A Domain Independent Data Mining Methodology for Prognostics

Létourneau, Sylvain; Yang, Chunsheng; Drummond, Chris; Scarlett, Elizabeth; Valdés, Julio; Zaluski, Marvin

Publisher’s version / Version de l'éditeur:

Essential technologies for successful prognostics : proceedings of the 59th Meeting of the Society for Machinery Failure Prevention Technology, Virginia Beach, Virginia, April 18-21, 2005, 2005

A Domain Independent Data Mining Methodology for Prognostics *

Létourneau, S., Yang, C., Drummond, C., Scarlett, E., Valdés, J., and Zaluski, M.
April 2005

* published in Conference Proceedings: Essential Technologies for Successful Prognostics, 59th Meeting of the Machinery Failure Prevention Technology Society. Virginia Beach, Virginia, USA. April 18-21, 2005. NRC 48123.

A DOMAIN INDEPENDENT DATA MINING METHODOLOGY FOR PROGNOSTICS

Sylvain Létourneau   Chunsheng Yang   Chris Drummond
Elizabeth Scarlett   Julio Valdes   Marvin Zaluski

Institute for Information Technology
National Research Council of Canada, Ottawa
{*firstname.lastname*}@nrc-cnrc.gc.ca

Abstract: Modern operation of complex systems such as trains and aircraft generates vast amounts of data. This data can be used to help predict component failures which may lead to considerable savings, reduce the number of delays, increase the overall throughput of the organization, and augment safety. Many data mining algorithms, such as neural networks, decision trees, and support vector machines, exist to learn models from vast amounts of data but their application to real world operational data from systems such as aircraft and trains is very challenging. For successful prognostics, several difficulties need to be carefully addressed including data selection, data fusion, data labeling, model integration, and model evaluation. This paper explains these issues and presents a methodology that we have developed to address them in a systematic manner. The paper discusses the application of the methodology to the rail and aerospace industries and highlights open problems.

Key words: Data mining; aircraft health monitoring; train wheel monitoring; component failure prediction.

1 Introduction

An adequate health management system for complex machinery requires diagnostic, prognostic, and repair management technologies. Although significant research is still going on in all three aspects, prognostic technologies lag behind the other two particularly in the maturity of the technologies and adoption by operators of complex machinery. Over the last few years, we have seen a number of researchers working on improving prognostics by incorporating techniques from Artificial Intelligence and Data Mining. In this paper, we will push this idea much further by proposing the construction of prognostic models using only data mining techniques, applied to historical data, and a minimal amount of engineering knowledge. The result is a domain independent methodology to construct prognostic models. These models are capable of predicting failures of specific components of complex systems such as aircraft and trains.

The goal of our methodology is to generate models to predict when to replace a component. These models must accurately recognize particular data patterns that indicate upcoming component problems. They must also be able to recognize problems within a reasonable period of time prior to the actual occurrence of the failure. For most components, a period of one day to three weeks in advance is appropriate. For components that are very expensive or difficult to obtain, an earlier warning would be preferable.

The proposed methodology is entirely based on data collected during the normal operation of the machinery. There are two kinds of data: maintenance data, describing repair actions carried out by the maintenance staff, and sensor data, recording values acquired from sensors monitoring system parameters. No particular domain knowledge is required. So as long as these two kinds of data are available, the methodology is applicable to any complex system. The information in the maintenance data is fairly consistent across applications. For each maintenance action, we have: the date of the action, the equipment identifier, the identifying numbers for parts installed and removed, the employee name, and typically a textual description of the work performed. The characteristics of the sensor data depend on the machinery considered. For instance, an Airbus A320 generates up to 19 types of sensor measurement reports corresponding to different stages of operation (e.g., engine start, takeoff, cruise) and each one contains between 20 and 150 data items (numeric and symbolic). On the other hand, the WILD (Wheel Impact Load Detector) system, which monitors the impact of train wheels on the rails, collects only a few measurements (e.g.; dynamic impact, train speed, train direction) for each wheel of the train.

There are two reasons to replace a component: as part of regular maintenance (imposed by regulations or operator policy) or because the part's condition has deteriorated and might fail. Only the latter is relevant for predictive models, as maintenance staff are already aware of regular maintenance requirements. Of course, a system cannot predict replacement for components without relevant data. Since the data available in our projects are related to aircraft engines and train wheels, our focus is on models that predict problems with components from these systems. It is also important to notice that we can only address components for which we have already observed a minimal number of failures.

