Figure 1: Illustration of some considered defects types to be detected by the algorithm.

The objective is to build a classifier which is able of detecting and categorizing different types of above defects. Hereafter the main steps followed to build the classifier.

1. *Engineering features generation on the temporal and frequency domains of data*

It is quite difficult to reason on the whole signals, mainly because of the high number of data points to analyse. Instead, features are generated and used to learn more about that signals. Here we are

generating features specified by signal processing experts.
Kurtosis is an example of such engineering features computed in the time domain.

- Kurtosis

$$\frac{\mu_4}{\sigma^4}$$

With:

- μ_4 : the fourth central moment
- σ^4 : standard deviation

Below some examples of engineering features elaborated in the frequency domain computed using Fast Fourier Transform (FFT). They provide statistical information about the frequency content of the signal.

- Frequency center

$$\frac{\sum_{i=1}^{N} f_i * p_i}{\sum_{i=1}^{N} p_i}$$

- Root variance frequency

$$\sqrt{\frac{\sum_{i=1}^{N} (f_i - center)^2 * p_i}{\sum_{i=1}^{N} p_i}}$$

With:

- p_i : $Power\ spectrum\ of\ x(i)$
- N : $Number\ of\ spectrum\ lines$
- f_i: $Frequency\ value\ of\ the\ i-th\ spectrum\ line$

2. Validation of the features using a visualization method

Features generation from signal data is a key aspect, since summarizing a signal with features is necessarily accompanied by a loss of information. Assessing the quality of the considered features is then important. We propose here a visualization method for this purpose.

On Figure 2 we can visually evaluate the relevance of the selected engineering features, based on an algorithm that allows 2D projection of the engineering features thanks to multi-dimensional scaling (MDS). Each signal is then represented by a data point and coloured according to its fault label. We can see for example that the group “Step Change” is far from other groups (except for “Saturation” group), which means that it is clearly separable from other groups using the selected features.

From this plot showing that different groups of faults could be well separated, we can predict that building a performant classification model is possible.

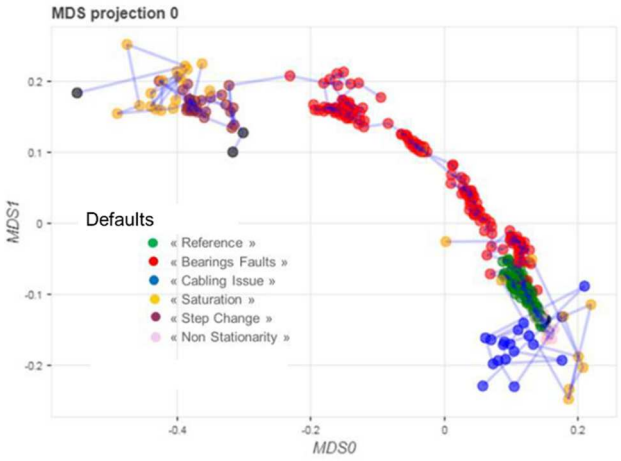

Figure 2: Visual assessment of the relevance of selected engineering features.

3. Learning data preparation

The learning data is composed of 390 samples, each corresponding to a signal summarized with: Firstly the features generated from it as described in the previous step (independent variables); and secondly its defect label (dependent variable).

No cleaning nor transformation was applied on this learning data. They were considered as delivered and annotated by the vibration expert.

4. Classifier construction

Once the learning data is prepared, multiple classification algorithms were applied: logistic regression, Gaussian mixture, k-nearest neighbors (KNN), random forests and Ada boost. This consists of the learning of a correlation between engineering features generated from the signals and their labels. The idea is to make predictions based on a majority vote, but if no majority is found then the result of the more pre-prioritized algorithm is retained.

5. Results

Preliminary results are obtained using simple validation (one iteration of a cross validation): 60% of the data is used to train the model and the remaining 40% of samples was used to make the tests. The table in Figure 3 reports the results, especially in terms of precision and recall for each output class. As we can see the values for both metrics are very high for each considered defect. Precision is ranging from 85% to 100%, and recall from 75% to 100%.

```
======================================================
                      Voting
======================================================

                  precision    recall  f1-score   support

   Bearings_Faults       1.00      1.00      1.00        68
     Cabling_issue       1.00      1.00      1.00         8
Non_stationnarity        1.00      0.75      0.86         8
         Reference       0.96      1.00      0.98        48
Sensor_saturation        0.91      0.83      0.87        12
       Step_change       0.85      0.92      0.88        12
```

Figure 3: Classification results obtained with the majority voting strategy.

The ultimate phase is to deploy the built classifier in order to continuously classify new signals. Whenever a new signal arrives it will go through the classification model and automatically tagged with the appropriate label (Figure 4).

