

**TIME SERIES ANALYSIS WITH DEEP LEARNING APPROACHES FOR  
INDUSTRY 4.0**

(ENDÜSTRİ 4.0 İÇİN DERİN ÖĞRENME YAKLAŞIMLARIYLA ZAMAN SERİSİ  
ANALİZİ)

by

**Ceren Nur BAŞ, B.Sc.**

**Thesis**

Submitted in Partial Fulfilment

of the Requirements

for the Degree of

**MASTER OF SCIENCE**

**in**

**COMPUTER ENGINEERING**

**in the**

**GRADUATE SCHOOL OF SCIENCE AND ENGINEERING**

**of**

**GALATASARAY UNIVERSITY**

Nov 2019

This is to certify that the thesis entitled

**TIME SERIES ANALYSIS WITH DEEP LEARNING APPROACHES FOR  
INDUSTRY 4.0**

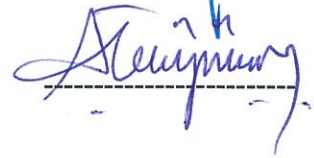
prepared by **Ceren Nur BAŞ** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering** at the **Galatasaray University** is approved by the

**Examining Committee:**

Assoc. Prof. Dr. Özlem DURMAZ İNCEL(Supervisor)  
**Department of Computer Engineering**  
**Galatasaray University**

Assist. Prof. Dr. B. Atay ÖZGÖVDE  
**Department of Computer Engineering**  
**Galatasaray University**

Assist. Prof. Dr. Ayşegül TÜYSÜZ ERMAN  
**Department of Computer Engineering**  
**Işık University**



Date:

14.11.2019

## **ACKNOWLEDGEMENTS**

I would like to acknowledge and thank the following important people who have supported and encouraged me during this project and throughout my master's degree.

First of all, I would like to express the most profound appreciation to my advisor Assoc. Prof. Dr. Özlem Durmaz İncel for her continuous support, motivation and knowledge during my master's study and related research. Her guidance helped me in all the time during my research.

I would like to thank also to my family and my friends for their support.

November 2019

Ceren Nur BAŞ

## TABLE OF CONTENTS

<b>ACKNOWLEDGEMENTS .....</b>	<b>iii</b>
<b>TABLE OF CONTENTS .....</b>	<b>iv</b>
<b>LIST OF SYMBOLS .....</b>	<b>vi</b>
<b>LIST OF FIGURES .....</b>	<b>vii</b>
<b>LIST OF TABLES .....</b>	<b>viii</b>
<b>LIST OF EQUATIONS.....</b>	<b>ix</b>
<b>ABSTRACT.....</b>	<b>x</b>
<b>ÖZET .....</b>	<b>xii</b>
<b>1. INTRODUCTION .....</b>	<b>1</b>
<b>2. LITERATURE REVIEW .....</b>	<b>4</b>
2.1. Industry 4.0 .....	4
2.2. Prognostics and Health Management .....	5
2.3. Similar Studies and RUL Prediction.....	6
<b>3. PHM DATASETS AND PRONOSTIA DATASET.....</b>	<b>8</b>
3.1. Summary of the PHM Datasets .....	8
3.1.1. Turbofan Engine Degradation Dataset .....	8
3.1.2. Femto Bearing Dataset.....	9
3.1.3. IMS Bearing Dataset.....	9
3.1.4. Milling Dataset .....	9
3.2. Previously Used Techniques on PHM Datasets .....	10
<b>4. METHODOLOGY .....</b>	<b>14</b>
4.1. Analyzed Dataset .....	14
4.2. Basic Theory of LSTM .....	15
4.3. Methodology .....	16
4.3.1. Utilized Python Libraries and Tools .....	17
4.3.2. Preprocessing .....	17
4.3.3. Health Indicator Calculation .....	22
4.3.4. Prediction Model.....	26
4.3.5. Experiments .....	27
4.3.5.1. Performance of Deep Network with Different Parameters.....	28
4.3.5.2. Performance of Deep Network with One Bearing.....	30
4.3.5.3. Performance of Deep Network with Different Dataset Clusters .....	31

<b>5. RESULTS .....</b>	<b>32</b>
<b>6. CONCLUSION .....</b>	<b>35</b>
<b>REFERENCES.....</b>	<b>36</b>
<b>APPENDICES.....</b>	<b>42</b>
Appendix A.....	42
Appendix B.....	46
Appendix C.....	48
<b>BIOGRAPHICAL SKETCH.....</b>	<b>50</b>



## LIST OF SYMBOLS

<b>CNN</b>	: Convolutional Neural Network
<b>CSV</b>	: Comma Separated Values
<b>CWT</b>	: Continuous Wavelet Transform
<b>DWT</b>	: Discrete Wavelet Transform
<b>GPR</b>	: Gaussian Process Regression
<b>HI</b>	: Health Indicator
<b>IoT</b>	: Internet of Things
<b>MAE</b>	: Mean Absolute Error
<b>LSTM</b>	: Long-Short Term Memory
<b>PHM</b>	: Prognostics and Health Management
<b>ReLU</b>	: Rectified Linear Unit
<b>RMS</b>	: Root Mean Square
<b>RNN</b>	: Recurrent Neural Network
<b>RUL</b>	: Remaining Useful Life
<b>SD</b>	: Standard Deviation

## LIST OF FIGURES

<b>Figure 4.1:</b> Overview of PRONOSTIA Platform.....	15
<b>Figure 4.2:</b> Structure of RNN and LSTM block.....	16
<b>Figure 4.3:</b> Construction steps of the prediction model.....	17
<b>Figure 4.4:</b> Accelerometer Signal and their DWT smoothed versions.....	18
<b>Figure 4.5:</b> Mean, Variance, and Standard Deviation Features Plotting of the Bearing3_1.....	20
<b>Figure 4.6:</b> Kurtosis, RMS, Skewness and Crest Factor Features Plotting of the Bearing3_1.....	21
<b>Figure 4.7:</b> OLS Regression Results.....	24
<b>Figure 4.8:</b> Health indicator of learning bearings.....	25
<b>Figure 4.9:</b> Health indicator of test bearings.....	26
<b>Figure 4.10:</b> Layers of the constructed network.....	27
<b>Figure 5.1:</b> Mean absolute error values for train and test data.....	32
<b>Figure 5.2:</b> Predicted and actual remaining useful lifetimes of bearings.....	33

## LIST OF TABLES

<b>Table 3.1:</b> Used techniques and goals on Turbofan Engine Degradation Dataset.....	10
<b>Table 3.2:</b> Used techniques and goals on Femto Bearing Dataset.....	11
<b>Table 3.3:</b> Used techniques and goals on IMS Bearing Dataset.....	12
<b>Table 3.4:</b> Used techniques and goals on Milling Dataset.....	13
<b>Table 4.1:</b> Bearing Dataset with operational conditions.....	15
<b>Table 4.2:</b> Time domain feature formulas.....	19
<b>Table 4.3:</b> Correlation values of features with HI.....	23
<b>Table 4.4:</b> Summary of experiments with different LSTM parameters.....	29
<b>Table 4.5:</b> Summary of experiments with only one bearing.....	30
<b>Table 4.6:</b> Summary of experiments with different train and test data clusters.....	31
<b>Table 5.1:</b> Performance comparisons of the proposed method with related researches...	34



## LIST OF EQUATIONS

<b>Equation 4.1:</b> Mean Absolute Error.....	27
<b>Equation 4.2:</b> Percent error of predicted RUL.....	28
<b>Equation 4.3:</b> Error rate for each bearing.....	28
<b>Equation 4.4:</b> Scoring Function.....	28

## **ABSTRACT**

Nowadays, the number of devices connected to the Internet has increased considerably, and this had led to the emergence of the term Internet of Things. Thanks to the Internet of things, devices can now decide and share information between them. Industry 4.0, which has recently become very popular, is a term used for data exchange and automation for production technologies.

The manufacturing industry produces a large amount of data with the help of the many sensors integrated on these devices. These data can be used to improve processes and product quality. For example, by analyzing these data, anomalies that may occur in the lines can be found in advance, the life expectancy of the devices in the factory can be calculated, and as a result, active preventive maintenance can be provided for these devices.

This study aims to find the remaining useful life of the components by performing time series analysis with deep learning methods in the literature. There are many studies on the estimation of the remaining useful life of the components in the literature. In these studies, various methodologies such as artificial neural networks, signal processing, regression have been used.

The quality of the data set has great importance in the creation of estimation models. We examined 4 PHM data sets provided by NASA. In this study, the properties of data, sampling frequencies, and pre-applied methods on data sets are summarized.

Femto-ST Bearing data set has been selected for this study. This dataset has high-frequency noise; therefore, we applied Discrete Wavelet Transform on the data. The features, such as mean, kurtosis, skewness, standard deviation, root mean square, crest

factor, variance, were extracted from this data set. Also, health indicator values were calculated to determine the remaining useful life of the bearings. We construct the remaining useful life prediction model, particularly using the LSTM (long-short-term memory) neural network. We performed various experiments to find the right LSTM parameters and to analyze how the prediction model works with different train and test sets.

Furthermore, we finally compared the results of our model with the results of previous studies on this dataset. Our results are underperformed compared by other studies. We concluded that, this dataset is not directly applicable to the LSTM network, and preprocessing needs more effort.

**Keywords:** Long-Short Term Memory, Remaining Useful Life, Deep Learning, Active Preventive Maintenance

## ÖZET

Günümüzde, internete bağlı cihaz sayısı oldukça artmıştır ve bu da nesnelerin interneti teriminin çıkışını sağlamıştır. Nesnelerin interneti sayesinde cihazlar artık kendileri karar verebilir ve aralarında bilgi paylaşabilir hale gelmiştir. Son zamanlarda çok duyduğumuz Endüstri 4.0 ise üretim teknolojileri için veri alışverişi ve otomasyon için kullanılan bir terimdir.

Üretim sanayisi cihazların birçok sensora sahip olması nedeniyle oldukça büyük miktarda veriye sahiptir. Bu veriler süreçlerin otomasyonu ve iyileştirilmesi için kullanılabilir. Örneğin bu verilerin analiz edilmesi ile üretim hatlarında oluşabilecek anomaliler önceden bulunabilir, fabrikadaki cihazların düzgün olarak çalışabilecekleri yaşam süreleri hesaplanabilir ve bunların sonucunda da bu cihazlar için aktif önleyici bakımlar sağlanabilir.

Bu çalışmanın amacı da derin öğrenme metotlarıyla zaman serisi analizi yaparak bileşenlerin kalan faydalı ömürlerini bulmaktır. Literatüre bakıldığında bileşenlerin kalan faydalı ömürlerinin tahminiyle ilgili birçok çalışma bulunduğu görülebilir. Bu çalışmalarda yapay sinir ağları, sinyal işleme, regresyon ve birçok makine öğrenmesi teknikleri gibi çeşitli metodolojiler kullanılmıştır.

Veri setinin kalitesi tahmin modellerinin oluşturulmasında büyük önem taşımaktadır. Bu çalışmada NASA'nın 4 PHM veri seti incelenmiştir. Bu incelemede verinin özellikleri, toplanma frekansları ve veri setleri üzerinde önceden uygulanmış metotlar özetlenmiştir.

Bu veri setleri arasından Femto Enstitüsü'nün rulman veri seti seçilmiştir. Kullanılan bu veri setinde yüksek frekanslı gürültü görülmektedir. Bu nedenle ivmeölçer verisi üzerinde Ayırık Dalgacık Dönüşümü uygulanmıştır. Bu veri seti üzerinden ortalama,

basıklık, çarpıklık, standart sapma, varyans, ortalama karekk, kret faktr gibi zellikler ıkarılmıřtır. Ayrıca, rulmanların kalan kullanım mrn belirlemek iin saėlık gstergesi deėerleri hesaplanmıřtır. Uzun-kısa sreli bellek(LSTM) sinir aėı kullanılarak kalan faydalı mr tahmin modeli oluřturulmuřtur. Doėru LSTM parametrelerini bulmak ve tahmin modelinin farklı ėrenme ve test verileriyle nasıl alıřtıėını analiz etmek iin eřitli deneyler yaptık.