Our methodology does not attempt to capture all possible failures. In particular, we cannot identify component failures due to maintenance or design problems. We do not have sufficient data for such problems; fortunately, they represent a small percentage of all component failures. The rest of the paper begins by describing the methodology, it then discussed two case studies, and then goes on to highlight open problems. Létourneau et al. [5] offers an alternative description of the methodology which focuses on predicting aircraft component replacement.

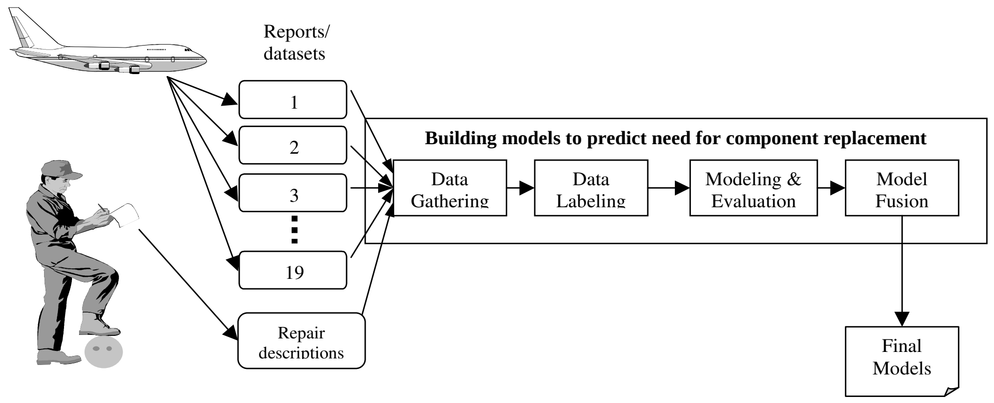

Figure 1: The 4 steps process to build models for component replacements.

2 The Methodology

Other research has used data mining to infer models from historical data. Our methodology builds on this by addressing issues not identified in earlier work. As illustrated in Fig 1, we stress four steps of particular importance in this type of application. They are data gathering, data labeling, model evaluation, and model fusion. The following sections explore these issues in turn and propose corresponding solutions.

2.1 Data Gathering

Most data mining algorithms require as input a data set containing examples consisting of vectors of attribute values. Modern machinery often generates many such data sets. The Airbus A320 generates up to 19 different data sets reporting the status of the aircraft in different phases of operation. The number of examples in each data set varies considerably. Our first problem is to select the data set(s) to use to build models for a particular component. Domain experts, if available, may select the most suitable data set(s) for the component of interest. Otherwise, we rely on data set descriptions and include all data set(s) that appears to be somehow related as to limit the risk of disregarding potentially useful information.

Not only must we select a data set, we must also select a subset of instances to use in the analysis. The data sets are typically so large, it is inefficient to build models using all instances. Simple solutions, such as random sampling, are also inappropriate. To build the desired predictive models, we must be much more focused. We base our analysis on data generated around each occurrence of component replacement. We first identify when replacement occurred and then retrieve instances around the time of this occurrence.

Retrieving replacement cases:

The maintenance databases contain reports on the repair and replacement of many different components. We might hope a simple query, matching all reports with the desired part number, would retrieve all relevant replacement cases. Unfortunately, retrieving occurences of failures may be turned out to be quite difficult. The difficulties come from errors in maintenance reports and the need to process free form texts.

We see many kinds of problems with maintenance reports. Some reports mention a component although it didn't fail. Some reports mention a component that failed but not in a way that we are trying to predict. Sometimes the part number entered is incorrect. Sometimes alternative part numbers used. Sometimes the right information is entered but in the wrong field. Sometimes the right part number is used but the wrong component is identified, there are often multiple components of the same type within the machinery. If a person studies the report it is usually possible to determine what has been replaced and why. But because the maintenance databases are typically very large, manual analysis is not possible. Since the types of error differ between applications, customized solutions are required. Section 3 presents the two solutions we have developed to retrieve failure information for train wheels and aircraft components.

