Summer 2021

DataScope: Predictive Diagnosis in IIoT-enabled Smart Manufacturing

Adam Geltz

Reilly Sollenberger

Peilong Li

DataScope: Predictive Diagnosis in IIoT-enabled Smart Manufacturing

Adam Geltz
Elizabethtown College
geltza@etown.edu

Reilly Sollenberger
Elizabethtown College
sollenbergerr@etown.edu

Dr. Peilong Li
Elizabethtown College
lip@etown.edu

Dr. Srikiran Chandrasekaran
CPNet, LLC
srikiran@cpnet.io

Dr. Bicheng Chen
CPNet, LLC
bicheng@cpnet.io

Dr. William Gordon
HPI Resources, LLC
gordonb@socialenterprisegroup.org

ABSTRACT

This a collaborative project between Elizabethtown College and CPNet, LLC that is looking to help apply predictive modeling with CPNet's domain knowledge to one of CPNet's clients IIoT manufacturing problems. CPNet has provided us with datasets taken from one of their clients in the hope that we can build a model that will be able to predict when a part within the machines they are looking at will fail and subsequently shut the machine down. We are trying to take their data and turn it into information that the company can take preemptive action on and save them downtime during operation.

In this project, we designed a health index that we used to create y-labels that we did not have in our dataset. We did this so that we could solve the remaining useful life problem of our project, which is where we attempt to determine how long the machine has to run before a failure or maintenance is needed based off of the health index using machine learning.

We then went on to build several machine learning models to solve our problem. First, we used traditional machine learning models such as Polynomial Regression, Tree-based Regression, Ridge, Regression and XGBoost. Later we turned to Neural Networks and built a Multilayer Perceptron, a Convolutional Neural Network and a Recurrent Neural Network.

From our experiments traditional machine learning models outperformed the neural networks. This was to be expected since the dataset wasn't that large, but it was worth testing regardless. Also, from our experiments it is apparent that a part of our health index the CPK (CPK analysis will be explained in greater detail later in the paper.) will need to be worked on in the future to provide better y-labels. All in all, this project did show that this problem is worth exploring in the future since the solution could be quite valuable in smart manufacturing.

1. INTRODUCTION

The goal of this project is to create a predictive maintenance model for CPNet's client. Through predictive maintenance we are hoping to predict the life of the machinery so that CPNet's client does not have to run the machine till death. Instead, the model could predict when the machine will fail and allow the workers to shut the machine down in advance and work on it or schedule maintenance before the machine fails. This will reduce the downtime of the machine since the work can be planned and also help eliminate faulty product that is made after the machine's parts are out of spec.

1.1 Existing Solution

One existing solution for this problem is Statistical Process Control (SPC), which is a way of performing quality control using statistical methods to monitor a process. We are hoping to be able to improve upon the ideas behind SPC with our research.

1.2 Our Solution

1.2.1 Original Plan of Attack

Originally, we were going to find a solution using a two-pronged approach. First, we would design away of detecting faults that would cause the machine to go down. Afterword's we would implement a predictive maintenance model to predict when these faults would happen in the future so that an alert could be generated and sent out before the machine would reach a fault and shut down.

1.2.2 Plan of Attack post project scope change

The scope of the original idea was deemed to be too large for the 10-week period allotted for this project, so we decided to just focus on the predictive maintenance aspect of it. This project could be expanded on in the future for other SCARP projects. After now knowing that we are now focused only on the predictive maintenance portion of the problem we decided to use Remaining Useful Life (RUL) that we learned about while doing some literature reviews to solve this problem.

1.2.3 Predictive Maintenance

The task that our model will be performing is predictive maintenance as stated earlier, in which we try and predict the RUL of the system or in this case the machines. The RUL can be calculated for any working part on the machine and is calculated by determining how far a unit or dimension of a unit is from its specification limits. To determine the RUL, we need to construct a Health Index (HI) since we have no labels for this dataset this will provide labels for us to make a supervised learning model from. These methods will be discussed in further detail in part 3 of the paper.