Son olarak, oluřturduėumuz modelin sonularını, literatrdeki diėer bu veri setini kullanan alıřmaların sonuları ile karřılařtırdık. Modelimiz, literatrdeki diėer alıřmalara gre daha iyi bir performans gsteremedi. Bu veri seti, LSTM aėı iin doėrudan uygulanamaz. n iřleme adımımda olduka aba harcanması gerekmektedir.

**Anahtar Kelimeler:** Uzun Kısa Sreli Bellek, Kalan Faydalı mr, Derin ėrenme, Aktif Koruyucu Bakım

## 1. INTRODUCTION

The current revolution of the Internet and machine-to-machine technologies has led to the emergence of the Internet of Things. The number of devices, “things” connected to the Internet, increases widely (Al-Fuqaha et al., 2015). The IoT makes the objects smart; thus, they can share information and coordinates decisions.

Industry 4.0 is a term for automation and data exchange in manufacturing technologies. Industry 4.0 contains various information technology paradigms such as cyber-physical systems, IoT, cloud computing, and cognitive computing. Nowadays, the manufacturing industry has a large amount of data that can be used to improve processes and product quality.

This large amount of data should be analyzed to extract the useful data, and it can be used for anomaly detection, lifetime estimation, active preventive maintenance or the amount of the used energy in the factory. These are important for the sustainability of factories and automation.

A time series is a series of data points, which is a sequence taken at successive time intervals. Time series analysis is used to predict future values based on previously observed values. Time-series data analysis can be beneficial for factory automation.

The methods to be used in the time series analysis may vary depending on the type of data and the information which is intended to be extracted. For example, simple statistical methods or several machine learning methods such as density-based, clustering-based, and support-vector machine learning techniques are used to predict anomaly detection. If it is intended to make numerical inferences, such as machine lifetime estimation, regression methods are used (Lei et al.).

The active preventive maintenance is one of the most critical parts of smart factories. The purpose of the active preventive maintenance is to trigger required maintenance as early as possible and to reach just-in-time maintenance. Thus, it provides near-zero downtime. Rapid decision making is important in increasing productivity.

In order to improve processes in the manufacturing industry, the produced sensor data can be used. The processing of that big data into significant information is fundamental for sustainable innovation. Due to the lack of smart analytics tool, big data of the manufacturing industry could be analyzed in a limited way, and it prevents the provision of Industry 4.0 term. Therefore, we focus on this topic in order to create a deep learning model for such active preventive maintenance.

***Main Research Question:***

Is it possible to improve processes in industry by analyzing industrial time series data with deep learning approaches?

In order to find answers to our research questions, we started our research with a literature review. Previous studies related to time-series analysis with deep learning approaches and Industry 4.0 are analyzed. The literature review shows that the methods to be used in the time series analysis may vary depending on the type of data and the information intended to be extracted.

After the literature review, we examined the PHM datasets to do experiments with our model. The quality of the dataset is important. The challenging part of this study is to find the dataset to be analyzed. Long-term and smooth data could not be reached easily. The results from machine learning models depend highly on the quality of the dataset. In this study, four PHM (Prognostics and Health management) datasets of NASA were examined.

We decided to perform our time-series analysis on Femto-ST Bearing Dataset, which is also called as PRONOSTIA dataset (Nectoux et al.). Firstly, we preprocessed the data due to high-frequency noise. We extracted features from accelerometer sensors and calculated HI

values, which indicates the condition of bearing. We construct the remaining useful life prediction model, particularly using the LSTM (long-short-term memory) neural network. Various experiments are performed in order to find the right LSTM parameters.

We compared the results of our model with the results of previous studies on this dataset. Our results are performed slightly worse than the other studies. We can say that this dataset is imbalanced and cannot be directly applicable to the LSTM network. Health Indicator values could be calculated more precisely. This thesis summarizes how this subject should be studied and how datasets can affect the prediction models.

This thesis is organized as follows: In Section 2, there is a summary of the literature review for this study. In Section 3, examined datasets and previously used techniques on these datasets are summarized. Section 4 describes the dataset and the steps of the proposed methodology in detail. In Section 5, the results of our experiments are presented and compared with the results of previous studies. Finally, Section 6 concludes this study with some remarks.



## **2. LITERATURE REVIEW**

This chapter provides an overview of the literature review about Industry 4.0, prognostics and health management, and similar studies.

### **2.1. Industry 4.0**

The Internet of Things is pervading rapidly day by day and aiming to improve the quality of life by connecting many smart devices, technologies, and applications. IoT makes the objects smart, so these objects can see, hear, think, and talk and share information among themselves and coordinate decisions. IoT can have important home and business applications to improve quality of our lives (Al-Fuqaha et al., 2015). Automation of everything around us can be realized through the IoT.

The Industry 4.0 era will form new thinking of production management and factory transformation by the teaming of interconnected systems and intelligent analytics. The manufacturing industry has large amount of data that can be used to improve processes and product quality. This may be possible by analyzing data effectively. According to (Lee et al., 2014), advanced prediction tools are needed in order to process data systematically and make more “informed” decisions.

Manufacturing big data consists of device data and product data. This data can be analyzed and used for active preventive maintenance, optimization of a production line, and energy consumption optimization (Jiafu et al., 2017).

## 2.2. Prognostics and Health Management

Prognostics and Health Management is primarily dealing with component wear and degradation. Remaining useful life prediction, fault diagnosis, and fault detection are targeted by PHM algorithms in order to provide factory-wide transparency (Lee et al., 2013). This can be achieved by analyzing sensory and system-level data.

There are many approaches to analyze the manufacturing data in industry and to create models for prognostics, such as the learning-based and signal processing-based approaches. The learning-based approaches are very common in industrial applications because they can learn from data without wide expertise about the process knowledge of the analyzed data. But signal processing-based approaches require the knowledge of certain parameters of the device (Gillespie & Gupta, 2017). For the signal processing, denoising and filtering processes are necessary and important. Feature extraction from time, frequency and time-frequency domains are needed for learning-based approaches. RMS, kurtosis, crest factor, and standard deviation could be given as the examples of these features. In this study, we also extract these time-domain features.

In order to select a suitable technique, problems from similar nature, and previously used techniques to solve these problems need to be analyzed. The technique to be used may vary according to the nature of the available data. For example, neural networks can be used for analyzing manufacturing data to calculate lifetime under specific processing conditions (Jiafu et al., 2017).

Health condition monitoring of machines is a crucial task to guarantee reliability in industrial processes. The quality of the dataset is a critical issue for machine learning models. The real-time data collected from a machine in the field will help to achieve optimal flexibility and robustness for handling different situations (Lee et al., 2014). The machine fleet data can be used to build clusters that represent different machine performances and working conditions based on similarities of the machines performing similar tasks or similar service times.

### 2.3. Similar Studies and RUL Prediction

For the RUL prediction, there are mainly three steps. Data acquisition, calculation of health indicator values, and prediction of failure time. According to Guo et al. (2017), HI values of a different bearing are different at a failure time; Therefore, the determination of a failure threshold is difficult. The success of the RUL prediction mostly depends on the performance of HIs and they focused on the construction of HI. They similarly extracted time-domain features as in our study. But they used these features with the combination of related similarity features while constructing HIs. While our study uses a simple statistical method for calculating HI values, they used the recurrent neural network.

The preprocessing is a crucial part of learning models and it affects their performance. Hong et al. (2014) proposed a preprocessing model. They use a wavelet packet-empirical mode decomposition for feature extraction and Gaussian Process Regression (GPR) for RUL prediction. They also used the PRONOSTIA dataset for their experiments.

Yoo & Baek (2018) presents a similar study. They used the same dataset and deep learning approaches to predict RUL. In this study, a novel time-frequency image is proposed in order to construct HI (Health Indicator) and predict the RUL (Remaining Useful Lifetime). The Convolutional Neural Network (CNN) is used to automatically discover useful features from the raw signal to construct the HI, and the Morlet-based CWT was used to extract image features from the raw vibration signal. CNN is used as a regression model to estimate the CWTCNN-HI based on training images. The HI shows the condition of the machine or component. The RUL is predicted by calculating the difference between the time at which the predicted HI reaches the threshold and the current time. A GPR algorithm has been used to predict the RUL of the bearings. The GPR model predicts future CWTCNN-HI by estimated CWTCNN-HI up to the current time.

Several studies in the literature use deep learning and feature extraction methods to construct HIs and regression methods to create prediction models. Different than the previous approaches, we used the LSTM neural network to create our prediction model

and statistical methods for the calculation of HI values. We used Discrete Wavelet Transform to filter data as different from other studies and extracted time-domain features as common.



### **3. PHM DATASETS AND PRONOSTIA DATASET**

This chapter presents the examination of the PHM datasets and previously used techniques on these datasets. More detailed analysis of the datasets can be found in the paper (Lei et al., 2018).

#### **3.1. Summary of the PHM Datasets**

In this section, details of the four PHM datasets are explained. The selection of the dataset is one of the challenging parts of this study because the results from machine learning models highly depend on the quality of the dataset.

##### **3.1.1. Turbofan Engine Degradation Dataset**

Run-to-failure data of turbofan engines are described in this dataset<sup>1</sup>. The thermodynamical simulation model has been used to create this dataset, and the degradation processes have been simulated as realistic as possible. It contains 4 sets of data, each of which is a combination of 2 failure modes and 2 operating conditions. The dataset parameters are unit number, time, 3 operational settings, and 21 sensor measurements. The dataset is eligible for data-driven approach since sufficient data and RUL values are available with the dataset.

---

<sup>1</sup> <https://ti.arc.nasa.gov/c/6/>

### 3.1.2. Femto Bearing Dataset

This dataset<sup>2</sup> contains 17 run-to-failure data of rolling element bearings acquired from a PRONOSTIA platform. This dataset provides more realistic data because of using a real experimental platform. Vibration signals and the temperature values are the parameters of this dataset. The data is sampled with 10 Hz frequency. Traditional fault diagnostic methods based on frequency analysis is not applicable.

### 3.1.3. IMS Bearing Dataset

This dataset<sup>3</sup> contains run-to-failure data of rolling element bearings. The data contains three sets (each set with four bearings) of tapered rolling element bearings. The vibration data was collected regularly as an indirect health indicator. The real experimental platform is used to create this dataset. One second vibration signal snapshots recorded at specific intervals. The sampling rate is set at 20 kHz. The frequency resolution is about 1 Hz. Therefore, fault diagnosis based on frequency analysis is possible. However, the data is insufficient for data-driven modeling.

### 3.1.4. Milling Dataset

This dataset<sup>4</sup> composed of 16 run-to-failure data which sampled from tool wear experiments of a milling machine. This dataset is the most realistic one among the others. Acoustic emission signals, vibration signals, and current signals were recorded at a frequency of 250 Hz. There are eight different operating conditions leading to only two samples for each operating condition. Counter for experimental runs in each case, flank wear, depth of cut, feed, material, AC spindle motor current, DC spindle motor current, table vibration, spindle vibration, acoustic emission at the table, acoustic emission at spindle are parameters of this dataset.

---

<sup>2</sup> <https://ti.arc.nasa.gov/c/18/>

<sup>3</sup> <https://ti.arc.nasa.gov/c/3/>

<sup>4</sup> <https://ti.arc.nasa.gov/c/4/>

### 3.2. Previously Used Techniques on PHM Datasets

In this section, we focus on used methods and their objectives. This summary helped us to decide which method to apply differently from previous studies and what we have to focus on an objective. Therefore, this step is important for this study. Table 3.1 summarizes the used techniques and goals on Turbofan Engine Degradation Dataset. Previous studies mainly focused on remaining useful lifetime on this dataset.

Table 3.1: Used techniques and goals on Turbofan Engine Degradation Dataset

Techniques	Goal
Artificial Intelligence (El-Koujok et al., 2011; Peng et al., 2012; Xi et al., 2014; Xu et al., 2014)	RUL
Recurrent Neural Networks (Heimes, 2008)	RUL
Multi feature fusion for developing composite health indices (Liu et al., 2013, Liu et al., 2017)	Developing Composite Health Indices
A genetic fuzzy rule-based system (Ishibashi & Júnior, 2013)	RUL
Similarity Based Prognostic approach (Wang et al., 2008)	RUL
Bayesian Approaches (Mosallam et al., 2016)	RUL
Hidden Markov Model (Romasso & Denoeux, 2014)	Health Estimation
Neuro-Fuzzy System (Romasso & Gouriveau, 2014)	RUL
Support Vector Regression (Khelif et al., 2017)	RUL

In Table 3.2, it can be seen that Femto Bearing Dataset has been analyzed for various goals such as RUL, condition monitoring, and constructing health indicators.