Selecting instances: Once we have the date and the part identifier for each component replacement, we retrieve the relevant instances from the selected sensor data sets. For each data set and replacement, we retrieve the data obtained m days prior to the replacement and n days after. The numbers m and n depend on the data sets and the component; we usually select m so that we have at least 200 instances available for learning. For example, when we have an average of two reports per machine per day, m will be set to 100 or above. For n, we simply set it as $n = 0.15 * m$. The selected values for m and n influence how we set the k parameter (defined in Section 2.2).

Finally, we add two new attributes to the initial sensor data: the time between sensor measurement and the actual replacement ('time from failure'), and a tag associating each instance with a specific replacement case ('problem ID'). We combine all instances from a selected data set to create the data set that we'll use to build the predictive model. The number of output data sets is equal to the number of data sets selected as potentially useful for the target component.

2.2 Data Labeling

Predicting the need for component replacement can be viewed as a classification task with two classes: replace component or don't replace component. The learned model will classify each new report as one of these two classes. Supervised learning cannot be directly applied to the sensor data, however, because it does not contain a class attribute. Our solution is automatic data labeling: we add a preprocessing step that computes the class value of all instances used in the analysis.

We set the class to 1 for all instances in a target window, defined as k days before component replacement. We set the class attribute to 0 for all instances outside the window. Obviously, k has to be smaller than m. We have used days to define the window rather than usage information (e.g., mileage, hours of operation, or cycles - pairs of takeoffs and landings) mainly because such information is generally not provided with the sensor data. In some applications, users have specifically asked for predictions in days. From a data analysis viewpoint, this choice is not ideal as repairs are more related to usage than to time. However, we experimented with cycles in an aerospace application and the results were quite similar to those using days. This is probably because the commercial aircraft considered in this study all had similar usage patterns.

For each component, we perform several experiments to find the best value for k. When designing the experiments, we consider the following factors:

- The target window for predicting component replacement. Different components will have different sized windows.
- The proportion of positive and negative instances. Without a significant proportion of positive instances, data mining approaches often have difficulties with model learning. To ensure that at least 10% of the examples are positive, we use the constraint $k > 0.1 * (m + n)$
- Pattern characteristics. To facilitate the learning of the models, we try to synchronize labeling with qualitative changes we observe in the data.

2.3 Modeling and Evaluating Models

For building models, we use standard data mining algorithms but we propose a novel method to evaluate models since current ones are not adequate for this kind of application. We build models with techniques such as decision trees, nearest neighbor, naive Bayes, rough sets, regression, and neural networks. Depending on the algorithm, extra preprocessing steps may be needed: selecting the most suitable subset of attributes, normalizing the attribute values, creating new features from the original ones, and discretizing continuous attributes. Techniques to perform these tasks can be drawn from different areas of research such as machine learning, statistics, pattern recognition, and data mining.

In practical applications such as ours, results must be properly evaluated. The evaluations must fairly estimate the model's performance on future data and account for domain specific criteria. There are several approaches for computing the expected performance, including hold-out validation, cross-validation, and bootstrapping. Unfortunately, none of these approaches is suitable for our application as they rely on random sampling to select instances from the population. Random sampling assumes that the instances are independent. In our application, the assumption of instance independence is simply not viable. Because of variations in system construction, operation, and maintenance, any two instances from a given system have a greater dependence than any two instances from different systems. If we were to use data from the same system for training and for validation, success would be much more likely than if we used data from a different system for validation.

Similar reasoning applies to component replacement data: two instances from the same problem will be more related than two instances from different problems. Given that the number of component replacements is significantly less than the total number of instances, random sampling would likely generate training and validation sets containing data from the same problem. The end result would be an overly optimistic evaluation of the model's performance. We solve this by splitting the data so that training and validation instances come from different subgroups of component replacement events. This is achieved by using the 'Problem ID' attribute added during data gathering. We obtain a more robust estimate by adapting the cross-validation method to account for the subgroups as follows:

1. Split the data into batches using the 'Problem ID' (one for each failure case).
2. Keep one batch for validation and use all others for training. Repeat this step so that each batch is used once for testing.
3. Average the validation results from each of the batches.

Evaluation function: At the heart of our modified cross-validation method is a measure of classifier performance. In machine learning research, performance is often summarized by either error-rate or accuracy. The error-rate is defined as the expected probability of misclassification: the number of classification errors over the total number of test instances. The accuracy is 1 minus the error-rate.