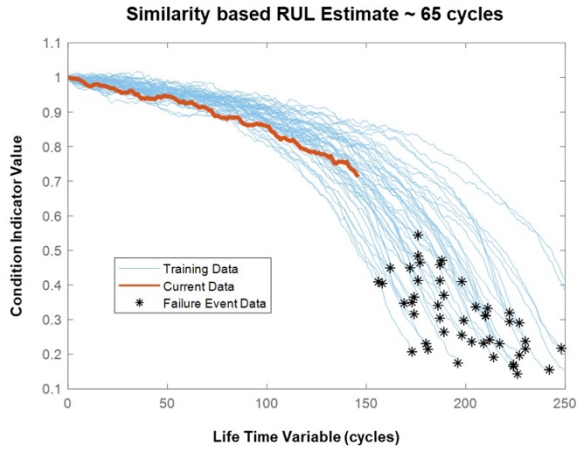

Figure 1. An example illustrating how RUL can be determined using the health index. Where at left the example starts at 1.0 being completely healthy and over time the health deteriorates and approaches 0 until a failure event is triggered.

1.3 Expected Results

Our expected results are that with the given data and tolerances for the machines provided to use from CPNet we will be able to build a RUL Regression model that will allow for prediction of machine failures. After which CPNet will be able to either implement the model or we will have at least explored an avenue that may work that they can either expand upon or leave for later research in the future for other SCARP projects.

1.4 Paper Organization

In the follow portion of the paper, the background, we'll look into some general approaches that CPNet has taken to solve this problem. Also, we'll look into some ways that predictive maintenance and RUL have been solved in research and application that we have discovered while researching a way to solve this problem. This section also shines more light onto topics such as RUL and Health Index.

After the background, in the Design section we will discuss the data used and the predictive models that will be used in greater length. In the fourth section, Experiments, we will explain the different machine learning models and methods that we plan on testing for this project. In section five, Model Evaluation, we will explain the findings from running the several different models and explain why we think some worked and some didn't.

In section six, timeline, we lay out the timeline of the project and explain how it has evolved as demands changed throughout the weeks. Section seven, Future Plans, discusses how we plan on continuing this project into the future. Following this section is the final section, section nine which lists the relevant reference material for our project.

2. BACKGROUND

In this section we will briefly discuss in 2.1 about who CPNet is and in 2.2 discuss the background for our problem.

2.1 Who is CPNet

CPNet as stated on their website is "… an IoT and AI company founded to bring Industry 4.0 technology to mid-market producers across several traditional manufacturing verticals, helping them to improve productivity of their legacy equipment at a fraction of replacement cost." I will site their website in the references section for further info about the company.

2.2 The Problem

CPNet has devised their own way of solving this problem. The problem once again is creating a way of indicating when a machine at their client's facility is going to fail before a failure occurs. Their method of indication involves the use of an anomaly detection model that then gives out sets of alerts based on whether or not a failure is immanent.

As stated earlier, in our project we are going to be creating a regression model based on predictive maintenance. In predictive maintenance the goal is for the model to estimate the remaining useful life (RUL) of a system. In this case the system is the machines of CPNet's client. This is done by creating a condition

indicator (in our case Health Index) that is used in the Regression model as the y-value for predictions.

As stated above the properly make a predictive maintenance model, we need y-values in the form of some 'condition indicator' which we are using 'Health Index' for. The Health Index indicator that we are using comes from a NASA white paper 'Damage Propagation Modeling for Aircraft Engine Run-to-Failure Simulation'. They use this Health Index for generating y-values to use in their model of RUL on their aircraft turbine engines. But it can very much be repurposed for generating y-values in our problem. After building the health index we are able to move on to model building for the RUL. It is made up of the minimum of all the minimum thresholds for all the dimensions of each machine part in the machine.

3. DESIGN

In this section we will be discussing the project model that we will be following that is distinct to most data science projects in 3.1. In 3.2 we will discuss the data that we will be working on and how we obtained it. In 3.3 we will discuss the predictive models that we will be using or have thought about using to solve our problem.

3.1 Jeff Hammerbacher's Model

The model we will be following is Jeff Hammerbacher's Model, in involves 7 distinct steps in a data science project. Where first the problem is identified, find our data sources, collect said data, data preparation, model building, model evaluation, and finally the results are given. In the following graph our outline is given in the context of our problem. Something to take not of is that Steps 1-3 were already figured out for us since CPNet already had a problem and the appropriate data needed to solve it.

