Data Mining For Prediction of Aircraft Component Replacement

Sylvain Létourneau[1] Fazel Famili[1] Stan Matwin[2]

[1]Institute for Information Technology
National Research Council of Canada, Ottawa
{sylvain.letourneau, fazel.famili}@iit.nrc.ca

[2]School of Information Technology and Engineering
University of Ottawa, Canada
stan@site.uottawa.ca

August 3, 1999

Abstract

The operation and maintenance of modern sensor-equipped systems such as passenger aircraft generate vast amounts of numerical and symbolic data. Learning models from this data to predict problems with component may lead to considerable savings, reducing the number of delays, and increasing the overall level of safety. Several data mining techniques exist to learn models from vast amounts of data. However, the use of these techniques to infer the desired models from the data obtained during the operation and maintenance of aircraft is extremely challenging. Difficulties that need to be addressed include: data gathering, data labeling, data and model integration, and model evaluation. This paper presents an approach that addresses these issues. We also report results from the application of this approach to build models that predict problems for a variety of aircraft components.

Keywords: Data mining, machine learning, aircraft health monitoring, component failure prediction.

1 Introduction

The operation and maintenance of modern sensor-equipped systems such as aircraft generates vast amounts of numerical and symbolic data. The data is generated by thousands of sensors installed in various components of the aircraft; it is then sent in real-time to ground stations and then stored in relational databases. Before being transmitted to the ground, a number of on-board computer systems monitor and analyze the data in order to make sure that various systems of the aircraft operate properly. However, once the data is stored in central databases, further data analysis is rarely performed. This paper presents an approach that makes use of this data in order to develop models to predict the need for replacement of various aircraft components before they become non-operational. The end goal is to implement these models in a flight data monitoring system that will receive as input (in real-time) the data from a fleet of commercial aircraft, will analyze it, and as output it will alert appropriate staff when there is a need for a component replacement. The monitoring system will use the automatically sensor data from the aircraft, and a predictive model described in this paper, to detect component problems and recommend their replacement. Such a system could help improving the airline's operation by: reducing the number of delays, reducing the maintenance costs, helping to obtain better maintenance planning, and increasing the level of safety.

The approach proposed in this paper applies techniques from the fields of Machine Learning and Data Mining on huge amounts of complex historical data in order to develop the predictive models required by the monitoring system. The approach described addresses four fundamental difficulties with existing data mining approaches: automatic selection of relevant data, automatic labeling of instances, evaluation method that accounts for dependencies between the instances, and a scoring function measuring the extent to which the results fit the domain requirements. By addressing these four issues, we believe that the proposed approach will help extending the range of potential applications for data mining techniques. Examples of other applications that can benefit from the approach developed in this paper are: prediction of problems in complex systems (e.g.: trucks, ships, trains, and cars), prediction of problems with complex industrial equipment for which a lot of data is continuously acquired, and prediction of critical events in medical applications (e.g. Emergency Room care). The fact that the proposed approach relies on a minimal amount of domain specific information will also facilitate the adaptation to other applications.

2 The application and the data used

The aim of the approach described in this paper is to generate a valid set of models to predict the need for replacement of an aircraft component. These models will have to accurately recognize particular patterns in the data that indicate upcoming problems with a component. Our approach makes use of data mining techniques to infer these models from the available data. Ideally, the models developed will be able to recognize problems within a reasonable period of time prior to the actual occurrence of the problem. For most components, a period of 1-3 weeks in advance is appropriate. For components that are very expensive or difficult to obtain, it may be preferable to receive alerts even before 3 weeks in order to allow enough time for proper actions.

There are two reasons to replace an aircraft component: either the component has to be replaced as part of regular maintenance (imposed by aerospace regulations or airline's policy) or it is in a deteriorated condition and needs to be replaced before it fails. Only the second type of replacement requires the development of predictive models because maintenance staff already know about regular maintenance requirements. It is also important to remember that no accurate predictions can be expected for components for which relevant data is not available. For instance, there is usually no data related to the quality of passenger seats, therefore one will not be able to predict the need for replacement of passenger seats. Since the majority of the data available for this project is related to the engines of the aircraft, we focus on models to predict problems with engine's components.

The approach presented in this paper does not attempt to capture all phenomena that may lead to a component failure. Two types of component failures that our approach is very likely to miss are: failure because of a problematic maintenance action, and failure due to a design problem with a specific component. Currently, we do not have sufficient data about the maintenance actions and design of the components to identify such phenomena. Fortunately, these problems do not represent a very high percentage of all component problems.

More than 3 years of data from a fleet of 34 Airbus A320 is available for this project and new data is continuously acquired. Each Airbus A320 generates around 1MB of data per month. The data consists of two major parts: (i) textual descriptions of all repair actions taken on these aircraft, and (ii) all parametric data acquired from sensors during the operation of the aircraft. The parametric data is obtained in the form of reports at different stages of operation. For example, during takeoff a report which consists of about 100 measurements is generated, in stable cruising conditions another report is generated with a different set of measurements. etc. An Airbus A320 generates up to 19 types of reports. These are generated at different frequencies and contain between 20 and 150 parameters (numeric and symbolic).