Because some errors can be more costly than others, it's often desirable to minimize the misclassification cost rather than the error-rate. The ROC method[7] is very popular in practical applications since it allows the user to evaluate models independent of the particular class frequency. Other possible metrics are precision and recall commonly used in the information-retrieval community[6]. Recall is usually defined as the ratio of relevant documents retrieved for a given query over the number of relevant database documents; precision is the ratio of relevant documents retrieved over the total number of documents retrieved.

Unfortunately, these metrics fail to capture two important aspects of our application. The first aspect is that the usefulness of a prediction is function of the time between the prediction and the actual replacement. Warning too early about a potential failure leads to non-optimal component use; warning too late makes proper repair planning difficult. We need an evaluation method that takes alert timeliness into account. The second aspect relates to coverage of potential failures. Because the learned model classifies each report into one of two categories (replace component; don't replace component), a model might generate several alerts before the component is actually replaced. More alerts suggests a higher confidence in the prediction. However, we clearly prefer a model that generates at least one alert for most component failures over one that generates many alerts for just a few failures. That is, the model's coverage is very important to minimizing unexpected failures. Given this, we need an overall scoring metric that considers alert distribution over the various failure cases. The next section presents a reward function to take into account first aspect (timeliness of the alerts) while the following section introduces a new scoring metric that addresses the second aspect (coverage of failures).

Reward function: In our methodology, we define a reward function for predicting the correct instance outcome. For each target component, the reward for predicting a positive

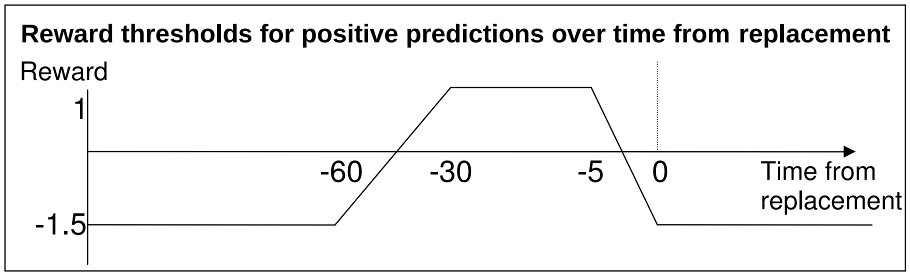

Figure 2: Example of a reward function for prediction of positive instances.

instance is based on the number of days between instance generation and the actual replacement. Figure 2 shows a graph of this function. The maximum gain is obtained when the model predicts the need for replacement five to thirty days prior to the component's replacement. Outside this target period, predicting a need for replacement can lead to a negative reward threshold, as such a prediction corresponds to misleading advice. Accordingly, false-positive predictions (predictions of a failure when there is no failure) are penalized by a reward of -1.5 in comparison to a 1.0 reward for true-positive predictions (predictions of failure when there is a failure).

Although the reward function's target period is not necessarily the same as the labeling period, the two are related. In general, we determine the reward function's target period according to the user requirements, and then fix the labeling period. For example, if the user says that predictions for a given component should be within one to three weeks in advance, we set the reward function's target period for between -7 to -21 days. We then experiment with various labeling periods around this target period and use the one that offers the best result.

The reward function in Figure 2 follows a piecewise, linear model. Such a model is convenient because it is comprehensible to the maintenance staff and is sufficiently complex to capture the relevant information. There are several ways to improve the reward function's precision. One possibility is to use higher-order polynomials instead of straight lines. Another possibility is to try to smooth the overall function. However, according to domain experts, increased complexity is rarely needed.

Scoring metric: Our reward function accounts for alert timeliness; to evaluate model coverage we must look at alert distribution over the different failure cases. The overall performance metric we propose to evaluate a model is:

$$score = (\sum_{i=1}^{p} score_i) * (NbrDetected/NbrOfCases)^{SignOfSumOfScores} \qquad (1)$$

where p is the number of positive predictions made by the model on validation instances during application of the evaluation method, $score_i$ is the score from the reward function for the i^{th} instance classified as positive, $NbrDectected$ is the number of replacement cases for which we have at least 1 positive prediction in the target interval (e.g., -30 to -5 days), $NbrOfCases$ is the total number of replacement occurrences we have for the given component, and $SignOfSumOfScores$ is the sign of the first term in the expression. The term

$(NbrDetected/NbrOfCases)^{SignOfSumOfScores}$ is introduced in the evaluation function to favor models that optimize the recall (The term $(NbrDetected/NbrOfCases)^{SignOfSumOfScores}$ is set to zero when $NbrDetected = 0$ and $SignOfSumOfScores$ is negative). The value of $\sum_{i=1}^{p} score_i$ makes the final score sensitive to precision of the model.