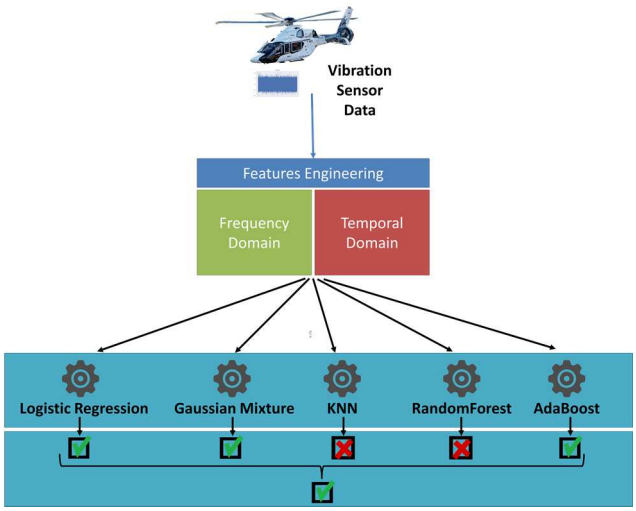

Figure 4: Continuous classification of vibration sensor data. [2]

Discussion

This use case showed that ML could be useful to automate and scale up the process of detecting defects in vibration data. The tested method is quite simple and relies on classical ML algorithms. It can be improved w.r.t several aspects. First, tests could be consolidated with a full cross-validation in order to ensure the high precision/recall values obtained are not due to overfitting. Also, the generation of engineering features could be automated using features generation tools such as TS-Fresh[2] jointly with optimization algorithms for best features selection as proposed in [3]. Moreover, the problem can be turned into an anomaly detection problem using unsupervised learning and only features generated from signals. Such an approach is proposed in [4,5]. Finally, the features generation step can be completely dropped and make the analysis directly on the signals. One way to achieve that, is to transform the signals into images and exploit deep learning classification algorithms / architectures developed in the literature for image classification in order to detect anomalies as proposed in [6].

4.2. UC#2 – Tail Rotor Power Virtual Sensor

VSs for helicopters consist of the enrichment of customer data with non-measured parameters or the duplication of existing ones. In the former case, we rely either on physical models or models learned from the very rich flight test data that makes it possible to correlate parameters which are present both on prototype and customer helicopters, with other parameters measured only on prototype helicopters. In the latter case, the model is built using in general ML with customer big data such as the one presented in [7]. In both cases the benefits are important for helicopter maintenance optimization and safety.

The use case presented here is about the application of ML to build a nonlinear model for the estimation of the TRP. In fact, TRP is measured only on prototype helicopters thanks to a complex FTI that requires frequent adjustments (Figure 5). This makes it not practical for customers to have it installed on their machines. However, TRP is needed for multiple important applications such as supporting incremental TBO extension of intermediate / tail gear boxes during maturity phase, comparison of real usage and design spectrum, failure anticipation, tail vibration indicators filtering and maintenance credit [8].

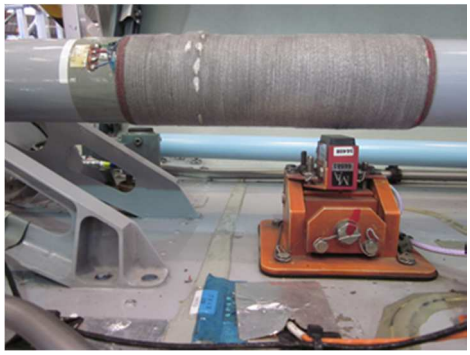

Figure 5: FTI for TRP measurement.

In the reminder of this section, the methodology of building the TRP VS will be detailed, then the results will be discussed and compared to those obtained

[2] https://tsfresh.readthedocs.io/en/latest/

with an analytical model developed by engineers. After that it will be shown how the built model can be deployed and monitored in production for processing new customer data. Finally, possible improvements of the method are discussed.

1. Model Construction Methodology

Hereafter a description of the data preparation and model learning steps that are followed to build the TRP VS.

a. Data preparation

First, with the support of an expert a set of parameters were identified as impacting the TRP. Then, these parameters as well as the TRP parameter were extracted from FT data of the considered helicopter (here H175). This represents about 50 FH. Some cleaning was finally applied in order to remove outliers that do not correspond to a physical reality (Figure 7 TOP-LEFT).

b. Learning

The learning data base formed in the previous step is divided into 2 subsets: the bigger one is used to train, optimize and validate a model by testing the following ML algorithms:

- *Random Forests*[3]: they are an ensemble learning method for classification, regression that operates by constructing a multitude of decision trees (nonlinear models) at training time. For regression tasks, the average prediction of the individual trees is returned.
- *Gradient Boosting*: also ensemble learning method but unlike random forests, Gradient Boosting algorithm sequentially combines weak decision trees in a way that each new decision tree fits to the residuals from the previous step so that the model improves. The final model aggregates the results from each step and a strong learner is achieved.
- *Linear Regression*: a linear approach to modelling the relationship between a scalar response and one or more explanatory variables.