Table 3.2: Used techniques and goals on Femto Bearing Dataset

Techniques	Goal
Signal Processing	Constructing Health Indicator(HI)
Artificial Intelligence	Constructing Health Indicator(HI)
Multi feature fusion and nonlinear dimension reduction (Guo et al., 2017)	Condition Monitoring
E-support vectors regression (Loutas et al., 2013)	RUL
Extended Kalman Filtering (Singleton et al., 2015)	RUL
Monotonic Score Calibration (Carino et al., 2015)	RUL
Signal Complexity and Gaussian process models (Boškoski et al., 2012)	Fault Prognostics
Sparse Representation model (Ren et al., 2015)	RUL
Distributed Neuro-Fuzzy System Feature Forecasting (Zurita et al., 2014)	Condition Monitoring
Proportional Hazard Model (Wang et al., 2015)	RUL
Support Vector Regression (Benkedjough et al., 2013)	RUL
Hilbert-Huang Transform, Support Vector Machine, Regression (Soualhi et al., 2015)	Health Monitoring



Table 3.3 shows that there are few studies for prediction of remaining useful lifetime by using IMS Bearing Dataset and previous studies mainly focused on degradation assessment and bearing failures.

Table 3.3: Used techniques and goals on IMS Bearing Dataset

Techniques	Goal
Wavelet filter based weak signature method (Qui et al., 2006)	Bearing prognostics
Empirical Mode Decomposition and Artificial Neural Network (Ben Ali et al., 2015)	Fault Diagnosis
Locality preserving projections and Gaussian Mixture Models (Yu, 2011)	Degradation Assessment
PCA and optimized LS-SVM Model (Dong & Luo, 2013)	Degradation Prediction
Wavelet filter based method and Self Organizing Map (SOM) (Qui et al. 2003)	Performance Degradation Assessment
Hidden Markov Model and Adaptive Neuro-Fuzzy inference system (Soualhi et al. 2014)	Prognosis of Bearing Failures
Weibull Distribution and Artificial Neural Network (Ali et al., 2015; Mohammad et al., 2010)	RUL
Proportional Hazard Model and Logistic Regression (Liao et al., 2006)	RUL
Relevance Vector Machine(RVM) (Widodo & Yang, 2011)	Prediction of Survival Probability of Bearing

We can see that there are few studies on Milling dataset in Table 3.4. Various goals have been targeted for this dataset. As mentioned before, the quality of the dataset has an important impact on the success of machine learning models.

Table 3.4: Used techniques and goals on Milling Dataset

Techniques	Goal
Relevance Vector Machine for Estimating Tool Wear and Recursive Least Square Algorithm (Zhang, 2011)	RUL
Distributed Fusion Filtering from Multiple Sensors (Wei et al., 2013)	Degradation Process Identification
Adaptive Gaussian Mixture Model (Yu, 2012)	Machine Tool Condition Monitoring
S-transform and Genetic Algorithm (Rad et al., 2014)	Extracting features for tool condition monitoring
The General Path Model(GPM) (Coble & Hines, 2014)	Degradation Based Diagnostics

As a result of this examination, we decided to use Femto Bearing Dataset and LSTM. We chose this dataset because it was created in a realistic experimental environment, the bearings were degraded naturally, and the deep learning approaches have not been applied widely to predict RUL on this dataset. In this study, our aim was using deep learning approaches. LSTM networks are a kind of RNN which are well designed with complex blocks to avoid vanishing gradients problem. LSTMs are applied to solve a large variety of problems recently. LSTM has not been applied to this data set before and we intend to create our model with a deep learning approach. Therefore, we chose LSTM for this study.

## 4. METHODOLOGY

This chapter provides a comprehensive explanation of the analyzed dataset, the theory of LSTM, and expression of our prediction model. In this thesis, our aim is predicting the remaining useful life of bearings accurately, by using the LSTM network. We also intended to show which method is better by comparing our results with the results of previous studies on this dataset.

### 4.1. Analyzed Dataset

As mentioned in the previous section, this dataset contains 17 run-to-failure data of rolling element bearings acquired from a PRONOSTIA platform. The overview of the PRONOSTIA platform is shown in Figure 4.1. This dataset is constructed under 3 different operating conditions as follows:

- First operating conditions: 1800 rpm and 4000 N,
- Second operating conditions: 1650 rpm and 4200 N,
- Third operating conditions: 1500 rpm and 5000 N.

Table 4.1 shows the distribution of bearings under these 3 operating conditions.

The characterization of the bearing's degradation is based on two data types of sensors: vibration and temperature. The vibration sensors consist of two miniature accelerometers positioned on the vertical and the horizontal axis. The acceleration measures are sampled every 10 s for a sample period of 0.1 s at 25.6 kHz frequency, and the temperature ones are sampled at 10 Hz. We did not use the temperature measurements while constructing our prediction model.

The bearings were not naturally degraded; therefore, each bearing's degradation pattern is different from each other.

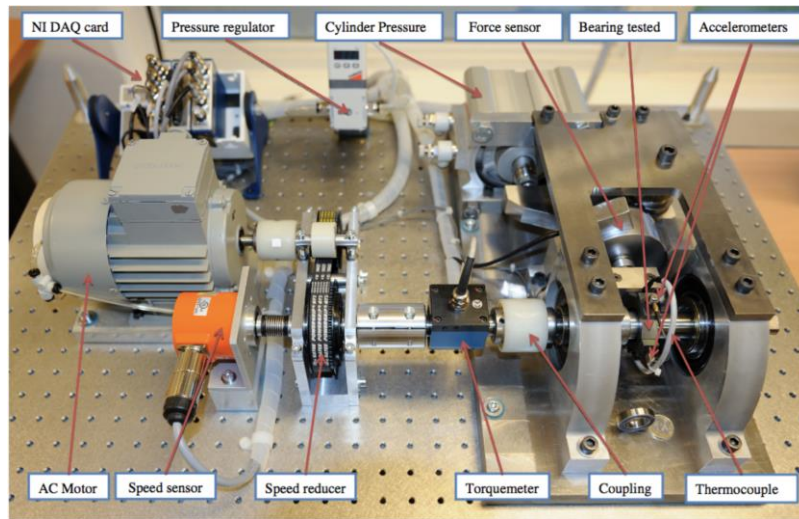


Figure 4.1: Overview of PRONOSTIA platform

Table 4.1: Bearing dataset with operational conditions.

Datasets	Condition1	Condition2	Condition3
Training Datasets	Bearing1_1	Bearing2_1	Bearing3_1
	Bearing1_2	Bearing2_2	Bearing3_2
Test Datasets	Bearing1_3	Bearing2_3	Bearing3_3
	Bearing1_4	Bearing2_4	
	Bearing1_5	Bearing2_5	
	Bearing1_6	Bearing2_6	
	Bearing1_7	Bearing2_7	

## 4.2. Basic Theory of LSTM

Long short-term memory is an artificial recurrent neural network which solves the vanishing gradients problems. In a basic implementation of LSTM, the hidden layer is replaced by a complex block. This complex block is composed of gates that trap the error in the block (Gamboa, 2017). Figure 4.2 shows the layers of the recurrent neural networks and the complex structure of the LSTM block. As you can see in Figure 4.2, an LSTM layer consists of an input gate, a cell, a forget gate and an output gate.

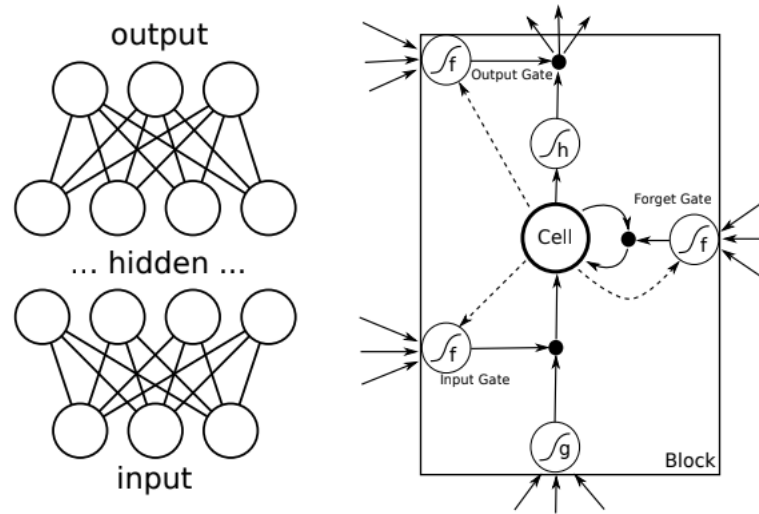


Figure 4.2: Structure of RNN and LSTM block

The cell state flow down through the whole chain, with just a little direct interaction. It acts as a kind of similar structure to a conveyor belt. Information could not be affected and remain unchanged during flow. The information can be added or removed by the LSTM to the cell state. In the meantime, gates regulate these processes and take decisions about letting information through. These gates are combination of a sigmoid neural net layer and a pointwise multiplication operation. The sigmoid layer outputs numbers between 0 and 1. How much of each component should be let through is described by this layer.

### 4.3. Methodology

The main goal of this thesis is to create a prediction model for the remaining useful life of degraded bearings. The steps of the building prediction model are described in Figure 4.3. In first step, we acquired PRONOSTIA dataset. As a second step we preprocessed this dataset. In this step we merged accelerometer files for each bearing and removed redundant fields from dataset. We also applied discrete wavelet transform and extracted features. In third step, Health Indicator values are calculated using statistical methods. As a final step we created our prediction model with LSTM network.

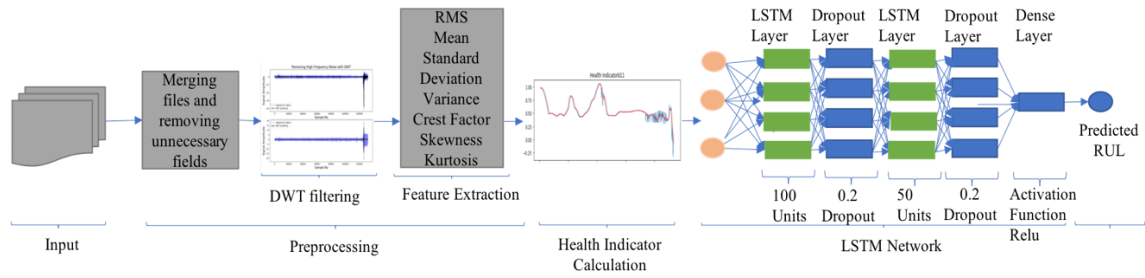


Figure 4.3: Construction steps of the prediction model

Normally, we do not need to calculate features for deep learning algorithms. PRONOSTIA dataset only consists of accelerometer data and each bearing has different degradation pattern. Therefore, we extracted features and calculated health indicator values.

We used python and tensor flow library to build our prediction model. Google Collaboratory Framework is used to run created scripts with python.

#### 4.3.1. Utilized Python Libraries and Tools

We used python while developing our prediction model. TensorFlow<sup>5</sup>, Keras<sup>6</sup>, Sklearn<sup>7</sup> and Pandas<sup>8</sup> libraries are used to preprocess the data and build our prediction model. Google Colab<sup>9</sup> Framework is used to run created scripts with python. It is an environment which allows to write and execute code, providing powerful computing resources.

#### 4.3.2. Preprocessing

Firstly, we preprocessed the raw data by removing unnecessary fields for analysis. There are many CSV files separately for each sampling of bearing accelerometer sensors. We

<sup>5</sup> <https://www.tensorflow.org>

<sup>6</sup> <https://keras.io/>

<sup>7</sup> <https://scikit-learn.org/>

<sup>8</sup> <https://pandas.pydata.org/>

<sup>9</sup> <https://colab.research.google.com>

combined these CSV files, so there is one file for each bearing. We did not use the temperature data since it is not available for all bearings.

This dataset has high-frequency noise; therefore, we needed to smoothen the data. Discrete Wavelet Transform allows us to deconstruct a signal into the low pass and high pass coefficients. High pass coefficients represent the high-frequency part of the signal. DWT filters out the high-frequency noise according to the given threshold. We applied Discrete Wavelet Transform on the horizontal and vertical accelerometer data. Figure 4.4 shows the smoothed accelerometer signals.