3 Application challenges

The idea of using data mining techniques to infer models from historical data is not new and has already been applied in other applications. On the other hand, our approach contributes to the domain by addressing a number of issues that are not taken into account in these classical applications of data mining. The first two issues are related to data preparation requirements prior to the application of data mining techniques, the third issue involves adequate evaluation of the results for the application considered, and finally, the fourth issue is about the fusion of results obtained during the analysis.

Data Gathering Most Data mining techniques require as input one dataset containing a set of examples described by a vector of attribute values. Modern aircraft such as Airbus A320 do not generate one, but up to 19 different datasets reporting the status of the aircraft in different phases of operation. The number of examples in each dataset varies considerably from one dataset to another. Given a component of interest, the first problem is to decide on which dataset(s) to use to develop the predictive models.

Once a dataset has been chosen, we must select the subset of instances to be included in the analysis. Considering the number of instances available in each dataset, it will be very inefficient to try to learn the models using all instances. Simple solutions such as random sampling are not appropriate either. In order to build the desired predictive models, we need to focus the analysis on the data generated around each occurrence of component replacement. The approach to retrieve this data is explained in Section 4.

Data Labeling Data mining approaches are typically classified in two categories: supervised and unsupervised [1]. The two approaches differ in the data they require as input and the type of tasks they can address. For instance, a supervised learning approach will require each instance to be defined by a number of attributes along with its class membership. Given this data as input, the supervised algorithm will try to learn models to predict the class of each instance using the other attributes. Supervised learning approaches are useful in both classification and forecasting tasks. On the other hand, unsupervised learning approach will take as input as set of examples without any class information and will try to find groups of similar instances.

Since there exists a variety of robust classification techniques, predicting the need for replacement of a component could be cast as a classification task with two class values: (i) there is a need to replace the component or (ii) there is no need for replacement. The model learned from the data will have to classify each new report received from the aircraft in one of these two classes. Supervised approaches are appropriate for this kind of tasks. However, the data that we receive from the aircraft cannot be directly used by a supervised learning approach because it does not contain the class attribute. The solution is to have a pre-processing step that automatically computes the membership value of all examples that will be used during the analysis. This process is called *Data Labeling* and it is described in Section 5.

Model Evaluation Proper evaluation of the results is a key factor in a practical application such as the one described here. The evaluation approach must respect two criteria: (i) provide a fair estimate of the performance of the model when applied to new data, and (ii) take into account important domain specific requirements. In Section 6, we present an approach to adequately come up with a good estimate for the performance of a model. This approach extends the well known cross-validation technique. We also discuss an evaluation function that accounts for important application constrains. This new evaluation function differs considerably from traditional evaluation criteria such as accuracy, and recall/precision.

Exploiting multiple sources of information Data mining algorithms are designed to take as input one dataset in a given format. As discussed earlier, among all datasets generated by the aircraft, more than one could be potentially useful to predict the need for replacement of the component of interest. The question is how could we efficiently combine the information from the different datasets that are all relevant. Two basic approaches exist: (i) merge the datasets into a new dataset during a pre-processing step and then we build the models from this new dataset, or (ii) build independent predictive models using different datasets and then we combine the output of these models to get the final prediction. The former approach is difficult to implement for a number of reasons. First, there is no obvious merging strategy because different datasets may contain a very different number of examples. Moreover, by merging datasets which already have a significant number of attributes we are likely to end up with a very complex dataset. This will contribute to make learning of the models even more difficult. Our experiments have been focusing on the second approach which turned out to be easier to implement while helping to improve the results. Models are developed independently from the different datasets, then we combine the output of these initial models to get the final prediction. In this paper, we do not fully address this fourth issue (merging of results) because we are still experimenting with different strategies to combine the models.

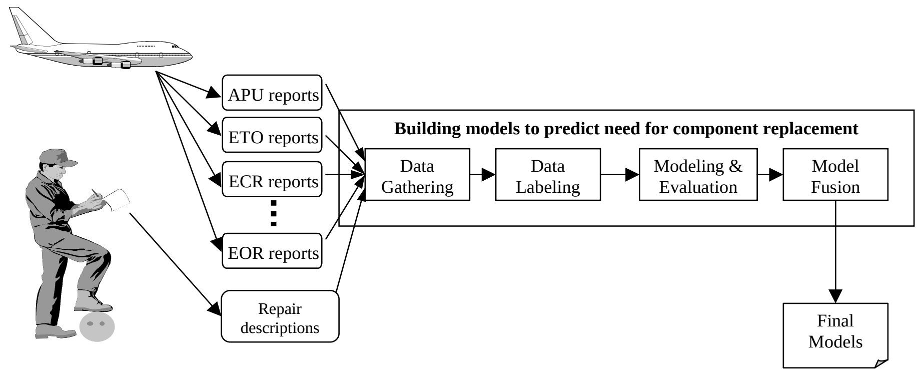

Figure 1: The 4 steps process to build models for component replacements.