The other subset is used to test the performance of the retained model (Figure 7 TOP-LEFT).

2. Results

A cross validation was performed and the results showed that the Gradient Boosting algorithm is the one which outperforms slightly Random Forests (R2 = 0.94, MAE = 4.25 vs. R2 = 0.87, MAE = 6.54 respectively) and outperforms clearly the linear regression (R2 = 0.64, MAE = 15.68) as shown on Figure 6. So, the retained model for deployment is the one built using the Gradient Boosting algorithm. On Figure 7 TOP-RIGHT we can see how predictions of the model (in green) are close to measured test data (in red). In blue, the result computed with the analytical model is less accurate than the ML model. It often over-estimates the reality because it was partially tuned using a polar estimated on FT data corresponding to hovering flight phases in which TRP is increased. Thus, an over-estimate can be observed during all forward flight phases where increased TRP is not required.

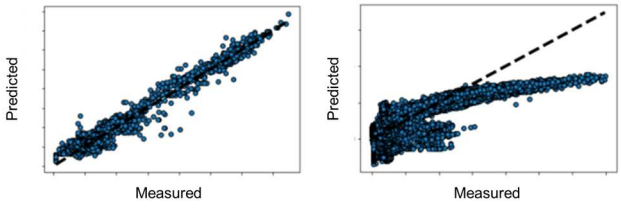

Figure 6: Non linear model results (R2=0.94 on the left) vs. linear model results (R2= 0.64 on the right).

As intermediate result of the learning process we get also the importance of each input parameter of the

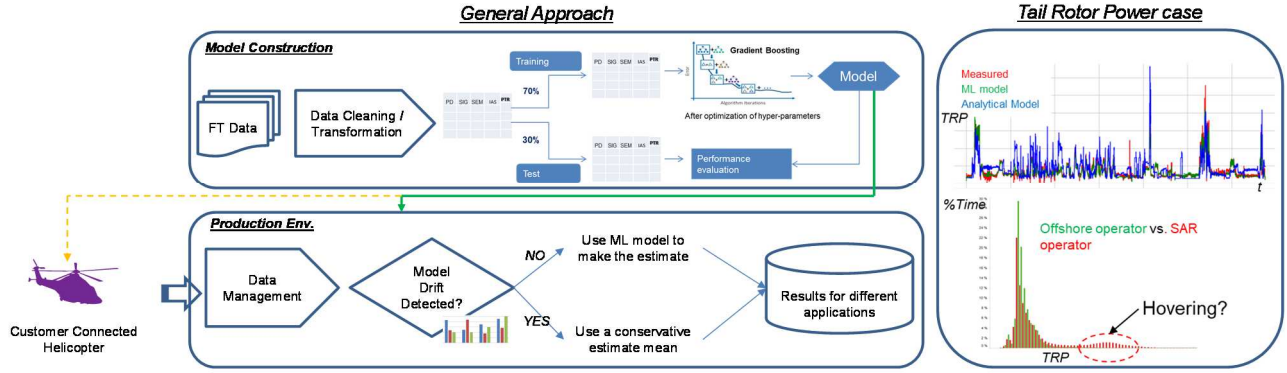

Figure7: Overall learning and deployment process of TRB VS.

[3] en.wikipedia.org/wiki/Random_forest

model. This information is important to understand what is impacting more the output of the model, and in this use case it is also used in the monitoring phase of the model as it will be detailed in the next section.

Moreover, partial dependence plots[4] can be used to visualize and analyze interaction between the target response (TRP here) and a set of input parameters. Figure 8 shows for example estimated marginal effect of one input parameter of the model (Indicated Air Speed (IAS) which is of medium importance) on TRP when all other variables are held at their average[9]. This helps to partially understand predictions of complex and nonlinear models.

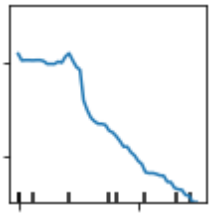

Figure 8: Partial dependency plot representing the estimated marginal effect of IAS (x-axis) on TRP when all other variables are held at their average.

