Bachelor's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of ICT and Natural Sciences

Jørgen Finsveen
Harald Wangsvik Fredriksen
Even Johan Pereira Haslerud

Health Management - Anomaly Detection and Classification

Development of machine learning algorithms for detection and classification of anomalies in machinery equipment

Bachelor's thesis in Computer Science
Supervisor: Saleh Abdel-Afou Alaliyat
Co-supervisor: Muhammad Umair Hassan
May 2024

NTNU
Norwegian University of
Science and Technology

Jørgen Finsveen
Harald Wangsvik Fredriksen
Even Johan Pereira Haslerud

Health Management - Anomaly Detection and Classification

Development of machine learning algorithms for detection and classification of anomalies in machinery equipment

Bachelor's thesis in Computer Science
Supervisor: Saleh Abdel-Afou Alaliyat
Co-supervisor: Muhammad Umair Hassan
May 2024

Norwegian University of Science and Technology
Faculty of Information technology and Electrical Engineering (IE)
Department of ICT and Natural Sciences

NTNU

ABSTRACT

Anomaly detection and classification is an important aspect of machine learning, particularly for enhancing condition monitoring services in marine vessels. Kongsberg Maritime is exploring the implementation of these technologies. This thesis addresses this challenge by proposing a framework for anomaly detection and classification in marine vessels.

In this project, the team aimed to detect anomalies in ship engines using a Long Short-Term Memory (LSTM) model. The engine was divided into five subsystems, with an LSTM model trained for each subsystem to predict its state. Anomalies were detected by comparing the predicted states with the actual states. This method proved effective, with all models achieving strong results in terms of Mean Squared Error and r^2.

While the project successfully established an anomaly detection solution, anomaly classification could not be implemented due to insufficient data. Nonetheless, Kongsberg Maritime is satisfied with the anomaly detection approach and is interested in further exploring the potential for anomaly classification using the anomaly detection framework as a basis.

The team employed traditional agile methodology principles throughout the project, adapting them to the context of machine learning. Despite some challenges, this approach was deemed successful.

ABSTRAKT

Deteksjon og klassifisering av anomalier er et viktig aspekt innen maskinlæring, spesielt for å forbedre tilstandsovervåkingstjenester i marine fartøyer. Kongsberg Maritime ønsker å utforske implementeringen av disse teknologiene. Denne oppgaven tar for seg denne utfordringen ved å foreslå et rammeverk for deteksjon og klassifisering av anomalier i skipsmotorer.

I dette prosjektet forsøker gruppen å oppdage anomalier i skipsmotorer ved å bruke en Long Short-Term Memory modell. Dette ble gjort ved å dele motoren inn i fem forskjellige delsystemer, og trene en Long Short-Term Memory modell for hvert delsystem som predikerer systemets tilstand. Prediksjonene sammenlignes med den faktiske tilstanden til delsystemene for å oppdage uregelmessigheter. Denne tilnærmingen viste seg å være tilstrekkelig for anomalideteksjon, og alle modellene oppnådde gode resultater i form av gjennomsnittlig kvadratfeil (MSE) og r^2.

Selv om prosjektet har etablert en deteksjonsløsning for anomalier, kunne ikke anomaliklassifisering implementeres på grunn av utilstrekkelig data. Kongsberg Maritime er likevel fornøyd med anomalideteksjonstilnærmingen og er interessert i å utforske potensialet for anomaliklassifisering ytterligere ved å bruke rammeverket for anomalideteksjon som grunnlag.

Teamet brukte tradisjonelle smidige metodikkprinsipper gjennom hele prosjektet, og tilpasset dem til konteksten for maskinlæring. Til tross for noen utfordringer ble denne tilnærmingen ansett som vellykket.

PREFACE

This project was proposed by David Vågnes on behalf of Kongsberg Maritime in Ålesund. The group would like to express their gratitude for this opportunity.

Furthermore, the group would like to thank their supervisor Saleh Abdel-Afou Alaliyat and co-supervisor Muhammad Umair Hassan at the Norwegian University of Science and Technology for their guidance throughout the project.