We extract many features such as mean, standard deviation, crest factor, variance, skewness, root mean square and kurtosis. While extracting these features 1-minute samples are used. Time domain feature extraction formulas are shown in Table 4.2.

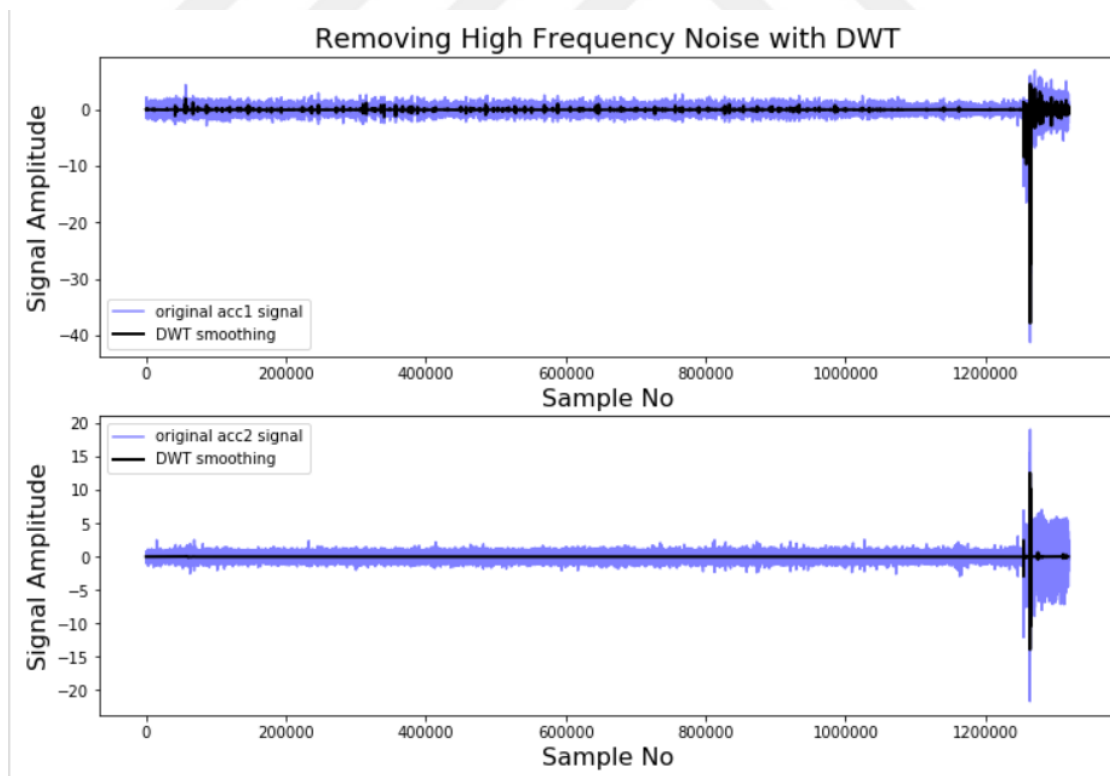


Figure 4.4: Accelerometer signals and their DWT smoothed versions

We also calculated the actual RULs. Then duplicate lines are removed and all learning datasets are merged into one data frame. We plotted the features throughout the lifetime of the bearings. Figure 4.5 and Figure 4.6 show the plotting of features of Bearing3\_1. These figures showed us which features are more meaningful and helped to select features for Health Indicator calculation. Preprocessing codes can be examined in Appendix A.

Table 4.2: Time domain feature formulas.

Name	Formula
Mean	$x_m = \frac{\sum_{n=1}^N x(n)}{N}$
SD	$x_{std} = \sqrt{\frac{\sum_{n=1}^N (x(n) - x_m)^2}{N - 1}}$
Crest Factor	$x_c = \frac{\max x(n) }{x_{rms}}$
Variance	$x_v = \frac{\sum_{n=1}^N (x(n) - x_m)^2}{N - 1}$
Skewness	$x_{ske} = \frac{\sum_{n=1}^N (x(n) - x_m)^3}{(N - 1)x_{std}^3}$
RMS	$x_{rms} = \sqrt{\frac{\sum_{n=1}^N x(n)^2}{N}}$
Kurtosis	$x_k = \frac{\sum_{n=1}^N (x(n) - x_m)^4}{(N - 1)x_{std}^4}$



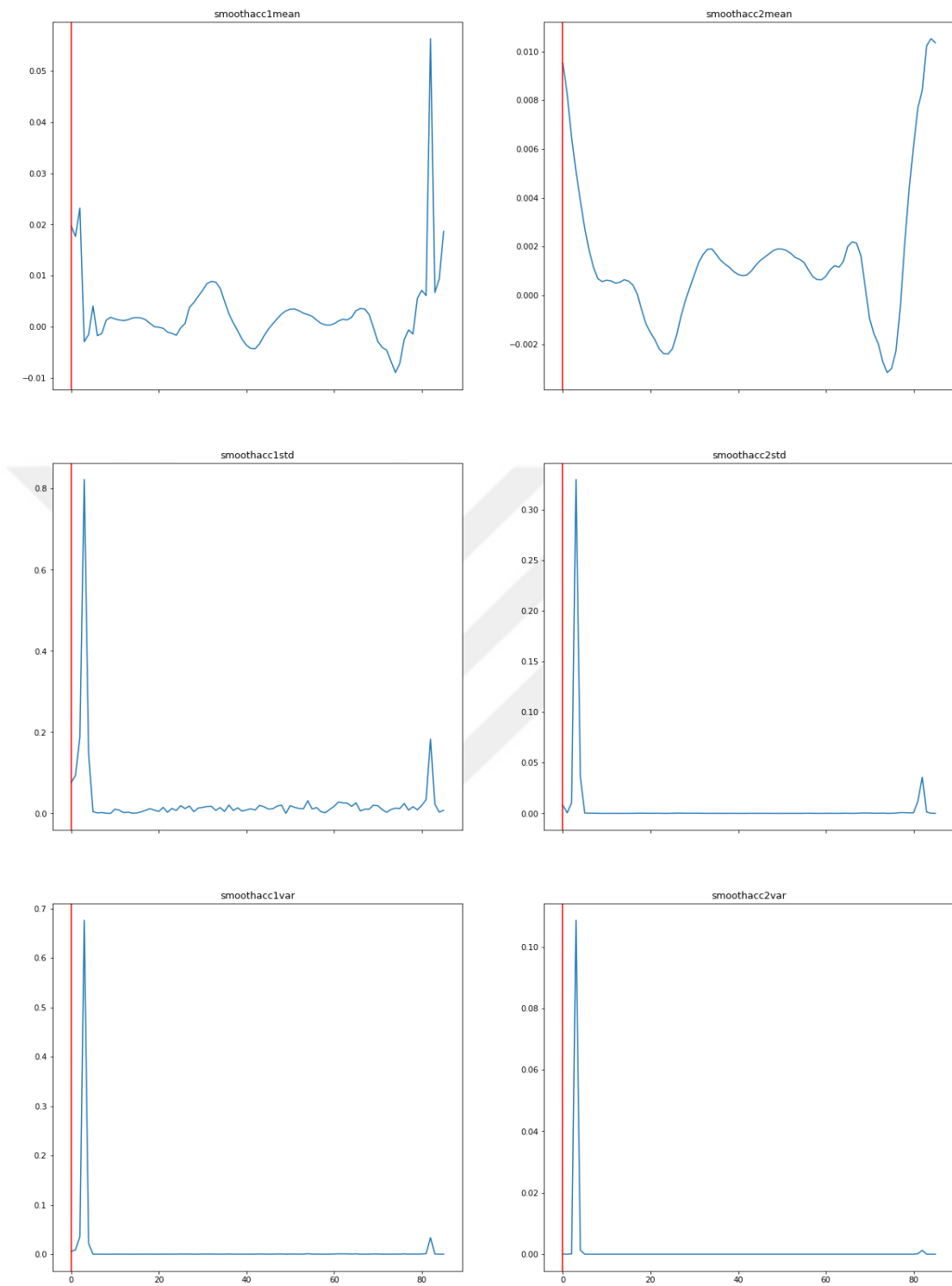


Figure 4.5: Mean, Variance and Standard Deviation Feature Plottings of Bearing3\_1

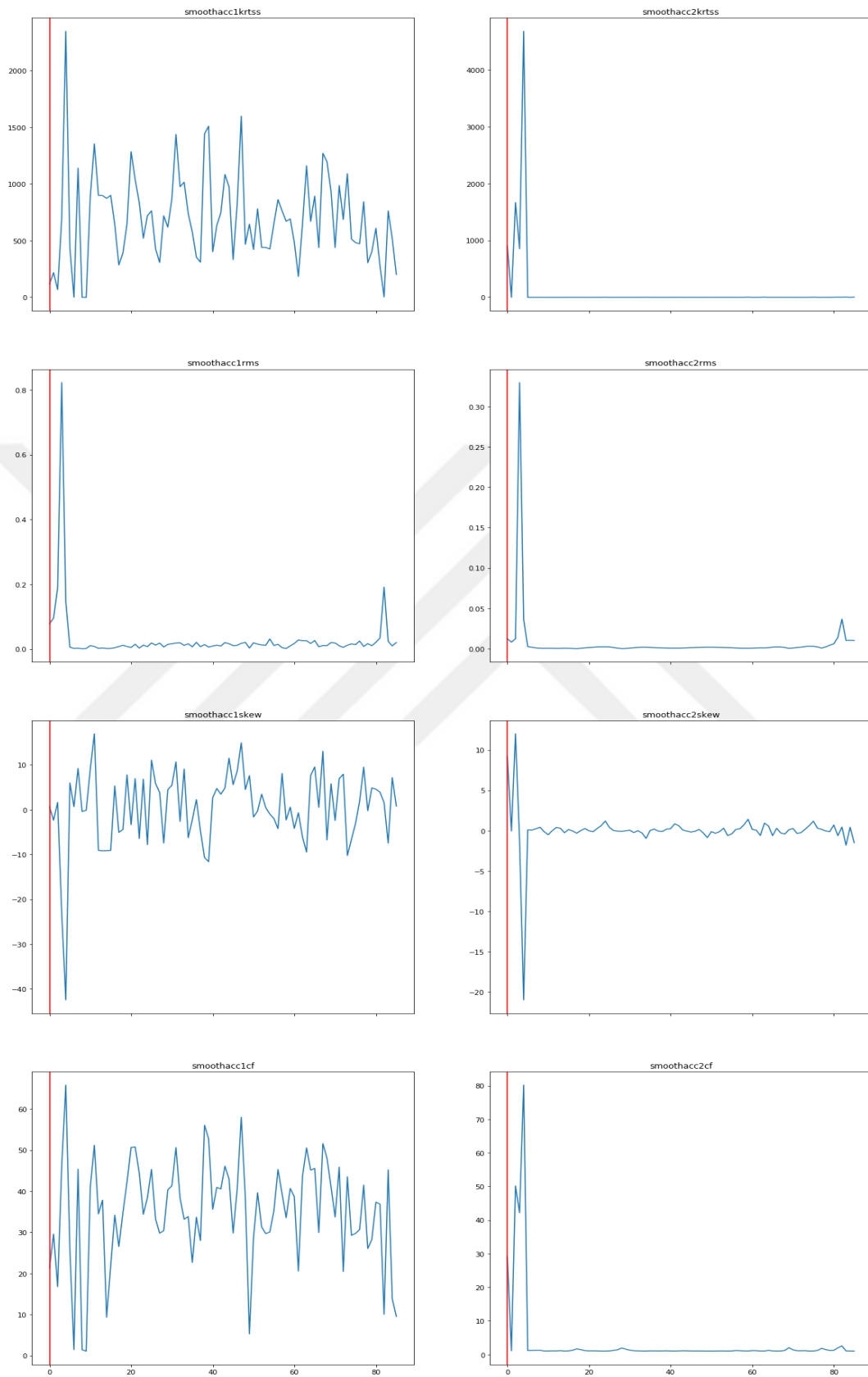


Figure 4.6: Kurtosis, RMS, Skewness and Crest Factor Feature Plottings of Bearing3\_1

According to Figure 4.5 we can say mean of acc1 and acc2 shows rising trend after degradation started. When we look at Figure 4.5 and Figure 4.6, the calculated features, rms, variance, standard deviation of acc1 and acc2, there are peaks when degradation is started. These are important features for Bearing 3\_1. We had to select features for all bearings to create a better model because the dataset has very variant degradation patterns for each bearing.