3. *Deployment*

The deployment consists here of the integration of the model into an existing on-ground production environment in order to continuously process new flights. So, the model takes as input the set of parameters determined during the development phase, and returns the TRP values for each moment of the customer flight. Then, results are stored in a database in order to be used in making practical business decisions.

The deployment could also be considered on board on helicopter, but this requires available on-board processing capabilities, additional specifications and possibly a certification if the targeted application is critical such as maintenance credit / alleviation.

The deployment in our case was done following 2 steps: first, the model was implemented in the big data platform [2] to estimate and store TRP results for all past flights already collected since the entry into service of the H175 helicopter. This represents more than 100 thousands FH. On Figure 7 BOTTOM-RIGHT we can see the results computed for 2 operators: a Search & Rescue (SAR) operator and an offshore operator. For SAR operator we can see more percentage of time spent in higher TRP values corresponding probably to higher time spent in hovering phase during operations.

Then, the model was implemented in the customer oriented platform to continuously process new coming flights (Figure 7 BOTTOM-LEFT).

4. *Monitoring*

As explained before, the model was learned from small FT dataset of small number of helicopter prototypes flown by few pilots in few locations. But it is deployed to augment data coming from customer helicopters located in different regions of the world, flown by different pilots and performing different types of missions. So, domains of the input parameters of the model during the execution phase might be different from what they were in the learning phase. This, could lead to incorrect predictions of the model and consequently to wrong (critical) decisions.

To prevent from this, we propose here to combine the Population Stability Index (PSI) statistical method developed in the literature [10] with the importance of the model' input parameters computed during the learning phase.

PSI is an industry standard for measuring how much a population has shifted over time or between two different samples of a population in a single number[5]. It can be used pro-actively to choose features or input parameters of the model during the learning phase that are not prone to rapid changes after deployment. Or, as in our case here it can be used in a reactive way as a trigger to relearn the model or apply an alternative model.

To understand how different two populations are, PSI does a bucketing of their underlying distributions and compares the percent of items in each of the buckets thanks to the formula 2. The result is a number that gives an idea of how the two populations are different. PSI results are in general interpreted as follows:

- IF (PSI < 0.1) THEN no significant change between the compared populations
- IF (PSI < 0.2) THEN a moderate change exists between the compared populations
- IF (PSI >= 0.2) THEN a significant change exists between the compared populations

$$(2) \qquad PSI = \sum((Actual\% - Expected\%) * \ln(\frac{Actual\%}{Expected\%}))$$

[4] https://scikit-learn.org/stable/modules/partial_dependence.html

[5] https://mwburke.github.io/data%20science/2018/04/29/population-stability-index.html

With:

- *Actual%:* the % of values in each bucket of the first population.
- *Expected%:* the % of values in each bucket of the second population.

PSI calculation is done for each bucket, then summed overall buckets for the distributions.

Illustrative example (code available here[6])

Let's consider the 2 distributions on Figure 9 which could be distributions of one input parameter of our model in learning and execution phases, respectively. Suppose the initial population is the blue one and the new population is the orange one. Visually we can observe a slight shift (drift) of the population. So, the objective here is to use PSI method to quantify this drift.

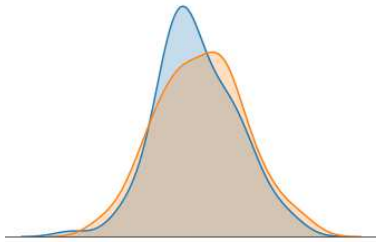

Figure 9: Two population distributions comparison using PSI. Image generated using the code mentioned above.

First, the initial population range is divided into a certain number of buckets (here we choose arbitrary 9). Then, the percentage of values in each of those buckets is calculated for the initial and new populations (Figure 10). Finally, PSI is calculated as specified before (here we obtain PSI = 0.103). Now, thanks to the PSI rules we can affirm that there is a moderate change w.r.t the initial distribution.

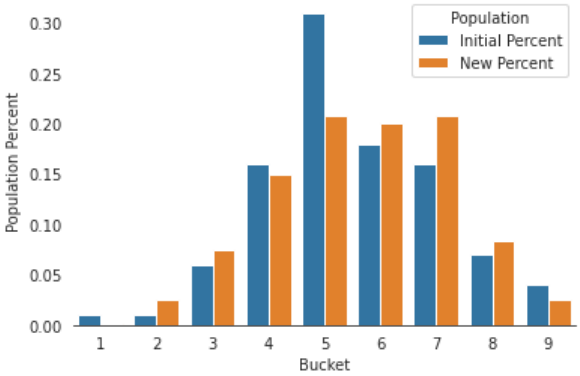

Figure 10: Population distributions bucketing (9 buckets). Image generated using the code mentioned above.