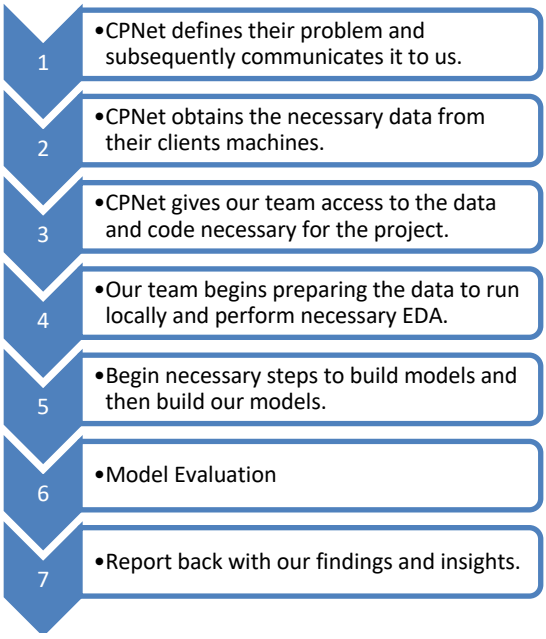

Figure 2. Jeff Hammerbacher's model updated with our steps per our project.

3.2 The Data

The data that we have was given to us from CPNet, so we conveniently did not have to source our data and then collect it, that process was done for us already. CPNet obtained this data from one of their customers, and the data comes from machines that customer has at their facility. The data they gathered from the machines was also aggregated for us by CPNet's lead data scientist making it easier to work on. These machines produce 700 data points per minute and sometimes 1400 depending on how the machine, so it was aggregated down to minute data.

The data itself comes from different parts and their dimensions on the machines at CPNet's customers facility. Certain part dimensions have upper and lower bound thresholds that they must stay within for the machine to remain operational. If the dimensions exceed either one of these limits, then the machine starts producing faulty product and will shut down.

To get the proper data we needed to work with, we had to use some of CPNet's code that involved some feature engineering on their part. We obtained the thresholds from them so that we could then compare the dimension data to the thresholds. We then are going to create CPK's for each of the features.

CPK is a standard practice in manufacturing that has been in use for many years now. Its purpose is to use statistics to measure how "in-control" the machine is. In order to generate it, you need only follow a simple formula using basic statistics, which will give you a single value output. A CPK of 2 is considered excellent, while 1.33 is considered acceptable. In our case, we wanted an average CPK for all of our specifications, rather than just one at a time. To accomplish this, we developed a function to derive the CPK for each specification, and used PCA to find the explained variance ratio for each spec. We then used these values to calculate a weighted average of all CPKs that could represent the overall health of the machine.

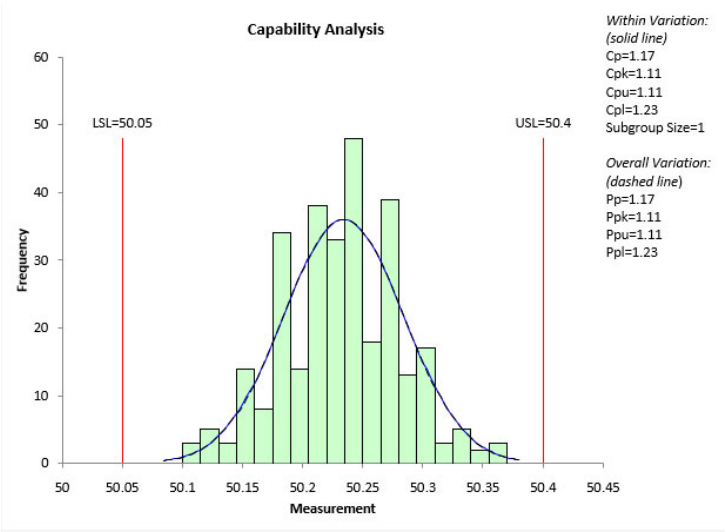

Figure 3. An example of CPK as discussed above.

Then we will perform principal component analysis (PCA) to find which features are contributing the most to the life of the part. Afterword's we will be able to generate our health index and subsequently the RUL.