### **4.3.3. Health Indicator Calculation**

After feature extraction, we calculated health indicator for the bearings. Firstly, we used first 15 minutes as 1 and last 15 minutes as 0 for the health indicator values. 1 indicates a healthy state and 0 indicates a degraded state. Then we calculated remaining health indicator values according to correlation of features with the health indicator.

Table 4.3 presents the correlation coefficient between the features and HI. None of the sensors have a remarkable high correlation between the features and HI. We decided to use features which have correlation coefficients greater than 0.2 to calculate other HI values. Here, we used simple ordinary least square(ols) model from statsmodels which is a python library. The results of this calculation are shown in Figure 4.7. These results can be used to improve Health Indicator calculation.

Then, we calculated health indicators for learning and test data frames. Related codes can be found in Appendix B. Figure 4.8 shows the change of health indicator values of bearings through lifetime and it presents a degradation pattern.

Table 4.3: Correlation values of features with HI

FEATURE	HI
Acc1 Mean	-0.077
Acc2 Mean	-0.279
Acc1 STD	-0.287
Acc2 STD	-0.221
Acc1 Var	-0.194
Acc2 Var	-0.125
Acc1 Kurtosis	-0.099
Acc2 Kurtosis	-0.204
Acc1 RMS	-0.287
Acc2 RMS	-0.220
Acc1 Skew	0.138
Acc2 Skew	0.054
Acc1 Crest Factor	-0.303
Acc2 Crest Factor	-0.318
HI	1.00

```

=====
                        OLS Regression Results
=====
Dep. Variable:          HI      R-squared:                0.319
Model:                  OLS      Adj. R-squared:           0.288
Method:                 Least Squares      F-statistic:              10.35
Date:                   Tue, 08 Oct 2019    Prob (F-statistic):       7.61e-12
Time:                   10:14:07          Log-Likelihood:           -99.205
No. Observations:      186      AIC:                      216.4
Df Residuals:          177      BIC:                      245.4
Df Model:               8
Covariance Type:       nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	0.8980	0.081	11.143	0.000	0.739	1.057
smoothacc2mean	-1.1147	0.204	-5.454	0.000	-1.518	-0.711
smoothacc1std	-104.5013	72.682	-1.438	0.152	-247.935	38.933
smoothacc2std	-96.0282	42.337	-2.268	0.025	-179.579	-12.477
smoothacc2krtss	0.7832	0.487	1.607	0.110	-0.178	1.745
smoothacc1rms	103.6101	72.707	1.425	0.156	-39.874	247.094
smoothacc2rms	96.3940	42.394	2.274	0.024	12.731	180.057
smoothacc1cf	-0.2609	0.169	-1.546	0.124	-0.594	0.072
smoothacc2cf	-0.6821	0.409	-1.667	0.097	-1.490	0.125