It is important to note that the first three issues must be addressed in order to apply data mining techniques to predict the need for aircraft component replacement. On the other hand, the fourth issue is only related to optimization and as we argue in Section 6, promising results can be obtained without fully addressing it. Figure 1 presents the main steps of the approach developed to build models for predicting the need for component replacement. The first three processes are further explained in the following sections.

4 Data Gathering

There are two tasks to perform during this step. The first one is to decide which datasets should be used during the analysis. We rely on the descriptions of the datasets and advices from domain experts to determine which dataset(s) may be relevant to the component under study.

The second task consists of retrieving all relevant instances from the selected dataset(s). Figure 2 illustrates the process followed to retrieve these instances. The first and the most difficult challenge of the Data Gathering step is to retrieve the required information about all occurrences of replacement of a given component. The information is retrieved from a database which contains descriptions of all maintenance actions performed on the aircraft. These descriptions are provided by the engineers and technicians who perform the actions. Examples of information used to describe the maintenance actions are: the date on which the action has been performed, the identifier of the aircraft, the identifier(s) for part(s)removed and/or installed, and textual explanation of the problem and work carried out. Ideally, a simple query returning all maintenance reports for which the `Part Removed` identifier is equal to the identifier for the component of interest will be enough to return all the desired information about all replacement occurrences.

Unfortunately, this is not possible because the field for part removed identifier is often not filled out properly. For instance, in many cases a component has been replaced but the part identifier is put within the textual explanation instead of being given in the field for part removed identifier. This implies that the textual explanation of the repairs also need to be processed in order to find all occurrences of replacement of a given component. Automatic processing of the textual explanations is very difficult because of the open language used by the maintenance staff. There are many abbreviations and acronyms, major grammatical and syntactical problems (e.g. no delimiters between prepositions, absence of verbs, absence of articles and complements), typing mistakes, and inconsistent use of abbreviations among reports. Manual analysis of the texts is not a viable solution either, there are simply too many reports that would need to be analyzed (more than 65000).

The approach we adopt to address this problem is represented by the first three processes in Figure 2. This approach is built around the Information Retrieval paradigm. First, we query the maintenance database in order to retrieve the information about all replacements for which the part removed identifier is equal to the identifier of the part of interest. Second, we use a tuned version of a keyword generation system called Extractor [2] to extract the key phrases from the textual explanations for these replacements. These key phrases are then used to extend the initial query which was uniquely based on the part identifier. The reasoning behind this approach is as follow: if the key phrases obtained correspond to the vocabulary used by the maintenance staff to talk about the component of interest, then these key phrases are very likely to help in finding other replacement occurrences for which the field for part removed identifier has not been filled. For instance, if the key phrases found by Extractor are “STARTER MOTOR”, “STARTER”, and “49400126” then the new query will try to find any maintenance report that contains the given part identifier in the appropriate field or any of the these three key phrases in the textual descriptions. As shown in Figure 2, this will probably add a number of new maintenance reports to the initial ones obtained from the first query. These new reports may talk about the component of interest but without necessarily referring to its replacement. A manual validation of the new maintenance reports is required in order to take out the reports not describing a replacement of the given component. Note that the number of reports that require manual validation is very small in comparison with the overall number of maintenance reports available in the database. In that sense, the proposed approach helps in reducing the amount of manual analysis by automatically pre-selecting potentially related reports. On the other hand, there is no guarantee that all related reports will be selected by the proposed mechanism. This depends on the completeness of the set of key phrases outputs by the Information Retrieval system. To evaluate the coverage of the proposed method, we asked a domain specialist to comment on the number of replacements found for 5 randomly selected components. In all cases, the expert concluded that the number of replacements found with our approach is very close to the exact number of replacements. In other words, for these 5 components, all or almost all occurrences of replacement have been retrieved by the proposed approach.

Once we have the date and aircraft identifier for each replacement of the component of interest, we can retrieve the relevant instances from the selected dataset(s). We consider relevant all instances obtained around the time of the replacement. In particular, for each selected dataset and for each occurrence of replacement, we retrieve the data obtained during m days prior to the replacement and n days after the replacement. The numbers m and n depend on the datasets and the component of interest. Our strategy is to select m so that we have at least 200 instances available for learning the patterns. For instance, with a dataset for which we have an average of two reports per aircraft per day, m will be set to be at least 100. For n, we simply set it as $n = .15 * m$. As we discuss in Section 5 (Data Labeling), the selected values for m and n influenced the setting of the k parameter.