Combination of PSI with parameters importance

We propose in this work to combine PSI result with model's input parameters importance as follow:

For each input parameter of the model:

- IF(No_Significant_Population_Change) THEN PARAM_OK
- IF(Moderate_Population_Change && High_Importance) THEN PARAM_KO
- IF(Moderate_Population_Change && NOT High_Importance) THEN PARAM_OK
- IF(Significant_Population_Change && Low_Importance) THEN PARAM_OK
- IF(Significant_Population_Change && NOT Low_Importance) THEN PARAM_KO

All defined rules as well as semantics associated to the parameters importance could change depending on the criticality of the application.

The prediction of the model is then considered as reliable if all its input parameters are tagged PARAM_OK. Otherwise, an alternative and conservative model should be considered for making the estimate. In our case the analytical model developed by engineers, which is more conservative than the ML model, is considered as alternative model since its output is based more on mechanics equations than FT parameters distributions (Figure 7 BOTTOM-LEFT).

Application to TRP case

PSI was calculated for each input parameter of the TRP model using more than 50 thousands H175 flights representing more than 100 thousands FH. Then, PSI values were averaged by helicopter serial number, by operator and by mission type in order to have a global assessment of the data drift w.r.t FT data used to learn the model.

As an example, for pedal position parameter, which is the most impacting parameter for TRP, PSI averaged values are under 0.10 for all mission types (Figure 11). This means that pedal position domain is almost the same for customer and prototype helicopters.

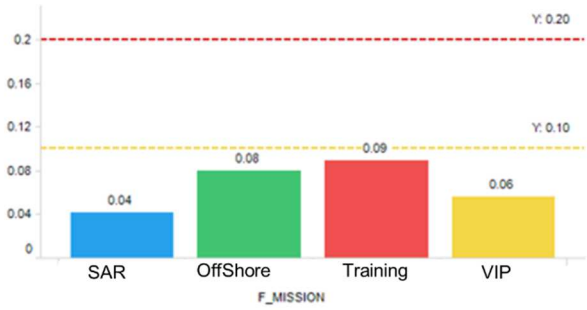

Figure 11: PSI computed values per mission type for pedal position parameter.

[6] https://github.com/mwburke/population-stability-index

In contrast, a moderate drift is observable regarding IAS parameter. As we can see on Figure 12, all PSI values are between 0.10 and 0.20. This means that there is a moderate difference of the IAS domain between customer and prototype helicopters. Nevertheless, since its importance is medium and its PSI indicates moderate change then the model still is applicable according to our defined rules.

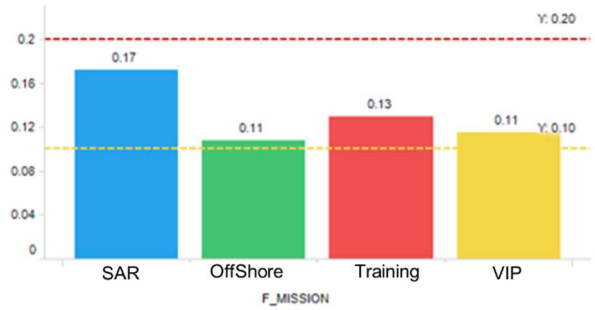

Figure 12: PSI computed values per mission type for IAS.

Discussion

The presented use case showed how ML could be efficiently used to build a useful virtual sensor for customer helicopters using data coming from prototype helicopters. The presented approach was complete and described all steps from design to production. The generated results can be already used in different applications.

The presented approach could however be improved. First, additional available FT data could be considered in the learning phase in order to better cover the flight domain. Also, a deeper optimization of Gradient Boosting hyper parameters – even very time consuming - could improve the precision of the model. Finally, the adapted PSI method is very interesting, however it considers one input parameter at a time. It is worth to investigate other methods or develop new ones that can combine different parameters, possibly in the way they are combined in the learned model. Moreover, PSI' statistical properties are worth to be investigated further such as done in [16], especially regarding the impact of fixed number of buckets on the final result.