```

=====
Omnibus:                49.553      Durbin-Watson:            0.309
Prob(Omnibus):          0.000      Jarque-Bera (JB):         9.963
Skew:                   0.144      Prob(JB):                 0.00686
Kurtosis:               1.903      Cond. No.                 3.79e+03
=====

```

Figure 4.7: OLS regression results

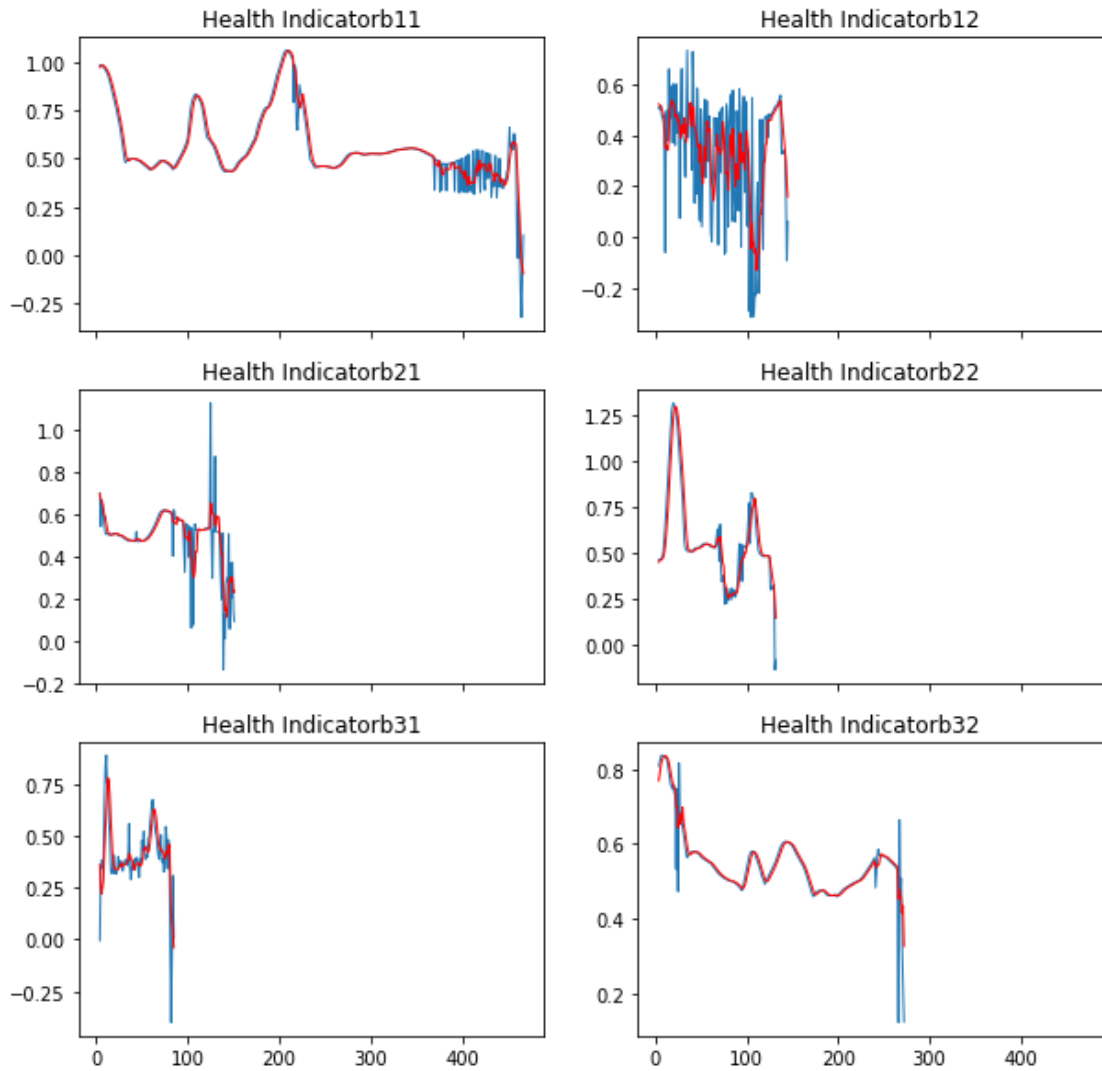


Figure 4.8: Health indicators of learning bearings

In Figure 4.9, we can see that the health indicators did not fit so well. This part needs to be improved in order to increase the prediction model success.

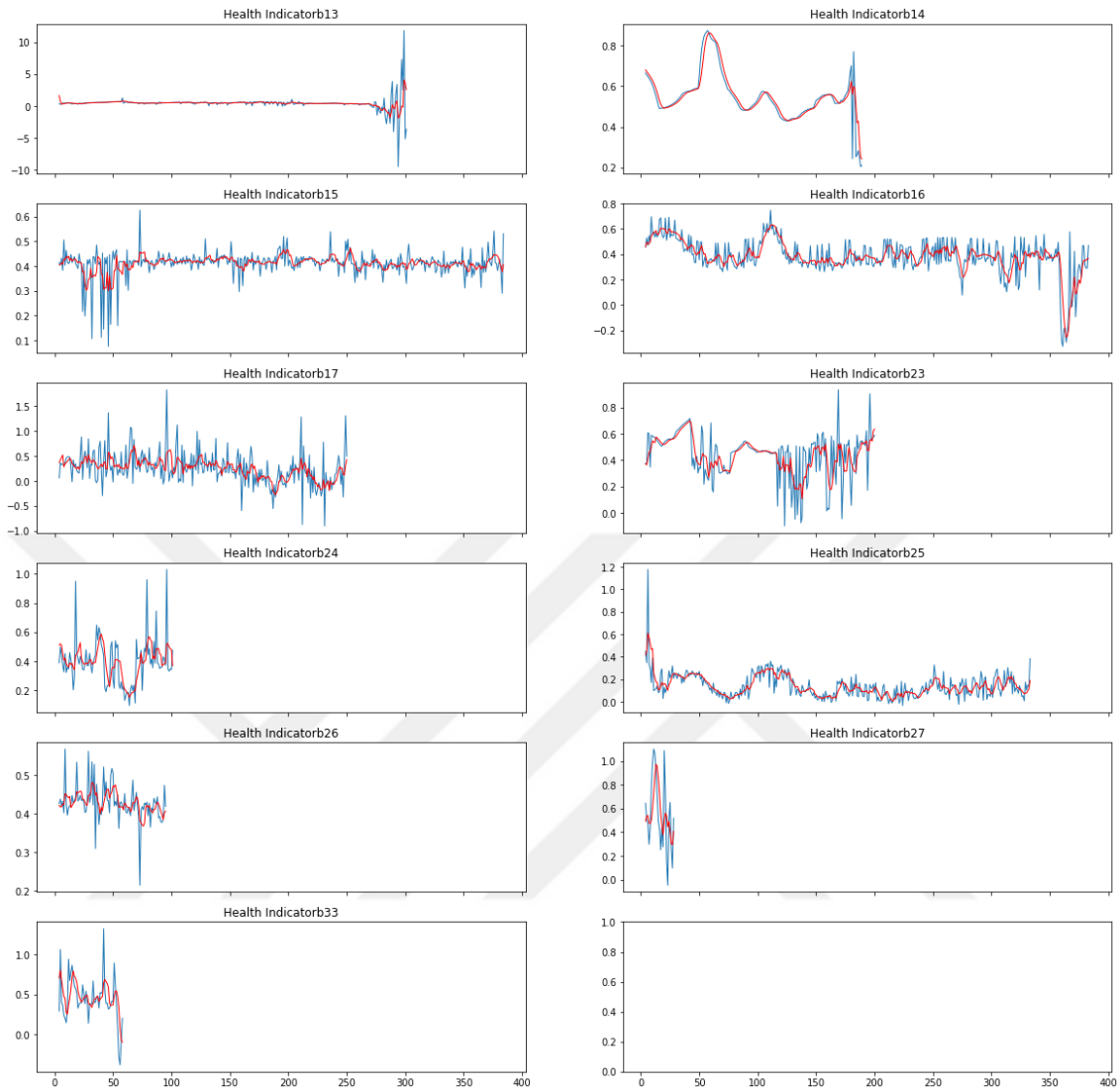


Figure 4.9: Health indicators of test bearings

#### 4.3.4. Prediction Model

As a last step, we construct the deep network. The first layer is an LSTM layer with 100 units and another LSTM layer with 50 units follows this layer. The number of units of LSTM layers were just initial values, then we performed experiments to find better values. After each LSTM layer, we applied dropout to control overfitting. Dropout value is set a commonly used value of 0.2. And then we add a dense output layer with a single unit and

ReLU activation since this is a regression problem. Figure 4.10 shows the layers of the constructed network.

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 25, 100)	46800
dropout_1 (Dropout)	(None, 25, 100)	0
lstm_2 (LSTM)	(None, 50)	30200
dropout_2 (Dropout)	(None, 50)	0
dense_1 (Dense)	(None, 1)	51
activation_1 (Activation)	(None, 1)	0
Total params: 77,051		
Trainable params: 77,051		
Non-trainable params: 0		
None		

Figure 4.10: Layers of the constructed network

#### 4.3.5. Experiments

After we obtained our first results, we did different experiments under three different settings and examined the performance results. We changed the parameters of LSTM, creating the LSTM network with one bearing dataset and, created different train and test data clusters. Then we compared the results.

Mean Absolute Error is used as the performance metric. Formula of the mean absolute error is given in Equation 4.1. Related codes are added in Appendix C.

$$MAE = \frac{1}{n} \sum_{i=1}^n |x_i - x|$$

Equation 4.1: Mean Absolute Error



Scoring function is also used to compare performances. Percent error function is defined in Equation 4.2 which is used for calculating error for each bearing:

$$Er = 100 \times \frac{ActRUL - PredictedRUL}{ActRUL}$$

Equation 4.2: Percent error of predicted RUL

And score is calculated, using Equation 4.3 and Equation 4.4:

$$Ai = \begin{cases} e^{(-\ln(0.5) \cdot (\frac{Eri}{5}))} & \text{if } Eri \leq 0 \\ e^{(+\ln(0.5) \cdot (\frac{Eri}{20}))} & \text{if } Eri > 0 \end{cases}$$

Equation 4.3: Error for each bearing

$$Score = \frac{1}{11} \sum_{i=1}^{11} Ai$$

Equation 4.4: Scoring Function

#### 4.3.5.1. Performance of Deep Network with Different Parameters

In this experiment, we aimed to find the best parameters for our model. Table 4.4 summarizes the results of this experiment. We used 4 different batch size and 4 different learning models with different hidden layers. We did not change the activation function or the sequence length. We used 0.2 Dropout after each LSTM layer and “ReLU” activation for each case. We examined both MAE(Train and Test) and Score values, in order to decide on the LSTM parameters.

When Table 4.4 is analyzed, we see that generally the LSTM units affect the performance of the model. When we compare the Case 5 and Case 8, we see that the score increased and MA decreased as the LSTM units increased. The results also shows that batch size moderately affects the results. Finally, as a result of this experiment we decided to use the parameters of Case 12.

Table 4.4: Summary of experiments with different LSTM parameters

Test Case	Batch Size	Epoch Size	Hidden Layers	MAE during learning	MAE (Train and Test)	Score
Case1	10	200	10-5	151.99	90.55	0.11
				102.45	53.62	
Case2	10	136	25-10	151.27	89.92	0.15
				100.64	60.14	
Case3	10	62	50-25	147.68	90.60	0.15
				101.06	59.98	
Case4	10	41	100-50	142.88	86.65	0.17
				93.77	58.68	
Case5	25	200	10-5	152.14	106.49	0.06
				120.99	35.14	
Case6	25	200	25-10	151.62	93.13	0.07
				105.59	47.33	
Case7	25	129	50-25	149.89	89.94	0.15
				100.45	59.92	
Case8	25	68	100-50	145.59	89.76	0.15
				94.64	52.09	
Case9	50	200	10-5	152.20	118.24	0.24
				134.15	33.25	
Case10	50	200	25-10	152.05	105.65	0.06
				120.74	35.54	
Case11	50	200	50-25	151.02	90.19	0.12
				100.98	55.13	
Case12	50	121	100-50	147.66	85.93	0.23
				94.60	46.67	
Case13	100	200	10-5	152.23	125.98	0.12
				142.62	37.99	
Case14	100	200	25-10	152.15	116.98	0.28
				133.15	32.56	
Case15	100	200	50-25	151.63	101.03	0.10
				114.69	37.96	
Case16	100	200	100-50	149.72	87.86	0.14
				97.29	49.63	

We could use GridSearchCV from the Sklearn python library for hyperparameter tuning. But it would be similar to this experiment. This experiment could be extended by changing the activation function or value of the dropout.

#### 4.3.5.2. Performance of Deep Network with One Bearing

In this experiment, we used parameters of the best result which we described in Table 4.4. Instead of combining all the data from all bearings, we performed experiments with only one bearing at a time. Our experiments show that: as the number of samples increased, performance of the learning model decreased. In Table 4.5, when we examine Case 5, we see that MAE is 19.18. Case 5 shows the performance compared to the merged dataset. But when we examine Case 1, it shows rather poor performance. We can say that, the dataset is still noisy. Therefore, the learning becomes difficult as the number of samples increases.

Table 4.5: Summary of experiments with only one bearing

Test Case	Bearing	Epoch Size	Train Size	Validate Size	MAE During Learning Process	MAE
Case 1	Bearing1_1	168	307	132	282.74 215.97	160.97
Case 2	Bearing1_2	86	81	36	75.28 57.66	42.65
Case 3	Bearing2_1	92	86	37	78.77 60.58	45.36
Case 4	Bearing2_2	68	72	32	66.89 51.08	37.76
Case 5	Bearing3_1	29	39	18	36.76 26.08	19.18
Case 6	Bearing3_2	143	170	74	157.07 120.24	89.15

### 4.3.5.3. Performance of Deep Network with Different Dataset Clusters

In this experiment, we created different Train and Test data clusters. Train clusters are created according to conditions. And we tested this train clusters with all test bearings and test clusters which are created according to the conditions as described in Table 4.1. Case 3 shows the best performance as summarized in Table 4.6.

Table 4.6: Summary of experiments with different train and test data clusters

Test Case	Train Bearings	Test Bearings	MAE During Learning Process	MAE	Score
Case 1	Bearing1_1	Bearing1_3	241.26	142.96	0.059
	Bearing1_2	Bearing1_4	192.30	42.81	
		Bearing1_5			
		Bearing1_6			
		Bearing1_7			
Case 2	Bearing2_1	Bearing2_3	65.21	28.32	0.06
	Bearing2_2	Bearing2_4	34.72	38.65	
		Bearing2_5			
		Bearing2_6			
		Bearing2_7			
Case 3	Bearing1_1	All Bearings	241.26	142.96	0.11
	Bearing1_2		192.30	38.2	
Case 4	Bearing2_1	All Bearings	65.21	28.32	0.04
	Bearing2_2		34.72	46.58	
Case 5	Bearing3_1	All Bearings	127.58	71.61	0.09
	Bearing3_2		92.95	37.75	

## 5. RESULTS

This chapter presents the comparison of the results of this study with previous studies. We chose the prediction model which is created with the parameters of Case 12 for comparison. For the selected LSTM network, the batch size is 50 and epoch size is 200. We used a merged dataset. This dataset contains information for each bearing. Mean absolute error is used as performance metric of the model. In the first epoch, mean absolute error is calculated as 147.66 in train dataset. And through the fitting process of model it reduced to 94.6059. Fitting process lasted 121 epochs. In the figure 5.1, can be seen how mean absolute error changed, it is reduced from about 89 to about 67 on test.

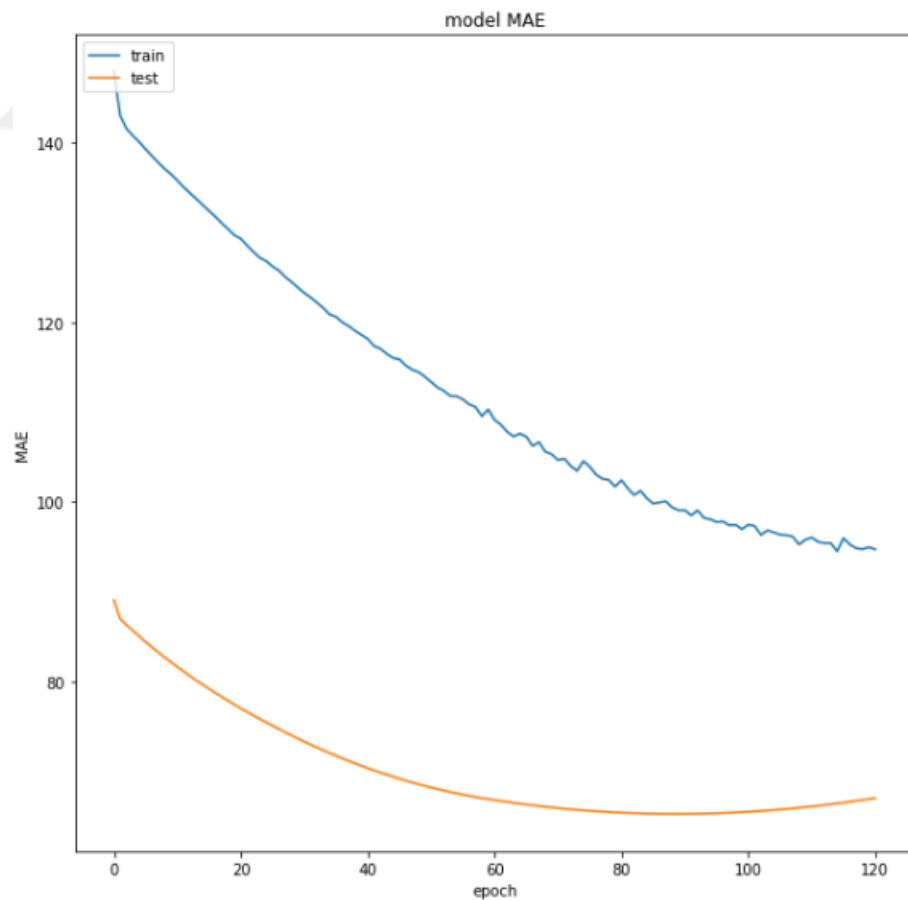


Figure 5.1: Mean absolute error values for train and test data

In Figure 5.2, the actual and predicted values of the test bearings can be seen. Mean absolute error on the test data is calculated as 46.67.

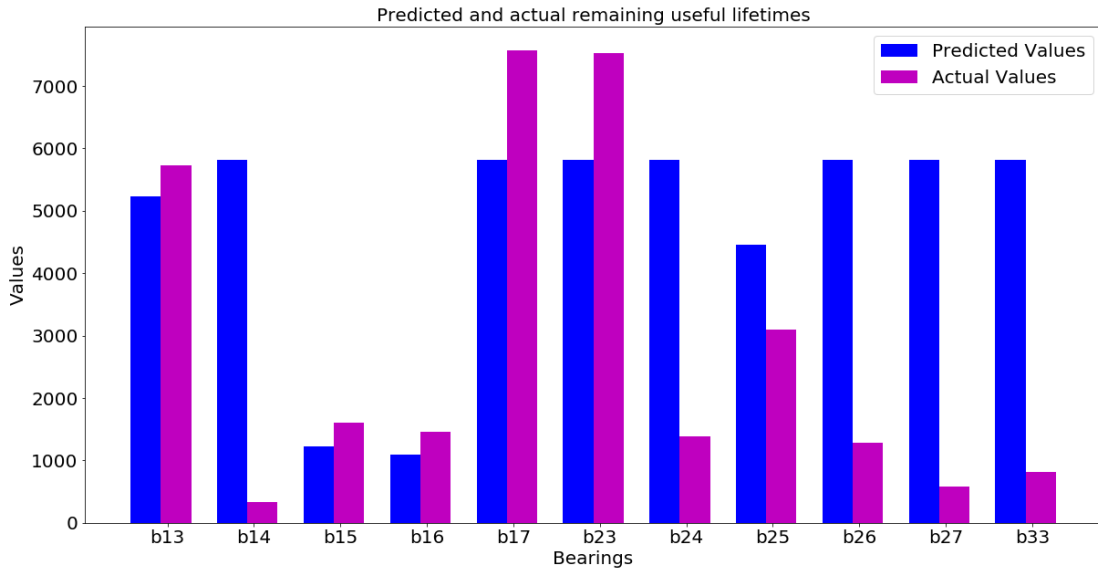


Figure 5.2: Predicted and actual remaining useful lifetimes of test bearings

We use scoring percent errors and scoring function as performance metrics to compare our results with the previous studies which are described respectively as Equation 4.2 and Equation 4.4 in previous chapter.

In Table 5.1, the comparison of our prediction model results with previous studies can be seen. When we compared our results, even if we have score as 0.23, mean and standard deviation values of errors are higher than the other studies.

A common problem in RUL prediction is that the predicted value remains the same or changes within a certain range. The prediction model exceeded 5817 value and the predicted value stayed the same. The merged dataset contains different bearings which shows different degradation patterns and have different life times. Therefore, same value of HI can mean different health condition for each bearing. Therefore, it causes a large variance of the predicted RUL values.

The results of our model need improvement. These results are not just related with our model. Each bearing has different degradation pattern and we do not know the reason of degradation. The lifetime of bearings varies between 30 minutes to 7 hours. We can say that dataset is imbalanced, and the inputs are not sufficient for output. An LSTM network did not perform well on this merged dataset.

Table 5.1: Performance comparisons of the proposed method with related researches

Testing Dataset	Total Time(s)	Actual RUL(s)	Predicted RUL(s)	Guo et al.	Hong et al.	Yoe and Baek	Proposed Method
Bearing1_3	18010	5730	5231	43.28	-1.04	1.05	8.22
Bearing1_4	11380	339	5817	67.55	-20.9	20.35	-1839
Bearing1_5	23010	1610	1223	-22.9	-278	11.18	21.56
Bearing1_6	23010	1460	1090	21.23	19.18	34.93	24.29
Bearing1_7	15010	7570	5816	17.83	-7.13	29.19	23.06
Bearing2_3	12010	7530	5817	37.84	10.49	57.24	22.43
Bearing2_4	6110	1390	5817	-19.4	51.8	-1.44	-321.55
Bearing2_5	20010	3090	4456	54.37	28.8	-0.65	-45.63
Bearing2_6	5710	1290	5817	-13.9	-20.9	-42.64	-361.70
Bearing2_7	1710	580	5817	-55.1	44.83	8.62	-977.30
Bearing3_3	3510	820	5817	3.66	-3.66	-1.22	-645.82
<b>Mean</b>				32.48	44.28	18.96	-371.94
<b>SD</b>				37.57	90.29	25.59	561.73
<b>Score</b>				0.26	0.36	0.57	0.23

## 6. CONCLUSION

In order to realize the term Industry 4.0 which is related with factory automation and sustainability, lack of smart analytic tools should be removed. The big data collected from various sensors can be used for anomaly detection, life time estimation and active preventive maintenance.