Ålesund - 20.05.2024

Jørgen Finsveen
Harald Wangsvik Fredriksen
Even Johan Pereira Haslerud

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ABBREVIATIONS

- **AI** Artificial Intelligence
- **API** Application Programming Interface
- **CM** Condition Monitoring
- **HM** Health Management
- **HMADC** Health Management, Anomaly Detection and Classification
- **HTTP** Hypertext Transfer Protocol
- **IQR** Interquartile Range
- **KM** Kongsberg Maritime
- **LSTM** Long Short-Term Memory
- **ML** Machine Learning
- **MAE** Mean Absolute Error
- **MSE** Mean Squared Error
- **NN** Neural Network
- **NTNU** Norwegian University of Science and Technology
- **PCA** Principal Component Analysis
- **PSV** Platform Supply Vessel
- **RNN** Recurrent Neural Network
- **RMSE** Root Mean Squared Error
- **RPM** Rotations per Minute
- **SSL** Secure Sockets Layer
- **TLS** Transport Layer Security
- **TPU** Tensor Processing Unit

CHAPTER

ONE

INTRODUCTION

1.1 Motivation

This project was presented to the group by Kongsberg Maritime (KM), who needed a system for equipment condition monitoring service. The group decided to accept this opportunity for several reasons. Firstly, they wanted to challenge themselves with a complex problem which would require them to do extensive research and learn new technologies in order to produce a viable solution. The group considers this to be one of the most significant qualities of engineers. Secondly, they sought a project in which they could gain experience implementing machine learning technologies in an industrial perspective. The last reason was that the group wanted to conduct their bachelor thesis in cooperation with a technologically advanced and respected company such as KM, which also seemed to be appropriate considering its influence on the local business community.

1.2 Project description

1.2.1 Background

The Health Management venture is intended as a series of preventative measures focusing on providing customers with operational stability. The objective is to monitor critical vessel machinery through collecting data logged from various sensors measuring different aspects of a given unit, and to analyse said data for detection and classification of anomalies as well as maintenance prediction. By utilizing condition monitoring, it is expected to help:

- Avoiding unplanned maintenance and keep critical assets operating.
- Predict maintenance needs which can be planned accordingly.
- Reducing the total cost of asset ownership.
- Maximizing the operational profitability.
- Ensuring sustainable maintenance and optimized machinery performance.

KM is already collecting and monitoring data from vessels at sea, however, they seek to find ways to automate the monitoring and classification of the data. In order to achieve this, KM would like to approach the problem as a subject to machine learning implementation. A satisfactory solution would be to implement machine learning techniques which makes an autonomous system capable of investigating machinery data in order to:

1. Detect anomalous long-term trends.
2. Classify detected anomalies.

1.2.2 Existing Solution

Kongsberg Maritime's present Health Management/Condition Monitoring service utilizes Failure Mode analysis of machinery data, which implies that one compares data with known failure patterns in operational data. Each anomaly detected is then classified by product experts. This system is used to monitor the overall health of marine vessels in order to identify equipment degradation, advising their clients, and schedule service on the equipment.

1.2.3 Problem Statement

The present condition monitoring service is limited to only being able to detect known failure patterns, leaving out potential anomalies which follows unknown failure patterns. The system is also resource consuming in terms of manual intervention such as product experts performing repetitive work tasks. Furthermore, it is expensive to maintain the current solution, and it has a low degree of scalability.

KM has been spent significant effort on attempting to implement machine learning for anomaly detection, but their attempts has not been successful due to their models only being capable of capturing instantaneous anomalies rather than anomalous trends over time. Their solution was considered to be incompatible with the needs of equipment condition monitoring according to the project proposal [1].

As there has been a vast number of technological advancements since the previous attempt to automate the Health Management service, KM seeks to approach the problem again with new insights and perspectives. The main objective is to enhance the service using ML, making it capable of detecting and classifying anomalous long-trend behavior by learning both known and unknown failure patterns. This formulation forms the basis for defining the following problem statement for this project:

How can machine learning be implemented in order to detect and classify anomalous behavior in machinery of various types?

1.2.4 Constraints

Due to KM having a large portfolio of clients which owns different types of marine vessels, the scope of the problem statement is relatively large. There are also several components in the vessels which are subject to condition monitoring, and it would be unrealistic that the group will be able to produce a solution which covers such a large domain in the given time-span. Therefore, are the following assumptions and delimitations made:

- The project will be focused on developing ML models which is able to detect and classify long-term anomalies in engines. Low- and high-speed rotating machines such as winches and thrusters will not be considered.
- Different pieces of machinery may require monitoring of different parameters describing important aspects of the condition of the engine. Therefore, it will be assumed that the group should make different models targeting different aspects of the machine rather than making a single, general model which covers the entire engine.
- Classification of anomalies must be based on an initial classification by a product expert.
- ML models must have a sensitivity-level for anomaly detection which is appropriate for the significance of the given anomaly.