4.3. UC#3 - H/C Flight Data Based Clustering

This use case is more an original idea of an application involving Human Machine Interaction (HMI), statistics and unsupervised learning, namely hierarchical clustering which intends to group together helicopters with similar flight profiles and conditions. The purpose is to support experts later on in incident troubleshooting and failure anticipation. The main interactions of the user with the application are the following:

1. First, a helicopter of interest (E.g. H/C ID 46852 on Figure 13) as well a group of H/Cs for comparison (E.g. other H/C IDs on Figure 13) are selected. Here the helicopter of interest is not impacted by any incident,
2. Then, the user chooses the flight phases to be analysed, and also a set of parameters to consider during each selected phase. E.g. On Figure 13 the user wants to analyse Outside Air Temperature (OAT) during ground phases, considering the last *X* FH of each H/C,
3. After that, corresponding data is extracted based on precomputed flight regimes and pre-defined intervals of each parameter (Figure 13). Then, data availability is checked and the top discriminating phases and conditions are automatically determined based on an adaptation of the *TF*IDF*[7] concept. In our illustrative example GROUND →]25°C, 30°C] and GROUND →]30°C, 35°C] were identified as more specific for the helicopter of interest,
4. Finally, based on the discriminating phases returned by the system and validated by the user, an automatic clustering is performed (Figure 14) in order to identify H/Cs with similar flight profiles and conditions. This way, it is possible to identify risky H/Cs if the H/C of interest was impacted by an incident. In our illustrative example based a real data, we see on Figure 14 that mostly helicopters of the same operator are grouped together, since in general they perform the same mission type (E.g. Operator ID 20 for the helicopter of interest). The only H/Cs which were not grouped together whereas they belong to the same operator 112, are 48260 and 46875. Indeed, according to the information we got from the customer support, the data extracted for the *X* last FH of the H/C 48260 correspond to a period in which this machine was used for training by the customer, so grouped with machines performing a mission close to training (here SAR mission).

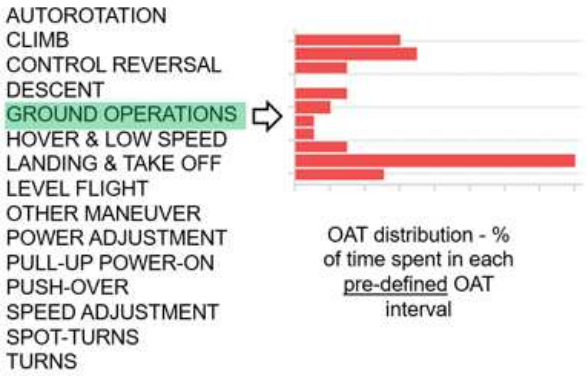

Figure 13: Selection of flight phases and parameters to be considered in the analysis.

[7] https://en.wikipedia.org/wiki/Tf%E2%80%93idf

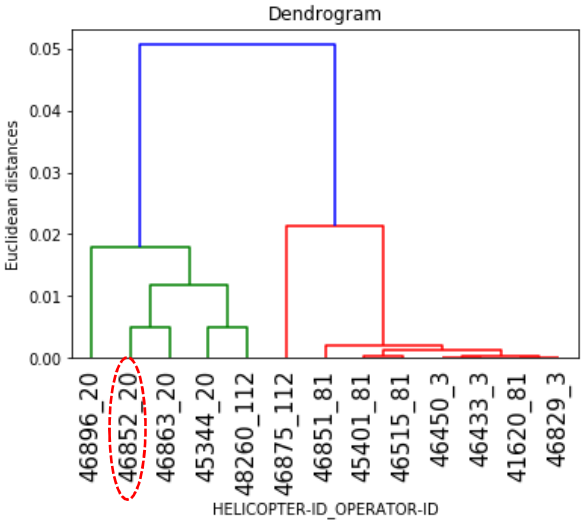

Figure 14: Hierarchical relationship between studied H/Cs (H/C-ID_OPERATOR-ID) based on the retained discriminating flight phases and conditions. Euclidean distance is used to assign data points to the closest cluster centroid.

In step 3 it is mentioned the adaptation of the *TF*IDF* concept. In fact, this method is used in information retrieval domain which is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus [11].

- TF means Term Frequency, which is the number of times a term occurs in a document of a collection or corpus. In other terms, the weight of a term in a document is proportional to its frequency in that document. There are several ways for computing the term frequency of a term *t* in a document *d*, but the simplest and more used one is the following (formula 3):

$$(3) \qquad tf(t,d) = \frac{f_{t,d}}{\sum_{t' \epsilon d} f_{t',d}}$$

- IDF means Inverse Document Frequency, it reflects the rarity of the term *t* in a collection of documents *D*, so how much it is informative. It is computed as follows (formula 4):

$$(4) \qquad idf(t,D) = \log \frac{\mathrm{N}}{|\{d \epsilon D : t \epsilon d\}|}$$

  - *N*: Total number of documents in the corpus *D*.
  - $|\{d \epsilon D : t \epsilon d\}|$: Number of documents that contain *t.*

Now, the TF*IDF is calculated as follows (formula 5):

$$(5) \qquad TF * IDF = tf(t,d) * idf(t,D)$$

The idea is to give a higher weight to terms that are frequent in a given document but rare in the whole collection.