In our approach, we use days as time metric instead of cycles (pairs of takeoffs and landings) or hours of operation mainly because the maintenance staff which are the target users prefer to obtain predictions about the need for replacement in days. From a data analysis point of view, this choice may not seem appropriate because one would think that repairs are more related to the use of a plane than to time. We verified this hypothesis through experiments

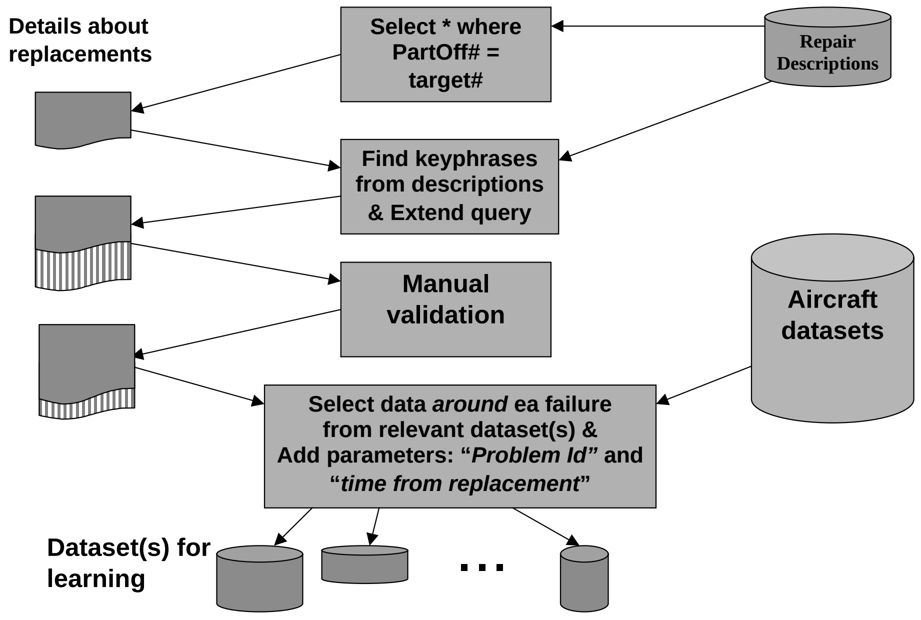
Details about replacements
Select * where PartOff# = target#
Repair Descriptions
Find keyphrases from descriptions & Extend query
Manual validation
Aircraft datasets
Select data *around* ea failure from relevant dataset(s) & Add parameters: "*Problem Id*" and "*time from replacement*"
Dataset(s) for learning
...

Figure 2: Retrieving relevant data for the analysis.

using cycles instead of days, but the results were quite similar. Consequently, we decided to only use days.

Finally, two new attributes are added to the initial raw measurements: the time between the observation is collected and the actual replacement time ("time from failure"), and a tag identifying each observation with a specific replacement case ("problem Id"). The instance from a given selected dataset are combined to create a dataset that will be used to build the predictive model. The number of output datasets is equal to the number of dataset(s)selected as potentially useful for the given component.

5 Data Labeling

As explained earlier, the data mining techniques we intend to use to build the models fall into the category of supervised learning algorithms. These algorithms require the input dataset to contain a class attribute (also called label or membership attribute). In our case, the class attribute may take two different values (0 indicating no need for replacement of the component and 1 indicating a need). In our approach, we set the class attribute to 1 for all instances obtained between the time of the replacement and the preceding k days (these k days define the window that we target for the predictions of the need for component replacement), and set the class attribute to 0 for all other instances observed outside that period of time. Obviously, k has to be smaller than m (see Section 4). Figure 3 presents the label value obtained from the labeling algorithm as a function of the number of days from replacement.

The interval for positive examples is predetermined for each component of interest. The choice for k is a function of:

1. Target period for prediction of the need for replacement of the component of interest. Different components may have different target periods.

2. Proportion of positive and negative examples. Without a significant proportion of positive examples, several data mining approaches are expected to have difficulties learning the desired models. To obtain at least 10% of positive examples, we use the following constraint $k > .1 * (m + n)$.

3. The complexity of the patterns to model. Some components may start giving indications of deterioration a while before they actually need to be replaced. The labeling approach should be adjusted accordingly.

For each component studied, we have performed a number of experiments with different values of k in order to satisfy the above criteria. Values obtained are between 10 and 40, depending on the component and the dataset used.

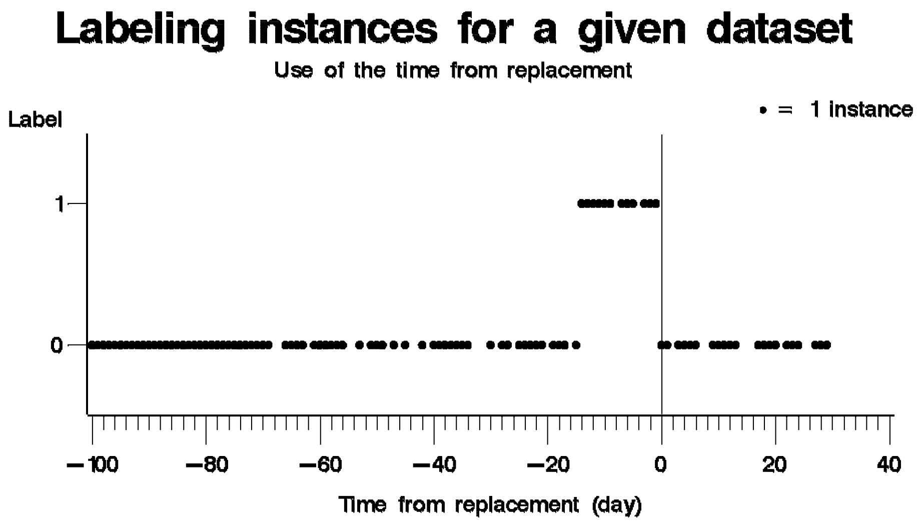

Figure 3: Labeling the instances.