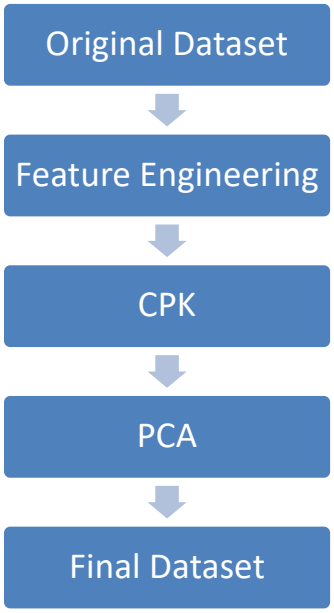

Figure 4. Data Transformation Pipeline

3.3 Predictive models

Since we developed the health index for our data and now have labels, this problem can be solved with supervised learning models. Also due to the nature of this being a remaining useful life problem we will want to use some sort of regression model to predict that path that the health index(y-axis) takes from 1.0 (healthy state) to 0.0 (failure state) over time (x-axis).

4. EXPERIMENTS

We will be testing out how both linear and polynomial regression work on our dataset. Most likely with remaining useful life problems then tend to curve down towards a failure state and don't move in a linear direction. But since it won't take much time, we will still build the linear model, even if it doesn't work it will still be nice to compare the polynomial model to. We are mostly going to be worried about building a polynomial model that will be able to fit the down trending curve of our health index over time. We may have to work with other regression models depending on how the polynomial model works. Some of the models that we are looking at trying to include, tree-based regression, ridge, regression and XGBoost. We will also be looking to try testing some neural network models after some of the more basic models are built and evaluated.

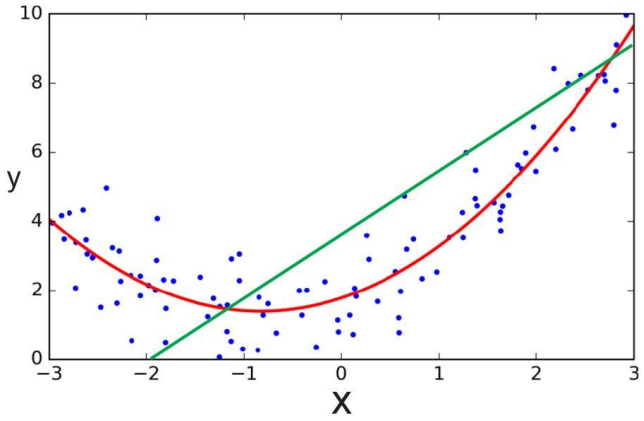

Figure 5. A comparison of linear vs. polynomial regression on curved data. Linear model plotted in green; polynomial model plotted in red.

The problem we have with building neural networks is that the dataset doesn't really lend itself well to those types of models. It's a somewhat small dataset compared to what usually is needed for training neural networks, but we are still going to experiment with them regardless since in the future we may have access to more data.

Some of the neural networks that we are looking at using are Multilayer Perceptron, Convolutional Neural Networks and Recurrent Neural Networks. For now, we will be building some basic versions of these models but in the future, we will be looking to use some models that are fairly popular and in use by others. For example, some CNN models we may use are LeNet-5, AlexNet, VGG-16 and others.

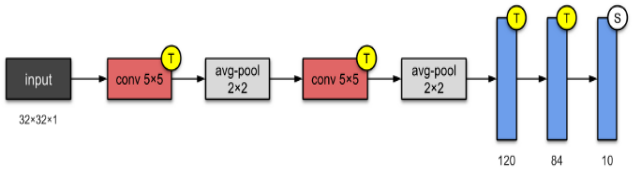

Figure 6. Here we have an illustration of the LeNet-5 CNN architecture.

5. MODEL EVALUATION

Since we weren't able to complete our final version of the CPK, we had to settle on using a simpler version of it so that we could at least generate some y-labels to evaluate our models on. Due to this a lot of the models performed poorly but that is to be expected since we don't have a finished CPK, this is something we look to complete in the future for further research and testing.

In regard to the models, the traditional machine learning models seemed to work better than the neural network models, this again was to be expected since neural networks often work better when t given much more data to work on. Since our dataset was not very large it makes sense that the neural nets didn't perform that well. Regardless, of the traditional machine learning models we did see some standouts and which ones seemed to work much better.