*TF*IDF adaptation*

In our application, the TF*IDF was adapted as follows:

- A term corresponds to a flight phase.
- A document corresponds to one H/C.
- *D* corresponds to the set of helicopters selected for analysis.
- *N* corresponds to the total number of helicopters selected for analysis.
- $f_{t,d}$: corresponds the time spent in a given flight phase *t* for a given helicopter *d.*
- $|\{d \epsilon D : t \epsilon d\}|$: Corresponds to the number of helicopters that have flown in the flight phase *t* during the considered time period.

TF*IDF assigns for each flight phase a weight in each helicopter. For a given helicopter, its discriminating flight phases are those with higher weights.

Discussion

The presented use case showed an original interactive data-based application that allows a user to better understand an event regarding one helicopter when considering the data from several helicopters, and eventually identify helicopters at risk. The interactive aspect is important, the final output of the application is a result of the user point of view and insights extracted by algorithms from the data. No similar application with the same objective was found in the literature. Another original aspect of the use case is the adaptation of the TF-IDF concept. In fact, to compare two groups of individuals we could have used classical statistical indicators such as T-student, Anova test, Hotelling's T-Squared, etc. However, in addition to their complexity for a non-specialist, these statistical indicators are often applicable when we have a high number of individuals (often > 30) and they also suppose particular distributions of the data which is rarely the case in real life facts. The adapted method, in turn, is clear, simple, applicable even with few individuals and easy to understand and implement. Moreover, it gives acceptable results. A mock-up is already designed for the implementation of the idea of the use case. Developments are ongoing with the objective to present a prototype to the future end-users in order to get their feedbacks and make possible enrichments / adjustments.

5. CONCLUSIONS

Through the 3 use cases reported in this paper, it is demonstrated that ML applied to helicopter data could be beneficial if used with awareness of the problems / systems and the data. Other aspects of ML, such as reinforcement learning, were not mentioned for lack of finalized use case application. Nevertheless, most frequent uses of ML were well covered by the presented use cases, and conclusions could be drawn from it regarding the mid-term perspectives of data science in general.

Indeed, from one hand, data is becoming an enabler for resolving problems that are more and more complex within helicopter industry. From the other hand, data is continuously increasing, since more helicopters are connected and more flights are performed. Challenges will increase accordingly, for example in terms of data storage and processing scale-up. Nevertheless, mostly opportunities will increase in terms of learning from experience for the continuous improvement for helicopter manufacturers and their customers. Data science will play a more important role to get insights from this variety of data. One key aspect for the success of data science in the industry is first to democratize data and make it of high quality, available and easily accessible for domain experts in order for them to be able to perform simple analysis and explorations. In addition to that, it is important to develop advanced and user friendly tools that mask non necessary algorithmic complexities for domain experts in order to support them in more complex and domain-specific tasks. This way, experts can be more easily involved in the development of data based solutions in order to achieve better results in terms of business values. A very interesting work is proposed in [12], which illustrates how helicopter data combined with expert knowledge and simulation models can improve helicopter landing gear' failure anticipation. In this case simulation completes flight data analysis results to characterise failures modes and accurately estimate the remaining useful life (RUL). Another important aspect is related to decision making support. In fact, often ultimate decisions are made taking into account not only results computed from flight data but also other inputs, some of which are data such as maintenance, supplier, design, logistics or simulation data. It is thus important to put together results computed from helicopter data with all other ones in a kind of Data Lake, in order to ease their joint analysis and support managers taking optimal decisions.

Regarding ML and AI in general, there are two types of applications in the helicopter and aviation domain where they could be applied. They could have applications that are not subject for certification. Such applications can be numerous, especially for continuously improving products of helicopters' manufacturers and the efficiency of their engineers; but also for reducing / eliminating unscheduled events and enhancing safety for customers. In this case we need only to follow the different classical steps for building the targeted ML model. However, for applications that require certification such as maintenance credit, ML and AI algorithms in general should be approved as ordinary software; which is not the case today due mainly to their black-box aspect and the difficulty to explain and interpret the results they produce. This constitutes a limit for AI adoption in aviation in general, which needs to be dealt with. Indeed, traditional ML algorithms such linear regression, decision tree or Bayesian classifiers can be regarded as transparent, explainable and interpretable. However, more powerful ML algorithms such as neural networks and ensemble methods are not. This requires additional steps to be interpreted or provide some learning guaranties. Actually this is an active domain of research [13,14]. Even authorities are now working to establish vision on the safety and ethical dimensions of development of AI in the aviation domain[8]. Working groups on AI in aviation are created, such as the EUROCAE WG114[9] and SAE G34, in order to prepare technical standards, guidelines and other material required to support the development and the certification of aeronautical systems implementing AI technologies. Other initiative such as DEEL project [15] are also working to address the certification of systems embedding ML algorithms. Clearly, this aspect should be given more importance in order to open new opportunities, especially for safety enhancement and condition based maintenance (CBM) for helicopters.