1.2.5 Solution Requirements

In order for a solution to be recognized as satisfactory for the problem, it should be capable of achieving the following:

1. Detect long-term anomalous trends in various machinery equipment with a reasonable accuracy and sensitivity.
2. Accurately classify the detected anomalies.
3. Prove to be a more efficient solution than the present condition monitoring service in terms of accuracy, scalability, and maintenance.

1.2.6 Stakeholders

Kongsberg Maritime will use the research and results of this project to find ways to enhance their current condition monitoring service. Should the project result in an accepted solution, they will implement their own machine learning solutions based on the project's solution. If the project does not culminate in an accepted solution, KM will use the conclusion and the research conducted in this project in order to approach the problem in other ways.

1.2.7 Consequences

A viable solution for the problem could result in a set of models capable of seamless integration into KM's existing systems, making their condition monitoring service a lot more efficient and scalable. Such an achievement would enable KM to enhance their offerings and deliver superior value to their clientele.

Furthermore, KM has defined a model known as the "Data Value Chain" which describes the process of collecting, processing and analyzing data in order to gather insight which can produce value to their clients.

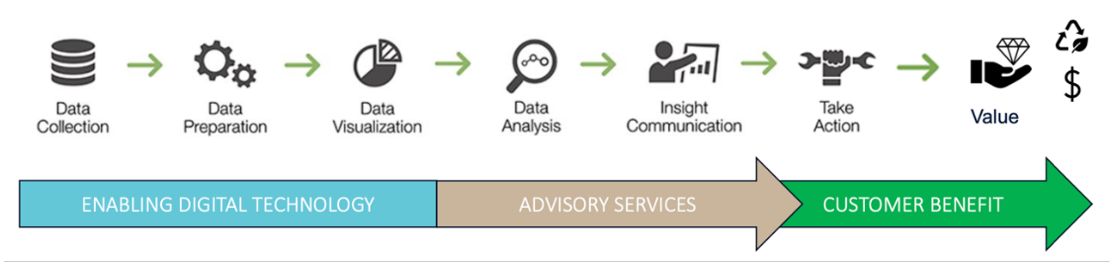


Figure 1.2.1: Data value chain
[2]

As seen in Figure 1.2.1, KM defines the three sub-chains marked with different colouring in the data value chain as subject to three different services. From left to right, these services are:

- **Enabling digital technology**: Health Management
- **Advisory services**: Marine & DP Operations
- **Customer benefit**: Eco Insight

Of these sub-chains, the first three links regarding data collection, preparation, and visualization are of most relevance for the HM service and therefore for this project. However, data analysis is a crucial part of the ML process and will therefore also be of relevance to the project.

It is believed that a satisfactory solution to the project's problem would provide the necessary resources to enhance the insight communication link and the following links in the data value chain as well as streamlining the HM and CM services which KM provides for their customers.

1.3 Related Works

ML and anomaly detection are fields which are subject to research in both the industry as well as in academia. As a way for the reader to gain more knowledge of the fields and the recent research, the following related works are presented:

A Deep Learning Method for the Prediction of Ship Fuel Consumption in Real Operational Conditions [3].

> This research paper addresses deep learning techniques for prediction of ship fuel consumption, based on 266 variables retrieved from a Kamsarmax bulk carrier. The data is subject to big data analysis utilizing a Decision Tree model for variable importance estimation. The deep learning predictions are made by implementing a Bi-directional Long Short-Term Memory model.

The article and the associated project shares similarities with this thesis. Both projects addresses machine learning for implementation in marine vessels, and both projects are encompassing the influence of real operational conditions, retrieved from sensor data. However, the projects aims towards different goals. While the mentioned article addresses fuel consumption, this thesis will address anomaly detection and classification of several components in ship engines.

Fault Detection with LSTM-based Variational Autoencoder for Maritime Components [4].

The research paper is delivered by NTNU and addresses anomaly detection in maritime components. The project approaches this using a Long Short-Term Memory based Variational Autoencoder. This model is trained on real data from the Gunnerus research vessel.

NTNU's own research is closely related to this thesis, where both projects attempts to use ML for anomaly detection in vessels. Features separating the projects from each other are some details such as which type of ship the anomaly detection is targeted towards, as well as this thesis also means to focus on both anomaly detection and classification, while NTNU's project exclusively addresses anomaly detection.