2.4 Model Fusion

We use model fusion for two reasons. First, model fusion can be used to combine models derived from different data sets. When more than one data set is relevant for a given component then we build models independently for each data set and use model fusion to combine predictions from the various models. Second, we can apply model fusion to help improve performance even if there is a single data set. We can learn various models using various techniques or parameter settings and combine them to obtain better performance than using any single model.

Bagging and boosting are two popular techniques to combine models but they are only applicable when there is a single data set and one kind of model (a single learning algorithm). For heterogeneous models or multiple data sets, we apply methods based on voting or stacking strategy. As we explain in the discussion of the WILDMiner project (next section), we successfully applied meta-learning as a way to combine heterogeneous models.

3 Case Studies

In the following we discuss our experience in applying the proposed methodology to two maintenance applications for the railways and for the airlines. Our description focuses on data gathering since customized solutions where required. In the case of the train wheel failure predictions, we also elaborate on model fusion because of its positive impact on results. All other steps were applied as described above.

3.1 The WILDMiner project

Wheel failures on trains represent a significant proportion of the operational cost for railways. For instance, Canadian railroads spend about $65M/year to repair and replace freight car wheels. Moreover, when not detected and fixed in a timely fashion, wheel failures may cause rail breakage and even train derailment. To avoid such events, many railroads have invested in the WILD technology. In Canada alone, there are more than twenty WILD systems installed at strategic locations on main line tracks.

The WILD systems measure the vertical force of each passing wheel over the site. Problem wheels typically produce higher WILD measurements due to an increase in the impact on the rail. The measurements are sent in real-time to a central site where fleet supervisors decide whether or not to replace/repair a given wheel. Depending on the actual WILD measurement, recommended actions range from scheduling a wheel inspection to stopping the train until the wheel is repaired. Often, the train is allowed to continue at a reduced speed to the nearest shop or siding where the offending car is uncoupled from the train. The current strategy is reactive and leads to many changes in the route or speed of the train during operation. This is problematic because it introduces delays, disrupts the overall schedule, degrades customer service, and reduces the railway's throughput capacity.

To minimize the actual number of disruptions during operation, we need a pro-active maintenance strategy based on a system that monitors the data in real-time and identifies the deteriorating wheels before they force speed reduction or train stoppage. This strategy requires models to predict wheel failures, which we are developing using the methodology introduced above.

We retrieve information about past failures from the maintenance data and then gather relevant sensor data. In this application, the maintenance database is relatively clean because the staff use a menu driven system to enter most of the information. The options selected through the menus are translated into codes in the maintenance database. To retrieve occurrences of wheel replacements, we simply search for specific codes. The only difficulty is in identifying the exact location of the wheel that has been replaced. The cars we worked on have 12 axles for a total of 24 wheels. To identify a particular wheel location, the maintenance staff enters in free form text the number of the axle (1 to 12) and the side of the faulty wheel (right or left). Unfortunately, we found many inconsistencies and errors in this text. For instance, technicians use many symbols to refer to the axle number 10 such as: 'a', '10', 'x', '0x'. The same problem also exists for other axles above 10. Moreover, we found that the staff sometime enters the wrong axle number (e.g., 6 or 8 instead of 7) or wrong side (left instead of right). These errors can negatively influence the modeling process by making it construct failure models that fit non-failing wheel data. We try to deal with errors in wheel location through extensive manual validation. When the ambiguity is too high, we generally ignore the case and ensure that potentially related sensor data is not included in the modeling process.

The sensor data for this application is as follow. Each time a train goes over a WILD system, sensors installed along the tracks measure the dynamic impact of each wheel on the rail. The system then sends these measurements to a central system along with contextual information such as the ID of the car, the speed of the train, and its direction. These information are then stored in a single table. We create the data set to build the predictive models by extracting from this potentially huge table all the data around occurrences of wheel failure replacements. We did most of our experiments with $m = 100$ and $n = 15$.