6 Building and Evaluating models

The datasets obtained after the application of the labeling process can be used to learn the desired models. Several data mining techniques are available to infer these models such as Decision Tree, Instance Based, Naive-Bayes, Rough Sets, Regression, and Neural Networks. Depending on the technique used, one may need to further pre-process the data in order to: (i) select the most suitable subset of attributes, (ii) normalize the initial attributes, (iii) create new attributes from the initial ones, or (iv) discretize the continuous attributes. Approaches to perform these tasks are proposed in different areas of research such as machine learning, statistics, pattern recognition, and data mining [3].

A critical issue in any data mining application is the evaluation of the learned models. In order to decide on the suitability of a model we need to get a good estimate of its expected performance on operational data (i.e. new data that will be provided as input to the model once it has been deployed). There are a number of methods to compute the expected performance; these include *hold-out validation*, *cross-validation*, and *bootstrapping*.

Hold-out validation consists of randomly splitting the data into a training and a testing set. In this approach, one would first use the training set to build the model, and then estimate its performance by using the model to classify the instances in the testing set. The performance on the testing set becomes the expected performance of the model on new cases. This method considerably reduces the number of instances available to build your models. The method is therefore unsuitable when only a small dataset is available. Hold-out validation assumes that the instances are independent of each other.

Cross-validation makes use of the available data in a more efficient manner than hold-out validation. Cross-validation starts by randomly splitting the data into k subsets of approximately equal size. Then, you train the model using $k-1$ subsets and evaluate it on the remaining one. The previous step is repeated until each subset has been used for testing once. The expected performance is finally computed from the testing results obtained from the different runs. When k equals the number of instances, the process is called *leave-one-out* cross-validation. Cross-validation requires more computer time than hold-out validation but provides a more robust estimation of the expected performance.

Bootstrapping is even more demanding in terms of computation time than cross-validation. In its simplest form, you repeat the analysis (train and validate) by randomly sampling (with replacement) the test and training instances from the population. Thousands of repetitions are usually required. Experimental comparisons of bootstrapping and cross-validation tend to show that bootstrapping is better than cross-validation with some learning approaches such as stepwise regression [4] and significantly worse than cross-validation with other learning approaches such as decision tree [5].

Unfortunately, none of these three approaches is adequate for the application considered here. All of these approaches rely on random sampling at some point to select instances from the population. The problem with random sampling is that it implicitly assumes that the instances are independent. Whenever this assumption is not too severely violated, repeated analysis methods such as cross-validation and bootstrapping will probably lead to reasonably good results. However, with our application this assumption is simply not viable. Because of variations in the construction, operation, and maintenance of the aircraft, it is clear that any two instances from a given aircraft are not as independent as any two instances from two different aircraft. This means that if we use data from the same aircraft for training and validation the likelihood of success would be much higher than if we validate using data from a different aircraft. Similar reasoning applies to the data related to a given occurrence of component replacement versus data related to different occurrences of component replacement; two instances related to a given occurrence of a problem will be more related than two instances related to two different problems. Considering the fact that the number of occurrences of component replacement is significantly less then the total number of instances, random sampling is very likely to generate training and validation sets that contain data related to the same occurrence of component replacement. The end result will be an over optimistic evaluation of the performance of the model.

A simple solution to this problem consists of splitting the data such that the training instances come from a sub-group of occurrences of component replacement and validation instances come from another sub-group of replacement occurrences. This splitting is easily achieved using the *Problem Id* attribute added during the Data Gathering process.

A more robust estimate could be obtained by adapting the cross-validation method so that it accounts for the above splitting constraint. The process is as follow:

1. Split the data into batches (one for each failure case). This is done using the parameter *Problem Id*.

2. Keep one batch for validation and train using all other batches. Repeat this step until each batch has been used for testing once.

3. Get final evaluation from validation results over the different runs.

We have implemented this approach using the SAS system and MLC++ [6]. The term LOBO (leave-one-batch-out) has been used to describe a similar evaluation method in [7]. We have been using this method to evaluate the performance of a variety of learning algorithms for different aircraft components.

6.1 Evaluation function

In the previous section, we explained a method to come up with a fair estimate for the performance of a model, but we did not explain the actual performance criteria we are trying to optimize. In machine learning research, the reliability of a classifier is often summarized by either its error-rate or accuracy. The error-rate is defined as the expected probability of misclassification. That is the number of classification errors over the total number of *test* instances. The accuracy is $1 -$ error-rate. When some errors are more costly than others, one would prefer minimizing the misclassification cost instead of the error rate. Recent work on evaluation of classifier reliability proposed the use of ROC (Receiver Operating Characteristics) curves [8]. Other related metrics from the information retrieval community are the precision and recall. In this context, the recall is defined as the ratio of relevant documents retrieved for a given query over the number of relevant documents in the database, and the precision is the ratio of relevant documents retrieved over the total number of documents retrieved.