XGBoost seemed to work the best by far, followed by Tree-based Regression. Ridge and Polynomial Regression didn't seem to work very well but at least had some decent scores. As for the neural network models, they did not perform well at all and while it was interesting to at least experiment with them, until the CPK is done and we have more data, we will probably want to focus on the traditional models.

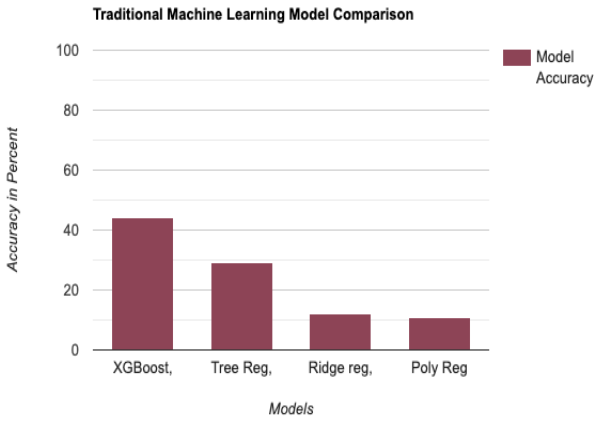

Figure 7. A comparison of the traditional machine learning models that we used. The Neural Networks are left out because their accuracies were all close to zero.

6. TIMELINE

The first two weeks of the project we were given the dataset from CPNet and spent some time exploring it and doing some standard EDA. We were also given their code with which they generate alerts based on anomaly detection, this code was dependent on a library they use internally so we had to refactor it so that we could run the code locally. After words in week 3 we spent time doing a literature search to find ways of solving our predictive maintenance problem and at the time the fault detection problem. After our weekly meeting at the end of week 3 we decided to focus solely on the predictive maintenance problem. In week 4 we began coding up the health index and did some preliminary graphs of the machine dimensions using the spec thresholds provided too us.

During weeks 5 and 6 we continued working on creating the CPK but ran into trouble with that due to time constraints and the complexity of the problem. So, we decided to just use a pre-built CPK that CPNet already made for the time being. During this time, we also began building our first models, XGBoost, Polynomial Regression, and Tree-Based Regression.

During week 7 we started researching Neural Networks and built our Multi-layer Perceptron model. In week 8 we built our Convolutional Neural Network and Recurrent Neural Network and began exploring some other models for the future such as LSTM and more advanced CNN models. In week 9 we prepared for and presented at the Landmark Conference. In week 10 we began cleaning up our existing code from the project. We then created a main file and an accompanying read me to explain the code. Afterword's we transferred the code over to CPNet.

7. FUTURE PLANS

While the project didn't go exactly how we planned at the beginning. We were still able to identify a problem that is worth solving in smart manufacturing and began making progress towards solving that problem. Although our results at the end weren't the best, it was still worth it to go through the motions of solving this problem, so we now know where we want to focus our efforts in the future to fix problem areas.

Going forward one of the first things that we'll need to be done, is to complete our CPK. That alone will hopefully be able to raise our model scores by a significant degree. Of course, there is also room for improvement in our models, both traditional and neural networks. Fine tuning them and utilizing some more advanced neural network models may also work provided the dataset is large enough for the neural networks.

We will also be looking to utilize a dataset that comes from NASA that they used to test predictive maintenance on turbofan degradation over time. We're hoping that since this dataset is larger and has already been used for similar purposes, that we can use it to test our code on. Afterword's we can take our findings from that and apply it again to our own dataset. This dataset should be large enough to make the neural networks effective as well.

8. REFERENCES

[1] Baru, Aditya. Three ways to Estimate Remaining Useful Lifefor Predictive Maintenance. https://www.mathworks.com/company/newsletters/articles/three-ways-to-estimate-remaining-useful-life-for-predictive-maintenance.html

[2] Saxena, Abhinav. Goebel, Kai. Simon, Don. Eklund, Neil. Damage Propagation Modeling for Aircraft Engine Run-to-Failure Simulation. https://ti.arc.nasa.gov/publications/154/download/

[3] McNeese, Bil. What is Cpk?
https://www.spcforexcel.com/spc-blog/what-cpk

[4] Karim, Raimi. Illustrated: 10 CNN Architectures
https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d