To build the models, we experimented with various learning and feature extraction techniques. The scores obtained from various learning algorithms were all around 290 (for the dataset considered, score values vary from -9500 to 1200) with a decent coverage but also a relatively high rate of false alerts. We applied meta-learning to perform model fusion. This process involves learning a meta-model that predicts failures based on predictions from a number of low-level models directly relying on the sensor data. We obtained our best meta-model using the J48 algorithm implemented in WEKA[8]. In addition to a global score of 650, this model has both a good coverage and very low false positive errors. The four heterogeneous low-level models used in this experiment were learned with the J48 and

Naive Bayes classifiers (two versions for each classifier). We consider that this model could be deployed with significant financial benefits. However, as we discussed in Section 4, a number of issues could be addressed to further improve the performance.

3.2 The ADAM Project

Our methodology was developed for ADAM[1], a collaborative project with a Canadian airline, and then generalized to other domains. The aim of this project is to develop a monitoring system that predicts aircraft component failures and identifies abnormal system behavior. The models are learned from operational and maintenance data for more than 75 aircraft: about half of these are Airbus A320 and the other half are Airbus A319. For the A320, we have accumulated more than 10 years of historical data.

The data gathering stage of the methodology was particularly difficult for this project. Most of the data stored in the maintenance database was input by hand resulting in great deal of variability in the quality of the reports. Often the field for the part-removed identifier is empty or contains erroneous information. The correct part identifier may be written in the textual description instead of the part-removed field. In fact, the free text portion of the reports were the often most reliable indicators of a part being replaced.

Therefore, we must process the textual descriptions to find all occurrences of component replacement. Automatically processing such descriptions is very difficult, as the maintenance staff uses informal language, which often includes abbreviations and acronyms, major grammatical and syntactical problems, typing mistakes, and inconsistent use of abbreviations among reports[3]. We extract key phrases by analyzing the textual descriptions retrieved. We tried to automate the keyword extraction process using a keyword-generation system but we later realized that it is often more efficient to perform this step manually.

We use an information retrieval based approach to solve this problem, called query reformulation. We start with the part number in part-removed field as the basic query. We then explore the reports that match this query and identify new values in different fields that seem associated with this part being replaced. Text fields are often the source of these new values. For example, if we find the key phrases ‘starter motor’, ‘starter’, and ‘49400126’, then the new query will search for any maintenance report that contains the part identifier in the appropriate field or any of the key phrases in the textual descriptions. We typically find new terms in these new reports and they are added to the query. New terms may also be added to make existing terms more specific and thus reduce the number of records they match. We continue reformulating the query until we have most of the relevant reports returned. To make sure of this we over-generalize the query and then manually filter reports that do not deal with part replacement.

Our approach provides no guarantee that all related reports will be selected. Its success depends on the completeness of the final queries we use to retrieve the reports. Specialists may be able to estimate the number of failures for the component of interest. In such case, we can evaluate our coverage by comparing specialists’ estimate to the number of replacements found. Although this process still requires much human intervention, the number of reports that require manual validation is very small in comparison with the overall number in the database. So our approach can be viewed as reducing manual analysis by automatically preselecting potentially relevant reports. We have also looked at ways to semi-automate this process to further reduce the work required by the human [2].

In Létourneau et al. [5], we report results from a fully automated large scale experiment involving components from the APU (auxiliary power unit) and the main engines. Since then, we performed extensive evaluations and studies focusing on components of interest to the airline involved in the project. The results are highly positive on a few components such as the APU starter motors. For instance, the models developed for the A320 APU starter can predict around 95% of failures between 1 to 21 days in advance without producing false alerts. The models for the A319 APU starter are also very good but with a small ratio of false alerts, which we explain by the lack of data for that fleet. For other components such as the various valves used to fine tune engine performance, we have been unable to build accurate predictive models. The next section elaborates on issues that we believe need to be addressed to overcome current difficulties.

4 Open Issues

The methodology presented in this paper is based on the steps needed to better apply data mining to predicting failures in complex equipment but many open problems remain. This section discusses some of these open problems and links them to the methodology and applications discussed above.