Unfortunately, all of these metrics fail to capture two important aspects of our application. The first one is the relation between the usefulness of a prediction and the time that separates it from the replacement. A too early warning about a potential failure will lead to a non-optimal use of the component while a too late advice may not let enough time for proper planing of the repair. What we need for this application is an evaluation method that takes into account the timeliness of the alerts. The second aspect is related to the coverage of potential failures. Because the learned model would be used to classify each report coming from the aircraft into one of the two possible categories (the component needs to be replaced or not), there is a potential for a model to generate several alerts before the component was replaced. More alerts may sometimes mean a higher confidence in the prediction. However, it is clear that we prefer a model that can generate at least one alert for most of the component failures, than a model that generates several alerts for only few failure cases. In oder words, the coverage of the model is very important because we want to minimize the number of unexpected failures. To take into account this preference, we need an overall scoring metric that considers the distribution of the alerts over the various failure cases. The next section presents a reward function to take into account the timeliness of the alerts while the following section introduces a new scoring metric that addresses the coverage issue.

6.2 Reward function

The criterion we have developed is based on the notion of reward. Predicting the correct outcome for an instance generates a reward. In its basic form, reward thresholds are fixed values (between 0 and 1), one threshold being defined for each possible class (or membership) value. In our application, we have extended this framework to accept: i) varying reward thresholds for different instances and ii) any real number as reward threshold. In particular, for each component of interest, we define a function that computes the reward for prediction of a positive instance based on the number of days between the time at which the instance is generated and the actual time of the replacement. Figure 4 presents the graph of such a function. For this example, the maximum gain is obtained when predicting the need for replacement between 5 and 40 days prior to the replacement of the component. We also observed that predicting a need for replacement of the component outside that target period may lead to a negative reward threshold since such a prediction corresponds to a misleading advice. According to this function, false positive predictions are penalized by a reward of -1.5 in comparison to a reward of 1.0 for true positive predictions.

We note that the target period for the reward function does not need to be the same as the labeling period, but the two are definitely related. In general, we first determine the target period for the reward function according to the user requirements and then we fix the labeling period. For instance, if the user says that predictions for a given component should be within 1-3 weeks in advance

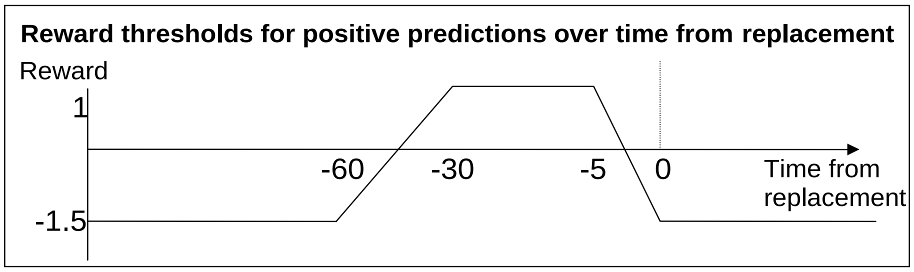

Figure 4: Example of a reward function for prediction of positive instances.

then we set the target period for the reward function between -7 to -21 days. Then we experiment with various labeling periods around this target period and take the one that leads to the best result. Setting the labeling period as the target period is often a good strategy.

The reward function illustrated in Figure 4 follows a piecewise first order polynomial model. Such a model is convenient because it is comprehensible for domain people and sufficiently complex to capture the relevant information. There are several ways to improve the precision of the function. One possibility is to use higher order polynomials instead of straight lines. As another possibility, one could try to smooth the overall function. However, domain experts often think that the increase in complexity is not justified.

6.3 Scoring metric

The above reward assignment method accounts for the timeliness of the alerts. However, to evaluate the coverage of a model, we also need to look at the distribution of the alerts over the different failure cases. The overall performance metric we propose to evaluate a model is:

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

where p is the number of positive predictions made by the model on validation instances during the application of the LOBO 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 in order to favor models that optimize the recall[1]. Let us observe that the value of $\sum_{i=1}^{p} score_i$ makes the final score sensitive to precision of the model.

7 Does it work? Experimental results

In the previous sections we presented solutions to fundamental issues in applying data mining techniques to predict the need for aircraft component replacement. A possible approach to evaluate the proposed solutions would be to compare each solution individually with alternative methods proposed in the literature. This is not realistic in our application for various reasons. First, we are addressing problems for which it is very difficult to find comparable methods in the literature. For instance, to our knowledge no one has proposed methods for data gathering and data labeling that can substitute the ones we proposed. Moreover, it is important to note that the proposed methods are very related. Therefore, evaluating solutions individually would not necessarily shed light on the usefulness of the overall approach proposed. These observations lead us to a different evaluation strategy in which we globally evaluate the potential of

[1]The term $(NbrDetected/NbrOfCases)^{SignOfSumOfScores}$ is set to zero when $NbrDetected = 0$ and $SignOfSumOfScores$ is negative

the proposed approach by applying it in a large-scale experiment. In this experiment, we test the potential of the approach to learn models for 16 aircraft engine components.

Since the aim was not to develop optimal models, we decided to simplify the process in various ways. First, we did apply pre-processing techniques usually required in order to optimize the results such as: cleaning the data by removing outliers, normalization of the data, creation of new parameters, and selection of the best subset of parameters (the same subset of attributes has been used for all learning approaches and all components). We only used one potentially useful dataset. There are at least 4 other relevant datasets that could have been used to infer models for these components. Moreover, the same settings have been used for the various parameters (i.e. $m = 100$,$n = 30$, and $k = 20$). Optimization of these parameters could have significantly improved the results.

For each of the 16 components, we have been using three different data mining techniques to learn the desired models: decision tree with C4.5 [9], Instance-Based (one nearest-neighbor) and Naive Bayes both implemented in MLC++ [6]. The evaluation has been performed using the methods described in Section 6. We used the same reward function for all components. The selected reward function is very similar to the one presented in Figure 4.

Table 1 presents the results obtained from this large-scale experimentation. The first column presents the part identification. For each part, we ran the three data mining approach as indicated in the second column. The third column indicates the number of folds used during the cross-validation experiment. As explained in Section 6, the number of folds used is equal to the number of occurrences of replacement we have found during the data gathering step. Finally, the fourth column shows the overall score of each model on the testing data. This score comes from the formula presented in Section 6.3.

Even if the data mining process has been substantially simplified during this experiment, we observe very interesting results. First, we note that we got at least one positive score for 11[2] components out of the 16 under study. This result clearly demonstrates the potential of the approach. Second, from the scores we can see that no classifier outperforms the other two for all components. This supports the idea that experiments with several data mining approaches are generally required in order to find the most suitable approach for the problem addressed. In our experiment, it is just like we have investigated 16 different tasks (one for each component) and each of these is likely to have its own "optimal" configuration in terms of techniques and parameter settings. It is therefore reasonable to think that these already promising results could be further improved.

An interesting aspect of the results obtained from the various classification techniques used is that the misclassification errors seem to be independent. In fact, correlation analysis of the predictions obtained from the three classifiers used lead to the conclusion that the different approaches tend to make mistakes on different cases. This observation is very important because it indicates a potential for improving the performance by combining predictions from the various classifiers. One should also remember that valuable models can also be developed using the other potentially relevant dataset(s) available and, therefore, further enhance the overall performance of the model.

[2]excluding the score value of 0.2 obtained for part 12

Part Id	Approach	Nbr of folds	Score
part 1	ib	16	16.9
	c4.5-rules	16	-33.8667
	naive-bayes	16	-103.383
part 2	ib	25	16.1133
	c4.5-rules	25	6.468
	naive-bayes	25	-111.813
part 3	ib	17	-22.5
	c4.5-rules	17	-34.5857
	naive-bayes	17	-75.6533
part 4	ib	36	140.195
	c4.5-rules	36	89.5222
	naive-bayes	36	-18.6225
part 5	ib	27	48.2733
	c4.5-rules	27	9.87677
	naive-bayes	27	-11.3261
part 6	ib	19	77.2133
	naive-bayes	19	48.8168
	c4.5-rules	19	15.5684
part 7	ib	31	10.1678
	c4.5-rules	31	-1.67809
	naive-bayes	31	-45.5741
part 8	c4.5-rules	41	-13.6536
	naive-bayes	41	-32.4486
	ib	41	40.2209
part 9	naive-bayes	12	9.9267
	c4.5-rules	12	2.98223
	ib	12	12.6
part 10	ib	28	42.16
	naive-bayes	28	0.52143
	c4.5-rules	28	-3.06133
part 11	naive-bayes	2	0.00
	c4.5-rules	2	-2.1467
	ib	2	-5.5667
part 12	c4.5-rules	16	0.2
	ib	16	-4.3733
	naive-bayes	16	-80.96
part 13	ib	53	48.088
	naive-bayes	53	1.45456
	c4.5-rules	53	-110.593
part 14	naive-bayes	40	-15.2416
	ib	40	-18.8133
	c4.5-rules	40	-43.8596
part 15	c4.5-rules	26	9.12
	naive-bayes	26	2.98458
	ib	26	-69.68
part 16	ib	31	99.0959
	c4.5-rules	31	9.40712
	naive-bayes	31	-40.0467

Table 1: Prediction results for 16 components from 3 data mining approaches obtained using one dataset. High score values mean good performance.

8 Related work

In terms of the tasks addressed, the closest works are found in the area of Reliability Analysis. These works concentrate on the development of models to assess the reliability of individual units or complex systems. The models developed may then be used to predict the probability that a specific component(or system) will fail within a given period of time. Predicting the failure of a component is obviously very similar to the prediction of the need for replacement of a component.