6. REFERENCES

[1] Ammar Mechouche, Nassia Daouayry, V. Camerini, Helicopter Big Data Processing and Predictive Analytics: Feedback & Perspectives. 2019. European Rotorcraft Forum (ERF). Paper n° 35, Poland.

[2] Ammar Mechouche, Adil Soubki, Stanislas Le Marchand, Antoine Fagni, Frederic Champavier. Anomaly Detection in Vibration Sensors Using

[8] https://www.easa.europa.eu/ai

[9] https://www.eurocae.net/news/posts/2019/june/new-working-group-wg-114-artificial-intelligence/

Features Engineering & Machine Learning. 2019. AIRBUS DATA DAYS, Toulouse.

[3] Tianyi LI, Olivier Regnier-Coudert, Jayant Sen Gupta, Rob Vingerhoeds. 2019. Features Generation and Feature Selection of Time Series Data. AIRBUS internal report.

[4] Paul Boniol, Themis Palpanas, Mohammed Meftah, Emmanuel Remy: GraphAn. 2020. Graph-based Subsequence Anomaly Detection. Proc. VLDB Endow. 13(12): 2941-2944.

[5] Jayant Sen-Gupta, Antoine Fagni. AI Gym - Time Series Anomaly Detection challenge. 2019.

[6] Gabriel Rodriguez Garcia, Gabriel Michau, Mélanie Ducoffe, Jayant Sen Gupta, Olga Fink. 2020. Time Series to Images: Monitoring the Condition of Industrial Assets with Deep Learning Image Processing Algorithms. CoRR abs/2005.07031.

[7] Nassia Daouayry, Ammar Mechouche, Pierre-Loic Maisonneuve, Vasile-Marian Scuturici, Jean-Marc Petit. 2019. Data-Centric Helicopter Failure Anticipation: The MGB Oil Pressure VS Case. BigData. 1784-1793.

[8] Brian Tucker, Drew Waller, Ankit Patel, Bell Textron Inc. Bell 525 Relentless — Using Tail Rotor Torque Measurements for Maintenance Credit. 2020. Vertical Flight Society's Annual Forum & Technology.

[9] Liu Zhihua, Yang Jian. Quantifying ecological drivers of ecosystem productivity of the early-successional boreal Larix gmelinii forest. Ecosphere. 2014. 5. art84. 10.1890/ES13-00372.1.

[10] Yurdakul Bilal, Naranjo Joshua. Statistical Properties of the Population Stability Index. 2019. Journal of Risk Model Validation, Vol. 14, No. 4, Available at SSRN: https://ssrn.com/abstract=3783305.

[11] Ramos Juan. Using TF-IDF to Determine Word Relevance in Document Queries. 1999.

[12] Antonin Rocher, Roland Becquet, Jean-Charles Maré. Model-based failure anticipation and predictive maintenance - A landing gear application. 2021. Vertical Flight Society's Annual Forum & Technology.

[13] Melanie Ducoffe, Sébastien Gerchinovitz, Jayant Sen Gupta: A High Probability Safety Guarantee for Shifted Neural Network Surrogates. 2020. SafeAI@AAAI 2020: 74-82.

[14] Adrien Gauffriau, François Malgouyres, Mélanie Ducoffe: Overestimation Learning with Guarantees. 2021. SafeAI@AAAI 2021.

[15] Hervé Delseny, Christophe Gabreau, Adrien Gauffriau, Bernard Beaudouin, Ludovic Ponsolle, Lucian Alecu, Hugues Bonnin, Brice Beltran, Didier Duchel, Jean-Brice Ginestet, Alexandre Hervieu, Ghilaine Martinez, Sylvain Pasquet, Kevin Delmas, Claire Pagetti, Jean-Marc Gabriel, Camille sChapdelaine, Sylvaine Picard, Mathieu Damour, Cyril Cappi, Laurent Gardès, Florence De Grancey, Eric Jenn, Baptiste Lefèvre, Grégory Flandin, Sébastien Gerchinovitz, Franck Mamalet, Alexandre Albore: White Paper Machine Learning in Certified Systems. 2021. CoRR abs/2103.10529.

[16] Ross Taplin, Clive Hunt. 2019. The Population Accuracy Index: A New Measure of Population Stability for Model Monitoring. Risks 7, 53; doi:10.3390/risks7020053.