We examined 4 PHM datasets of NASA in order to decide which dataset will be used to analyze for active preventive maintenance. The previous methods used on these datasets and their goals are also summarized in tables. So, we can easily focus on which methods we can use as distinct from previous studies.

In this thesis, Femto-ST Bearing Dataset is used for time series analysis and we constructed a prediction model for remaining useful lifetime of degraded bearings. While we are constructing our model, we used LSTM network. Time domain feature extraction is also applied on dataset before construct the model.

Finally, we compared our prediction model results with the previous studies, and we can say that our prediction model needs to be improved. To create a better model, we have to focus on health indicator calculation and LSTM network layers can be differently applied. Deep learning could be used to health indicator calculation.



## REFERENCES

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols and Applications. *IEEE Communication Surveys & Tutorials*, Vol. 17, No.4, pp. 2347-2376.

Ali, J.B., Chebel-Morello, B., Saidi, L., Malinowski, S., Fnaiech, F. (2015). Accurate bearing remaining useful life prediction based on Weibull distribution and artificial neural network. *Mech. Syst. Signal Process.* 56, 150–172.

Ben Ali, J., Fnaiech, N., Saidi, L., Chebel-Morello, B., Fnaiech, F. (2015). Application of empirical mode decomposition and artificial neural network for automatic bearing fault diagnosis based on vibration signals. *Appl. Acoust.* 89, 16– 27.

Benkedjouh, T., Medjaher, K., Zerhouni, N., Rechak, S. (2013). Remaining useful life estimation based on nonlinear feature reduction and support vector regression. *Eng. Appl. Artif. Intell.* 26, 1751–1760.

Boškoski, P., Gašperin, M., Petelin, D. Bearing fault prognostics based on signal complexity and Gaussian process models. *IEEE International Conference on Prognostics and Health Management*, Denver, CO, 2012, 1–8.

Carino, J.A., Zurita, D., Delgado, M., Ortega, J.A., Romero-Troncoso, R.J. Remaining useful life estimation of ball bearings by means of monotonic score calibration. *IEEE International Conference on Industrial Technology*, IEEE, Seville, 2015, 1752–1758.

Coble, J., Hines, J.W. (2014). Incorporating prior belief in the general path model: A comparison of information sources. *Nucl. Eng. Technol.* 46, 773–782.

Dong, S., Luo, T. (2013). Bearing degradation process prediction based on the PCA and optimized LS-SVM model. *Measurement* 46, 3143–3152.

Eker, O.F., Camci, F., Jennions, I. K. (2012). Major Challenges in Prognostics: Study on Benchmarking Prognostics Dataset. *European Conference of Prognostics and Health Management Society* 2012.

El-Koujok, M., Gouriveau, R., Zerhouni, N. (2011). Reducing arbitrary choices in model building for prognostics: an approach by applying parsimony principle on an evolving neuro-fuzzy system. *Microelectron. Reliab.* 51, 310–320.

Gamboa, J. (2017). *Deep Learning for Time Series Analysis*. arXiv:1701.01887v1.

Gillespie, R., Gupta, S. (2017). Real-time Analytic at the Edge: Identifying Abnormal Equipment Behavior and Filtering Data near the Edge for Internet of Things Applications, Paper SAS645.

Guo, L., Gao, H., Huang, H., He, X., Li, S. (2016). Multifeatures fusion and nonlinear dimension reduction for intelligent bearing condition monitoring. *Shock Vib.* 2016, 1–10.

Guo, L., Li, N., Jia, F., Lei, Y., Lin, J. (2017). A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing* 2017, 240, 98-109.

Heimes, F.O. Recurrent neural networks for remaining useful life estimation. *IEEE International Conference on Prognostics and Health Management*, Denver, CO, USA, 2008, 1–6.

Hong, S., Zhou, Z., Zio, E., Hong, K. (2014). Condition assessment for the performance degradation of bearing based on a combinatorial feature extraction method. *Digit. Signal Process.* 2014, 27, 159-166.

Ishibashi, R., Júnior, C.L.N. GFRBS-PHM: A genetic fuzzy rule-based system for PHM with improved interpretability. IEEE International Conference on Prognostics and Health Management, Gaithersburg, MD, USA 2013, 1–7.

Jiafu, W., Shenglong, T., Di L., Shiyong, W., Chengliang, L., Haider, A., Athanasios, V. V. (2017). A Manufacturing Big Data Solution for Active Preventive Maintenance. IEEE Transactions on Industrial Informatics, Vol. 13, No.4, pp. 2039-2047.

Khelif, R., Chebel-Morello, B., Malinowski, S., Laajili, E., Fnaiech, F., Zerhouni, N. (2017). Direct remaining useful life estimation based on support vector regression, IEEE Trans. Industr. Electron. 64, 2276–2285.

Lee J., Lapira, E., Bagheri, B., & Kao, H.A. (2013). Recent Advances and Trends in Predictive Manufacturing Systems in Big Data Environment. Manufacturing Letters, 1(1), 38-41.

Lee, J., Kao, H. A., Yang S. (2014). Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment. The 6<sup>th</sup> CIRP Conference on Industrial Product-Service Systems.

Lei, Y., Li, N., Guo, L., Li, N., Yan, T., Lin, J. (2018). Machinery Health Prognostics: A Systematic Review from Data Acquisition to RUL Prediction. Mechanical Systems and Signal Processing 104, 799-834.

Liao, H., Zhao, W., Guo, H. (2006). Predicting remaining useful life of an individual unit using proportional hazards model and logistic regression model. Reliability and Maintainability Symposium, IEEE, 2006, 127–132.

Liu, K., Gebraeel, N.Z., Shi, J. (2013). A data-level fusion model for developing composite health indices for degradation modeling and prognostic analysis, IEEE Trans. Autom. Sci. Eng. 10, 652–664.

Liu, K., Chehade, A., Song, C. (2017). Optimize the signal quality of the composite health index via data fusion for degradation modeling and prognostic analysis. *IEEE Trans. Autom. Sci. Eng.* 14, 1504–1514.

Loutas, T.H., Roulias, D., Georgoulas, G. (2013). Remaining useful life estimation in rolling bearings utilizing data-driven probabilistic E-support vectors regression. *IEEE Trans. Reliab.* 62, 821–832.

Mahamad, A.K., Saon, S., Hiyama, T. (2010). Predicting remaining useful life of rotating machinery based artificial neural network. *Comput. Math. Appl.* 60, 1078– 1087.

Mosallam, A., Medjaher, K., Zerhouni, N. (2016). Data-driven prognostic method based on Bayesian approaches for direct remaining useful life prediction. *J. Intell. Manuf.* 27, 1037–1048.

Nectoux, P., Gouriveau, R., Medjaher, K., Ramasso, E., Chebel-Morello, B., Zerhouni, N., & Varnier, C. (2012). PRONOSTIA: An experimental platform for bearings accelerated degradation tests. In *IEEE International Conference on Prognostics and Health Management, PHM'12*, 1-8. IEEE Catalog Number: CPF12PHM-CDR.

Peng, Y., Wang, H., Wang, J., Liu, D., Peng, X. A modified echo state network based remaining useful life estimation approach. *IEEE International Conference on Prognostics and Health Management, Denver, CO, USA, 2012*, 1–7.

Qiu, H., Lee, J., Lin, J., Yu, G. (2003). Robust performance degradation assessment methods for enhanced rolling element bearing prognostics, *Adv. Eng. Inform.* 17, 127–140.

Qiu, H., Lee, J., Lin, J., Yu, G. (2006). Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics. *J. Sound Vib.* 289, 1066–1090.

Ramasso, E., Denoeux, T. (2014). Making use of partial knowledge about hidden states in HMMs: an approach based on belief functions, *IEEE Trans. Fuzzy Syst.* 22, 395–405.

Ramasso, E., Gouriveau, R. (2014). Remaining useful life estimation by classification of predictions based on a neuro-fuzzy system and theory of belief functions. *IEEE Trans. Reliab.* 63, 555–566.

Ren, L., Lv, W., Jiang, S. (2015). Machine prognostics based on sparse representation model, *J. Intell. Manuf.*, 1–9.

Rad, J.S., Zhang, Y., Chen, C. (2014). A novel local time-frequency domain feature extraction method for tool condition monitoring using S-transform and genetic algorithm. The 19th World Congress of the International Federation of Automatic Control, Cape Town, South Africa, 1–6.

Singleton, R.K., Strangas, E.G., Aviyente, S. (2015). Extended Kalman filtering for remaining-useful-life estimation of bearings. *IEEE Trans. Industr. Electron.* 62, 1781–1790.

Soualhi, A., Razik, H., Clerc, G., Doan, D. (2014). Prognosis of bearing failures using hidden Markov models and the adaptive neuro-fuzzy inference system. *IEEE Trans. Industr. Electron.* 61, 2864–2874.

Soualhi, A., Medjaher, K., Zerhouni, N. (2015). Bearing health monitoring based on Hilbert-Huang transform, support vector machine, and regression. *IEEE Trans. Instrum. Meas.* 64, 52–62.

Wang, T., Yu, J., Siegel, D., Lee, J. A similarity-based prognostics approach for remaining useful life estimation of engineered systems. *IEEE International Conference on Prognostics and Health Management*, IEEE, 2008, 1–6.

Wang, L., Zhang, L., Wang, X.-Z. (2015). Reliability estimation and remaining useful lifetime prediction for bearing based on proportional hazard model. *J Cent South Univ* 22, 4625–4633.

Wei, M., Chen, M., Zhou, D. (2013). Multi-sensor information based remaining useful life prediction with anticipated performance. *IEEE Trans. Reliab.* 62, 183– 198.

Widodo, A., Yang, B.-S. (2011). Application of relevance vector machine and survival probability to machine degradation assessment. *Expert Syst. Appl.* 38, 2592–2599.

Xi, Z., Jing, R., Wang, P., Hu, C. (2014). A copula-based sampling method for data-driven prognostics, *Reliab. Eng. Syst. Safety* 132, 72–82.

Xu, J., Wang, Y., Xu, Li. (2014). PHM-oriented integrated fusion prognostics for aircraft engines based on sensor data. *IEEE Sens. J.* 14, 1124–1132.

Yoo, Y., Baek, J. (2018). A Novel Image Feature for the Remaining Useful Lifetime Prediction of Bearings Based on Continuous Wavelet Transform and Convolutional Neural Network. *Appl. Sci.* 2018, 8 1102; doi:10.3390/app8071102.

Yu, J. (2011). Bearing performance degradation assessment using locality preserving projections and Gaussian mixture models. *Mech. Syst. Signal Process.* 25, 2573–2588.

Yu, J. (2012). Machine tool condition monitoring based on an adaptive Gaussian mixture mode. *J. Manuf. Sci. Eng.* 134, 1–13.

Zhang, D.D. (2011). An adaptive procedure for tool life prediction in face milling. *Proc. Inst. Mech. Eng., Part J: J. Eng. Tribol.* 225, 1130–1136.

Zurita, D., Carino, J.A., Delgado, M., Ortega, J.A. Distributed neuro-fuzzy feature forecasting approach for condition monitoring. *IEEE Conference on Emerging Technology and Factory Automation, IEEE, Barcelona, 2014, 1–8.*

## APPENDICES

### Appendix A

#### #Discrete Wavelet Transform and Feature Extraction Functions

```
import pandas as pd
import numpy as np
import os
import pywt

from scipy.stats import kurtosis

def lowpassfilter(signal, thresh = 0.63, wavelet="db4"):
    thresh = thresh*np.nanmax(signal)
    coeff = pywt.wavedec(signal, wavelet, mode="per" )
    coeff[1:] = (pywt.threshold(i, value=thresh, mode="soft" ) for i in coeff[1:])
    reconstructed_signal = pywt.waverec(coeff, wavelet, mode="per" )
    return reconstructed_signal

def crest_factor(row):
    x = np.max(np.abs(row))/np.sqrt(np.mean(np.square(row)))
    return x

def rms(row):
    x = np.sqrt(np.mean(np.square(row)))
    return x

def calculaterul(row, seconds, operationtime):
    rul = operationtime - seconds
    return rul

def createDataFrameWithFeatures(path, operationtime, windowsize):
    filenames = os.listdir(path)
    filenames.sort()

    names = ['hours', 'minutes', 'seconds', 'microseconds', 'acc1', 'acc2']
    results = pd.DataFrame([], columns = names)

    seconds = 0;
```

```

for f in filenames:
    if f.startswith("acc") and f.endswith(".csv"):
        namedf = pd.read_csv(os.path.join(path, f), names=names, engine='python')
        namedf['time'] = namedf.apply(lambda row: timeconvert(row, seconds), axis=1)
        namedf['actualrul'] = namedf.apply(lambda row: calculaterul(row, seconds, operati
ontime), axis=1)

        seconds = seconds + 10;

        namedf = namedf.drop("hours", axis=1)
        namedf = namedf.drop("minutes", axis=1)
        namedf = namedf.drop("seconds", axis=1)
        namedf = namedf.drop("microseconds", axis=1)

        results = results.append(namedf, sort = False)

results = results.drop("hours", axis=1)
results = results.drop("minutes", axis=1)
results = results.drop("seconds", axis=1)
results = results.drop("microseconds", axis=1)

print(results.columns)
print(results.shape)

signalacc1 = results['acc1'].values
signalacc2 = results['acc2'].values
print(signalacc1)

rec1 = lowpassfilter(signalacc1, 0.4)
rec2 = lowpassfilter(signalacc2, 0.4)

results['acc1filtered'] = rec1
results['acc2filtered'] = rec2

print(results.head(10))

start = time.time()
rolling_mean = results['acc1filtered'][0:windowsize].mean()

results['smoothacc1mean'] = 0
results['smoothacc2mean'] = 0
results['smoothacc1std'] = 0
results['smoothacc2std'] = 0
results['smoothacc1var'] = 0
results['smoothacc2var'] = 0
results['smoothacc1krtss'] = 0

```



```

results['smoothacc2krtss'] = 0
results['smoothacc1rms'] = 0
results['smoothacc2rms'] = 0
results['smoothacc1skew'] = 0
results['smoothacc2skew'] = 0
results['smoothacc1cf'] = 0
results['smoothacc2cf'] = 0

rowcount = results.shape[0]
for i in range(0, rowcount, windowsize):
    results['smoothacc1mean'][i:i + windowsize] = results['acc1filtered'][i:i + windowsize].mean()
    results['smoothacc2mean'][i:i + windowsize] = results['acc2filtered'][i:i + windowsize].mean()
    results['smoothacc1std'][i:i + windowsize] = results['acc1filtered'][i:i + windowsize].std()
    results['smoothacc2std'][i:i + windowsize] = results['acc2filtered'][i:i + windowsize].std()
    results['smoothacc1var'][i:i + windowsize] = results['acc1filtered'][i:i + windowsize].var()
    results['smoothacc2var'][i:i + windowsize] = results['acc2filtered'][i:i + windowsize].var()
    results['smoothacc1krtss'][i:i + windowsize] = results['acc1filtered'][i:i + windowsize].kurtosis()
    results['smoothacc2krtss'][i:i + windowsize] = results['acc2filtered'][i:i + windowsize].kurtosis()
    results['smoothacc1skew'][i:i + windowsize] = results['acc1filtered'][i:i + windowsize].skew()
    results['smoothacc2skew'][i:i + windowsize] = results['acc2filtered'][i:i + windowsize].skew()
    results['smoothacc1rms'][i:i + windowsize] = rms(results['acc1filtered'][i:i + windowsize])
    results['smoothacc2rms'][i:i + windowsize] = rms(results['acc2filtered'][i:i + windowsize])
    results['smoothacc1cf'][i:i + windowsize] = crest_factor(results['acc1filtered'][i:i + windowsize])
    results['smoothacc2cf'][i:i + windowsize] = crest_factor(results['acc2filtered'][i:i + windowsize])
def reduceDataFrame(results, bearingId, windowsize):
    reduced = pd.DataFrame([], columns = results.columns)
    results['time'] = results['time'] // 60
    results['actualrul'] = results['actualrul'] // 60

rowcount = results.shape[0]
y = 0
for i in range(0, rowcount, windowsize):
    reduced.loc[y] = results.iloc[i]

```

```
y = y + 1  
reduced['id'] = bearingId  
  
print(reduced.head())  
print(reduced.tail())  
return reduced
```



## Appendix B

#HI Calculation for train and test bearings

```
import itertools
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.formula.api import ols

#Read train and test datasets
traindf = pd.read_csv('/content/gdrive/My Drive/Femto/mergedreducedtrainbearings.csv',
engine = 'python', index_col=False)
testdf = pd.read_csv('/content/gdrive/My Drive/Femto/mergedreducedtestbearings.csv',
engine = 'python', index_col=False)

temp_df = traindf.copy()

# assigning na to every observation in HI column
temp_df['HI'] = np.nan

train_bearings = ['b11', 'b12', 'b21', 'b22', 'b31', 'b32']

for i in range(0,6):
    print(train_bearings[i])
    # assigning one to first 15 observations of every unit
    temp_df['HI'].iloc[a:a+15] = 1

    a += max(traindf[traindf['id'] == train_bearings [i]]['time'])
    a = int(a)

    # assigning zero to last 15 observations of every unit
    temp_df['HI'].iloc[a-15:a+1] = 0
    a+=1

df_reg = temp_df.copy()

# dropping the unrelated columns
df_corr = df_reg.copy().drop(['acc1', 'acc2', 'acc1filtered', 'acc2filtered', 'id', 'time', 'actual
rul'], axis=1)

# correlations are calculated for features to be selected
df_corr_ = df_corr.corr()
df_corr_[['HI']]
```

```
m = ols('HI ~ smoothacc2mean+smoothacc1std+smoothacc2std+smoothacc2krtss+smoothacc1rms+smoothacc2rms+smoothacc1cf+smoothacc2cf', df_corr).fit()
```

```
df_test_slice = test_df.drop(['acc1', 'acc2', 'time', 'acc1filtered', 'acc2filtered', 'actualrul', 'smoothacc1mean', 'smoothacc1var', 'smoothacc2var', 'smoothacc1krtss', 'smoothacc1skew', 'smoothacc2skew', 'id'], axis = 1)
```

```
test_df['HI']=m.predict(df_test_slice)
```

# A function to calculate moving average of the dataframe

```
def moving_average(df, unit):
    temp_df = df[df['id'] == unit]
    rolling = temp_df['HI'].rolling(window = 5)
    rolling_mean = rolling.mean()
    return rolling_mean
```

```
test_bearings = ['b13', 'b14', 'b15', 'b16', 'b17', 'b23', 'b24', 'b25', 'b26', 'b27', 'b33']
```

```
all_rolling_means = [moving_average(test_df, test_bearings[i]) for i in range(0,11)]
test_df = test_df.assign(MA_HI = list(itertools.chain.from_iterable(all_rolling_means)))
test_df.dropna(inplace = True)
test_df.head()
```

```
df_train_slice = traindf.drop(['acc1', 'acc2', 'time', 'acc1filtered', 'acc2filtered', 'actualrul', 'smoothacc1mean', 'smoothacc1var', 'smoothacc2var', 'smoothacc1krtss', 'smoothacc1skew', 'smoothacc2skew', 'id'], axis = 1)
```

```
traindf['HI'] = m.predict(df_train_slice)
all_rolling_means = [moving_average(traindf, train_bearings[i]) for i in range(0,6)]
traindf = traindf.assign(MA_HI = list(itertools.chain.from_iterable(all_rolling_means)))
traindf.dropna(inplace=True)
traindf2.head()
```

## Appendix C

### #LSTM Network

```

import keras
import keras.backend as K
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
import matplotlib.cm as cm
import seaborn as sns
import os

from sklearn import preprocessing
from sklearn.metrics import confusion_matrix, recall_score, precision_score
from keras.models import Sequential, load_model
from keras.layers import Dense, Dropout, LSTM, Activation

sequence_length = 25
def gen_sequence(id_df, seq_length, seq_cols):
    data_matrix = id_df[seq_cols].values
    num_elements = data_matrix.shape[0]
    for start, stop in zip(range(0, num_elements -
seq_length), range(seq_length, num_elements)):
        yield data_matrix[start:stop, :]

sequence_cols = ['time', 'smoothacc2mean', 'smoothacc1std', 'smoothacc2std', 'smootha
cc2krtss', 'smoothacc1rms', 'smoothacc2rms', 'smoothacc1cf', 'smoothacc2cf', 'MA_HI']

seq_gen = (list(gen_sequence(traindf[traindf['id']==id], sequence_length, sequence_cols
)) for id in traindf['id'].unique())

seq_array = np.concatenate(list(seq_gen)).astype(np.float32)

def gen_labels(id_df, seq_length, label):
    data_matrix = id_df[label].values
    num_elements = data_matrix.shape[0]
    return data_matrix[seq_length:num_elements, :]

label_gen = [gen_labels(traindf[traindf['id']==id], sequence_length, ['actualrul'])
              for id in traindf['id'].unique()]

label_array = np.concatenate(label_gen).astype(np.float32)

```

```

def r2_keras(y_true, y_pred):
    SS_res = K.sum(K.square( y_true - y_pred ))
    SS_tot = K.sum(K.square( y_true - K.mean(y_true) ) )
    return ( 1 - SS_res/(SS_tot + K.epsilon()) )

nb_features = seq_array.shape[2]
nb_out = label_array.shape[1]

model = Sequential()
model.add(LSTM(
    input_shape=(sequence_length, nb_features),
    units=100,
    return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(
    units=50,
    return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(units=nb_out))
model.add(Activation("relu"))
model.compile(loss='mean_squared_error', optimizer='rmsprop', metrics=['mae', r2_keras])

model_path = '/content/gdrive/My Drive/Femto/regression_model.h5'

history = model.fit(seq_array, label_array, epochs=200, batch_size=50, validation_split
=0.3, verbose=1,
    callbacks = [
        keras.callbacks.ModelCheckpoint(model_path, monitor='val_loss', save_b
est_only=True, mode='min', verbose=0)]
    )

```

## **BIOGRAPHICAL SKETCH**

Ceren Nur BAŞ was born on March 31, 1992 in Şanlıurfa. After graduating from Burak Bora Anatolian High School in 2010, she made a gap year in Germany with the intention of sharing culture and learning a new language. After that year, she began to study in Computer Engineering department in Galatasaray University. She made her final project on daily routine follow-up using motion sensors and graduated from Computer Engineering department in 2015. After that she enrolled in M.Sc. program in Computer Engineering department in Galatasaray University. In the last 4 years she is working as a software developer at Siemens AG.