Improvements to quality of maintenance data: To gather the right data for the analysis, we need to be able to identify all relevant failure events. Retrieving these events is quite difficult in practice due to data consistency and reliability issues. This could be improved by enhancing the tools available for data entry. For instance, in our aerospace application, the introduction of tools to validate the information on parts removed and installed would be of great help. Additionally, personal should understand that our methodology is dependent on the quality of the maintenance practices. If components are replaced without good reasons, the models are likely to recommend replacement of healthy components. Making maintenance personnel aware of the detrimental effects of inaccurate maintenance data on prognostics and health management should help motivate them to increase data quality.

Improvements to quality of sensor data: Most of the sensors in current commercial aircraft have been installed for diagnostic purposes. Prognostics may require different or additional information. An issue that needs to be accounted for in the design of future complex equipment. Current projects like the Joint Strike Fighter (JSF) which clearly includes prognostics and health management as a core functionality are particularly encouraging in that respect. In addition to integrating the right sensors, we also need to ensure their reliability over the life of the equipment. In practice, we observe that organizations invest considerably in data collection and storing technologies but they rarely implement mechanisms to ensure that the data they get is reliable. For instance, in the WILDMiner project we have seen WILD systems that appear highly uncalibrated. Similarly, some of the aircraft sensor values we get are questionable. A complete prognostics and health management system should definitely include data validation techniques.

Data normalization to control variance: Inevitably, when we mix sensor data from complex equipments operating in heterogeneous environments we obtain very high variances. Such deviations may prevent learning techniques from discovering patterns of interest in the data. This is particularly true when there is limited data (few examples of failures) or when a failing component has only marginal effects on sensor measurements. We believe that this explains why we have difficulties building accurate predictive models for some aircraft engine components.

To resolve this problem, we need to integrate a number of complementary techniques. Domain engineering models can help to account for part of the variance associated with equipment characteristics and operational conditions. Configuration management systems can help keeping track of physical differences between individuals in a given fleet. Finally, techniques for contextual normalization [4] and data reduction may be used to further reduce the variance in the data that cannot be explained by component failures.

Methods to perform cost/benefit analysis: The evaluation method presented in this paper accounts for important technical constraints such as failure coverage and alert timeliness. However, this is not sufficient to assess the potential business value of models to predict failures of a given component. Ideally, we would like an extension of the current method that takes into accounts the various costs (undetected failures, early replacement, false alerts, implementation of the models) and returns the expected gain in dollars. However, this may not be practical as organizations may be unable to adequately evaluate these costs. We should therefore also investigate other alternatives.

5 Conclusion

This paper generalizes a methodology we have developed to build predictive models for complex equipment using data mining and historical data. The methodology focuses on the use of classification techniques to build the predictive models. We describe how to select relevant historical data and a method for automatic labeling. The methodology includes a novel method to evaluate models. After discussing the application of the methodology to predict train wheel failures and aircraft component failures, the paper describes a number of open problems. The case studies demonstrate the applicability of the proposed methodology to various domains. In particular, we observe that most of the steps are applied without any customization. Our current work in this area tries to address the open problems discussed.

References

[1] ADAM project at the National Research Council of Canada. Website. `http://iit-iti.nrc-cnrc.gc.ca/projects-projets/adam_e.html`.

[2] Drummond, C. (2004). Iterative semi-supervised learning: Helping the user to find the right records. In *Proceedings of the International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems*, (pp. 1239–1240).

[3] Farley, B. (1999). From free-text repair action messages to automated case generation. In *Proceedings of the AAAI 1999 Spring Symposium on AI in Equipment Maintenance and Support*, (pp. 109–118)., Stanford University, Stanford, California, U.S.A.

[4] Létourneau, S., Famili, A. F., & Matwin, S. (1998). A normalization method for contextual data: Experience from a large-scale application. In *Proceedings of the 10th European Conference on Machine Learning (ECML-98)*, (pp. 49–54).

[5] Létourneau, S., Famili, A. F., & Matwin, S. (1999). Data mining for prediction of aircraft component replacement. *IEEE Intelligent Systems and their Applications*, *14*(6), 59–66.

[6] Makhoul, J., Kubala, F., Schwartz, R., & Weischedel, R. Performance measures for information extraction. In *Proceedings of DARPA Broadcast News Workshop, (Herndon, VA), February 1999.*

[7] Provost, F. & Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In Heckerman, D., Mannila, H., Pregibon, D., & Uthurusamy, R. (Eds.), *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, (pp. 43–48). AAAI Press.

[8] Witten, I. H. & Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann.