Final Report

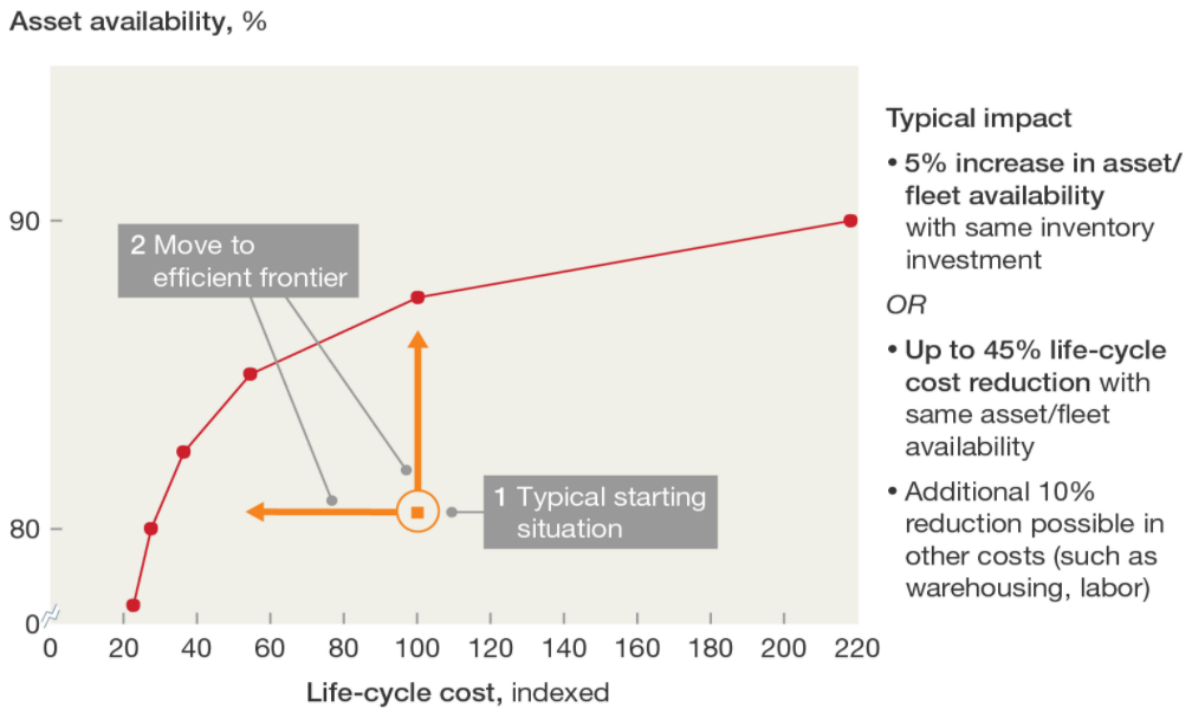
1. Executive Summary

MRO optimizer, one of the Hamiltonian Systems' products, is a program to optimize Maintenance, Repair, and Operations (MRO) inventory for the asset-intensive industries, such as steel, airline, oil, and paint. The current MRO optimizer is mainly based on a statistical judgmental model that needs manual intervention based on past work order data. However, this scope of optimization and customization is very limited.

Our objective is to improve the product by developing machine learning models based on provided features to help customers to achieve better failure prediction (Remaining Useful Life of items) and inventory management.

2. Problem Statement

MRO is one of the most challenging aspects to be managed by large companies since it consists of a large network of vendors who sell products at various prices. The main obstacle is unpredictable demand based on failure. While firms want to achieve the optimal level asset availability and life-cycle cost, the trade-off is real and must be evaluated.



Source : https://www.mckinsey.com/business-functions/operations/our-insights/smarter-choices-raise-asset-availability-and-reduce-operating-costs

Defining assets that pose risks to profitability due to downtime and the optimal inventory arrangement is the challenge.

Looking at reliability issues and maintainability issues, this problem of staying within the maintenance monthly budget can be tackled.

3. Project Objectives

Our objective is to explore a better inventory management approach in the asset-intensive industries that can be incorporated in Hamiltonian Systems' product suite. We aim to provide solutions for firms to be able to increase their asset availability while saving their maintenance spendings.

We decided to develop machine learning models to predict the Remaining Useful Life (RUL) of each item in each asset, based on provided features such as past usage, asset criticality, and failure data.

We also hope to learn to improve our teamwork and cross-functional communications skills by working on a real-world project.

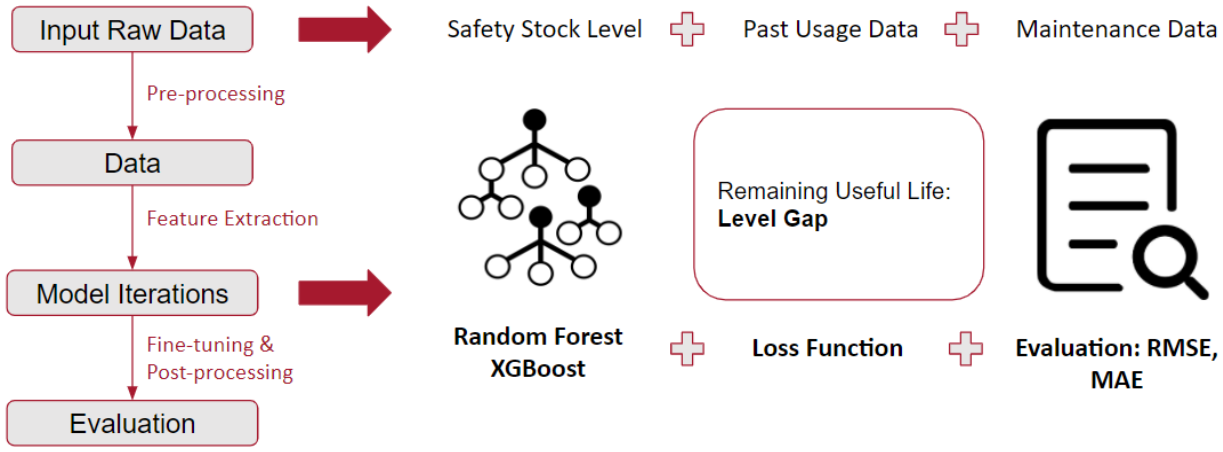
4. Value and Business Impact

Many companies are still relying on manual inspections to make decisions on procurements and asset management, which is both inefficient and uneconomic. We interviewed existing clients of Hamiltonian Systems to understand that the main concern is that current models used do not prioritize items and assets that need replenishment. Our Remaining Useful Life model will help firms schedule their work orders and aid in staying within the monthly maintenance budget allocated. It would lead to higher profitability and competitiveness. Gaining insights from the clients, they showed a high willingness to pay for an MRO optimizer, which will help reduce their maintenance spendings and achieve better asset management. Thus, the business impact of our solution is significant in terms of money and resource allocation.

To be more specific, analyzing the industry trends, we expect asset-intensive industries to save 10%-12% of their maintenance spendings ($25 million - $3 billion) with our product. Our failure prediction helps our clients schedule the orders with precise quantities in a timely manner to achieve better asset management and smarter spending with MRO items. By prioritizing work order purchases, it will optimize labor and reduce redundant activities from weeks to hours. This will also provide a competitive advantage to the clients by capturing asset availability resulting in a reduction of downtime.

5. Project Methodology

We used a machine learning pipeline to pre-process the raw data, extract features, and to fine-tune our model iterations in order to evaluate the best model possible. The graph below shows an overview of our approach.



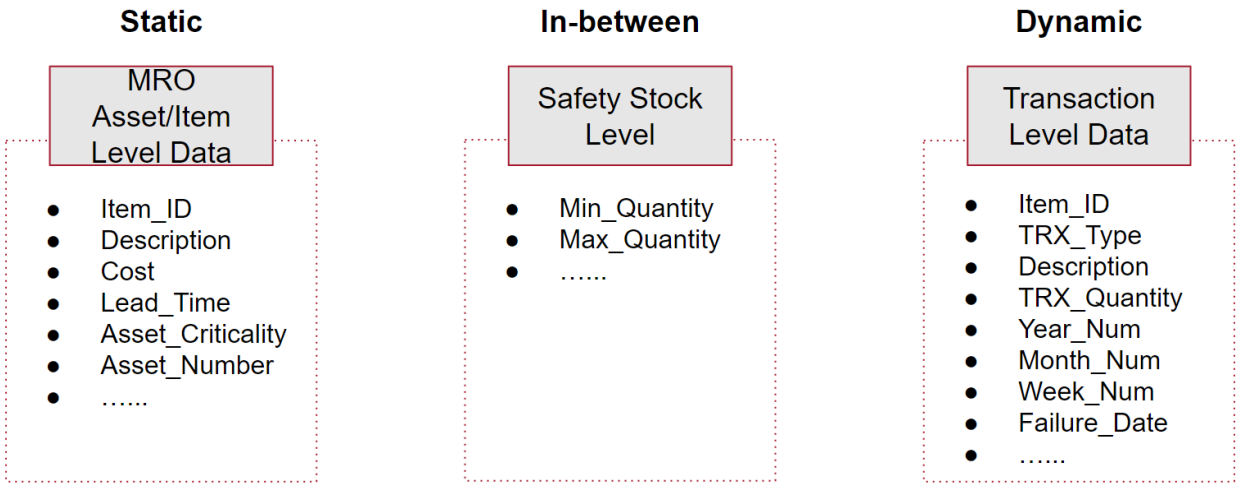
We zeroed down on Random Forest and XGBoost since it performed most optimally.

5.1 Data Analysis

We performed some analysis on the data we have. On a high level, there are about ~95,000 distinct items that are used across 65 different organizations. There are around 440,000 transactions that happened in the past 5 years with an average transaction quantity of 6. This has been consistent from years 2015 to 2020. The year 2018 saw the highest number of transactions - 102,000 which is a 47% year-over-year growth from the previous year.

There are some imbalances and variances in the data. We found that top 10 organizations contribute 60% transactions, while many organizations only account for a few items and their transactions. From the perspective of safety stock levels, 49% of the transactions saw on-hand quantity to be greater than max quantity, while 20% had on-hand quantity to be less than min quantity. This means the Min-Max levels defined by maintenance staff are not very accurate, and that a more scientific approach should be applied to monitor the inventory better.

Going deeper into the features of our datasets, we can divide them into three broad categories: Static, Dynamic, and In-between based on their characteristics and status. Static features are item-level data that provides the information of each item, such as Item_ID and Description. Dynamic features are transaction level data that vary every time when a transaction happens. These include features like TRX_Quantity, TRX_Type, and also failure data such as Failure_Date (it varies for each transaction). Finally, there are safety stock level data that are considered “In-between”, for example, Min_Quantity and Max_Quantity. See graph below for a better understanding.



From these analyses, we obtained a good understanding of the data that helped the problem solving and modeling process.

5.2 Data Preparation

The dataset we have contains around 5 years records for various maintenance, repair and operation items and work orders. Therefore, we first merged the tables together using SQL queries in Oracle SQL developer to create a master table that contains all relevant features: past usage data, maintenance data, failure data, safety stock level, and so on.

Then comes the data pre-processing. There are some features that were cleaned, transformed, or combined. For example, for the asset criticality, we transformed the string values to categorical numbers because it's enough to indicate the criticality and could be used directly in modeling. We did the same for the work order types and item IDs. We also handled null values with the help of business teams to decide on the appropriate alternative.

Once our dataset was ready, we went ahead with feature extraction for the model building phase.

5.3 Feature Engineering

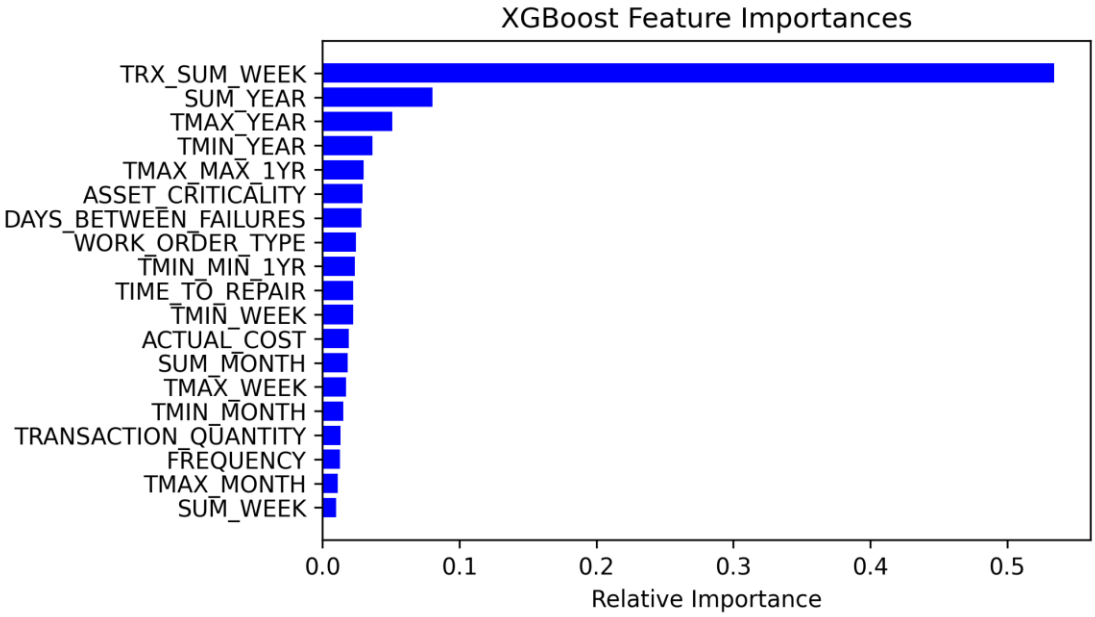
Feature engineering is another important step. One of the chief factors that helped us predict remaining useful life more accurately, was to create a unique ID which combined item number, organization code, and asset number. This gives us a closer look at the transactions because the combination of these three features is the smallest unit that defines failure. We also created usage features that correspond to the Item_ID.

Another important feature we extracted is the dependent feature - Remaining Useful Life (RUL). We calculated RUL for each item based on the difference between two dates and the quantity used during that period. For more precise modeling, we further defined two types of RUL features: Average RUL which aggregates maintenance and usage data over time, and Variable RUL that varies with time.

5.4 Feature Importance Analysis

To help us understand the impacts of different features on RUL, and the relationships between different features, we also conducted a feature important analysis with the XGBoost model (as shown in the graph below) and gained several important insights.

Firstly, the result shows that item and asset level usage related features are the most important features in our case, like TRX_SUM_WEEK. In terms of transaction time, annual level features, like SUM_YEAR, TMAX_YEAR and TMIN_YEAR, are also of great significance. This is partially because some transactions happen only on an annual basis. Besides, asset criticality and failure-related data are found helpful in our case as well. The feature importance analysis inspired us to construct and choose the most significant features in our modeling stage.



5.5 Modeling

For the modeling part, we decided to use Random Forest and XGBoost. The reason we used these two algorithms is that they are good at modeling non-linear relationships. Also, these algorithms can help us find out the most important features to do further analysis. The loss function we used is the difference between predicted and actual RUL, and the evaluation is Root Mean Squared Error (RMSE) as well as the Mean Absolute Error (MAE).

We did iterations of different models, evaluate their performances, and go back to change the features and fine-tune the model hyperparameters.

5.6 Results

We compared two algorithms: Random Forest and XGBoost, as well as two approaches: Average RUL and Variable RUL.

We used the Mean Absolute Error (MAE) as the metric (RMSE mentioned above yielded slightly higher results), the result is shown in the table below:

	Average RUL	**Variable RUL**
Random Forest	219.71	272.48
XGBoost	212.28	261.06

As we can conclude from the table: from the perspective of the algorithms, XGBoost performs slightly better than Random Forest; when it comes to different calculations of RULs, Average RUL performs better than Variable RUL.

The variable RUL, intuitively, should perform better than the average alternative because it takes into account the dates that the failures happened. However, the results suggest the opposite. This could result from a lack of time-varying features: the approach of Variable RUL itself may be better, but it needs enough features that are changing with the transaction dates. When more features are available, Variable RUL could be more superior.

The reason XGBoost performs slightly better could be because of the gradients that this algorithm uses, which could be more suitable for this case. Nonetheless, both algorithms have the same level of performance. Since these algorithms are able to find the most important features, other methods such as neural networks are not expected to achieve much better results.

7. Conclusions and Recommendations

We have learned that MRO datasets are often challenging to deal with. First of all, there are many items or assets that are only used a few times throughout the years, making it difficult to predict the failure. Moreover, some useful features, such as the sensor data, are still not available in many companies, which makes the prediction harder. If we can resolve these two problems, the machine learning algorithms can give us more insights and more accurate predictions.

Despite the challenges that we faced, we found that the usage data and the failure data are helpful. This finding helps to highlight the features that can be used in the future. Among the usage data, the annual level tends to be more important, partly because the transactions sometimes happen only a few times a year. The combination of item ID, organization code, and asset number can be useful too, because it reflects the actual transactions in a failure.

In conclusion, MRO is a promising but underinvested area. We have found that some features and future steps can be helpful, but big companies are still not investing much in this area, even though they can end up saving more on MRO spendings. Therefore, we recommend further study and research in this area.

8. Lessons Learned

This project is really an educational journey, we have all learned a lot from each other, from the project itself, and from everyone we worked with.

One of the biggest lessons that we learned is how to solve a real-life industry problem that could really make a difference. And it turned out to be very different from what we did for the coursework. The challenges mostly came from the data. Firstly, companies provided a lot of different features, some of which may be redundant or ambiguous. This is unlikely to happen in the school assignments, where the features are well-defined and the emphasis is on the approaches to use them. Secondly, some of the features that intuitively can be helpful are not available. This is also what a real-world problem asks us: how to extract useful information from whatever is available at hand.

Furthermore, we learned a lot during the teamwork and communication with everyone. In this unusual time, we were unable to meet each other in person - the only way to reach out is through emails, conference calls, or social media. Facing these difficulties, we had to quickly adapt to this new style, become more responsive and responsible, and try to deliver our work through screens in a professional manner.

Last but not least, we gained more experience in reporting our progress regularly, which is very common in companies but not so often in colleges. This extra responsibility of the weekly updates makes us more productive and keeps us on track, and more importantly, it prepares us for future careers.

9. Suggestions for Future Work

Although our team, Heinz College, and Hamiltonian Systems have put great effort into this project, there is still room for improvement. We suggest the following steps that can be taken to make advances in solving this problem.

First, as our analysis has shown, more features can be added to make our model better. One of the common categories is the sensor data, which is often used to predict the failure of each item. The sensors give a more accurate measurement of the conditions of items, like temperature cycle which map the reason for failure hence they can be really helpful. According to our understanding of the companies in the asset-intensive industries, some of them are

already considering deploying IoT devices or sensors to better monitor the inventory. When these features are available, we believe that the prediction can be more reliable.

Second, it would be better if the model can capture something more than the time of the failure, such as the quantity needed to make maintenance. While the Remaining Useful Life tells us the next date of the failure, the quantity tells us how many are required. Combining these two can give the companies a better understanding of their next procurement.

Finally, it is also important to make sure that the model is able to integrate well with the existing software of Hamiltonian Systems. The deployment of the model will require a good knowledge of the company's system infrastructure and the implementation of suitable programs such as a good API.