In reliability analysis, the data is usually obtained through a planned experiment in which one observes the evolution of a sample of units and record the failure times. The analysis almost always starts by fitting the failure time values to a known probability distribution. Knowledge about this distribution is then used to infer models to predict *the time to failure* of similar units. The approach introduced in this paper differs considerably from this process. First, the data available in our application does not come from an experiment and we have no control on the variables measured and on the quality of the data received. As a consequence, major pre-processing work (as explained in Sections 4 and 5) had to performed in order to retrieve the desired data for the analysis. Second, the parameter *time from replacement* does not play such an important role during the modeling step. On the other hand, we extensively use it during pre-processing of the data. This was required in order to compensate for the fact that most data mining approaches are not designed to benefit from time information. Finally, the evaluation methods are also different. In our application, we rely on a novel metric that meets specific domain requirements to evaluate the models. In reliability analysis, the evaluation is mostly based on traditional statistical measures such as mean-squared error. Crowder et al. [10] present an up-to-date account of the analysis of reliability data. Researchers in the area of Survival Analysis make use of techniques similar to the ones find in reliability analysis, but mainly in the context of medical and biological applications [11].

There has been research on applying data mining techniques to identify problems with aircraft gas turbine (e.g. [12, 13, 14]). These works are, however, limited to the identification of current problem(s) with the engine and do not try to predict potential problems in advance. The data used is also different; our approach makes use data obtained during normal operation of the engines while the above works are based on data acquired in engine test cells. The applicability of the results is therefore very different.

9 Conclusion and further work

We have presented an application in which data mining techniques are applied on operational and maintenance data. Our work has been data-driven. Having decided the kind of model to be developed (a classifier), we have focused on addressing the complex characteristics of the data. Our data is heterogeneous (different data sets, some data is numerical, some is text, and some needs to be interpreted as messages in natural language). The data is also time-sensitive (most of it comes with a time-stamp which is relevant for model building). Moreover, the data is unprepared for the development of the model: it lacks class labels or ranking. Finally, it was clear from the outset that adequate methods for the evaluation of models for this kind of application were lacking.

The contribution of this paper is therefore in the areas of data gathering, data labeling, and in its novel approach to evaluation. In data gathering, we show how a simple process based on the use of off-the-shelf NLP tools can help with data cleaning and integration. In data labeling, we propose an empirical, time-sensitive approach which trades off the proximity of an alarm to the time of replacement against the requirement of a reasonable number of training instances. In evaluation, we suggest an evaluation function which combines the timeliness of an alarm, the recall of positive training instances, as well as their precision.

In the course of this project, we have produced several different models (alarm systems) for our application. Originally, we expected to experiment with different models and choose the one with the best performance. However, our work to date has shown that it is more appropriate to consider model fusion. We are currently working on model fusion, combining all or some of the models in order to improve the performance of the resulting classifier. In this work, we are exploring some of the ideas on the ensembles of classifiers [15]. Since, as mentioned in Section 6 , the errors of different classifiers are uncorrelated, we expect an improved performance from a set of combined classifiers.

Acknowledgment The authors are grateful to Air Canada for supplying the data and explaining different aspects of aircraft health monitoring, and to the Natural Sciences and Engineering Resarch Council of Canada for the support to the first and third author.

References

[1] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley, New York, 1972.

[2] P. Turney. Extraction of keyphrases from text: evaluation of four algorithms. Technical Report NRC 41550, National Research Council of Canada, 1997.

[3] D. Pyle. *Data Preparation for Data Mining.* Morgan Kaufmann, San Mateo, CA, 1999.

[4] B. Efron. Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78:316–331, 1983.

[5] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth International Group, 1984.

[6] R. Kohavi, D. Sommerfield, and J. Dougherty. Data mining using MLC++: A machine learning library in C++. In *Tools with Artificial Intelligence*, pages 234–245. IEEE Computer Society Press, 1996. Received the best paper award.

[7] M. Kubat, R.C. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30:195–215, 1998.

[8] F. Provost and T. Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In David Heckerman, Heikki Mannila, Daryl Pregibon, and Ramasamy Uthurusamy, editors, *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, page 43. AAAI Press, 1997.

[9] J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, CA, 1993.

[10] M.J. Crowder, A.C. Kimber, R.L. Smith, and T.J. Sweeting. *Statistical Analysis of Reliability Data.* Chapman & Hall, London, 1991.

[11] D. Collett. *Modelling Survival Data in Medical Research.* Chapman & Hall, London, 1994.

[12] T.W. Brotherton and G. Charddedon. Automated rule extraction for engine heath monitoring. In *Evolutionary Programming VII: 7th International Conference, EP98*, pages 725–734, 1998.

[13] P. Turney and M. Halasz. Contextual normalization applied to aircraft gas turbine engine diagnosis. *Applied Intelligence*, 1993.

[14] W.E. Dietz, E.L. Kiech, and M. Ali. Classification of data patterns using an autoassociative neural network topology. 1989.

[15] T.G. Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18(4):97–136, 1997.