Machine Learning for Anomaly Detection: A Systematic Review [5].

IEEE published this report, in which the objective was to conduct a systematic literature review of ML models used for anomaly detection. The paper reviews 290 research papers on this topic, and models are analysed in four perspectives being the applications of anomaly detection, the ML techniques used, model performances, and anomaly classification. The report presents a summary of different implementations, emphasising their performance and popularity.

This thesis shows some relevance to IEEE's report. Both addresses anomaly detection and classification, and approaches to this may be mentioned in both papers. But while IEEE's report focuses on reviewing different ways to implement anomaly detection and classification, it does not present an implementation of its own.

Research on anomaly detection and real-time reliability evaluation with the log of cloud platform [6].

This research paper addresses the usage of an ensemble learning model for anomaly detection of system logs generated by large-scale information systems. The project culminates in a model which accurately detects anomalies in system logs through an approach which differs from traditional machine learning methods.

This paper shows resemblance to this thesis, both addressing anomaly detection through machine learning methods. Both projects are also motivated to find alternative approaches to this rather than using more traditional machine learning technologies. What sets the projects apart are amongst other the target system, being system logs in the research paper, and sensor measurements of a vessel engine for this thesis.

1.4 Thesis Structure

This thesis is divided into seven chapters:

Chapter 1: Introduction

Describes the purpose of this thesis and the problem it is related to, as well as introducing the solution envisioned by the stakeholders.

Chapter 2: Theory

Provides the theoretical background of the development of the solution to the problem thesis.

Chapter 3: Methods

Contains a overview of the development process and justification of the decisions made along the way.

Chapter 4: Results

Presents the results of the project.

Chapter 5: Discussion

Contains reflection around the results and the development strategies.

Chapter 6: Conclusions

Summarises the findings of the project, draws conclusions, and discusses future works.

Chapter 7: Social Impact

Presents the impact of the thesis in social, ethical, sustainability, and financial aspects.

CHAPTER
TWO

THEORY

The following chapter introduces the theoretical background, presenting concepts, technologies, and methodology principles used throughout the project. Theories introduced in this chapter will be referenced to throughout the report as the different concepts displays their relevance.

2.1 The Scientific Method

The scientific method is a fundamental tool for gathering new knowledge based on empiricism. "Among the activities often identified as characteristic of science are systematic observation and experimentation, inductive and deductive reasoning, and the formation and testing of hypotheses and theories." [7]. [8] claims that the hypothetical-deductive method is acknowledged as the most fundamental method in modern science. It describes the process of making an observation, formulating a hypothesis, identify expected results, test the hypothesis, and lastly, attempt to falsify or confirm the hypothesis. Below, Figure 2.1.1 represents a simplified flowchart of the steps in the hypothetical-deductive method:

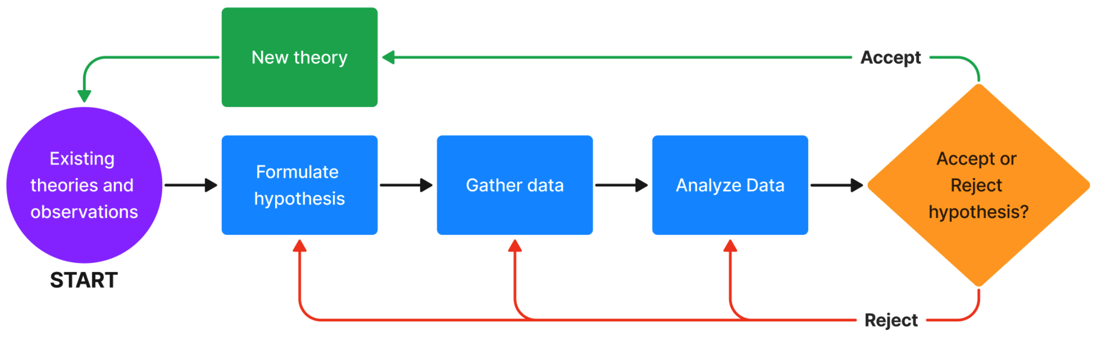


Figure 2.1.1: Hypothetical-deductive method flowchart

The problem domain of this project makes it necessary to conduct a fair amount of research and experimentation. When studying data, its relations, and how ML can be implemented to learn its nature, the team consider it to be paramount that the project is conducted based on well-known procedures for scientific work.

2.2 Project Tools

2.2.1 Development Tools

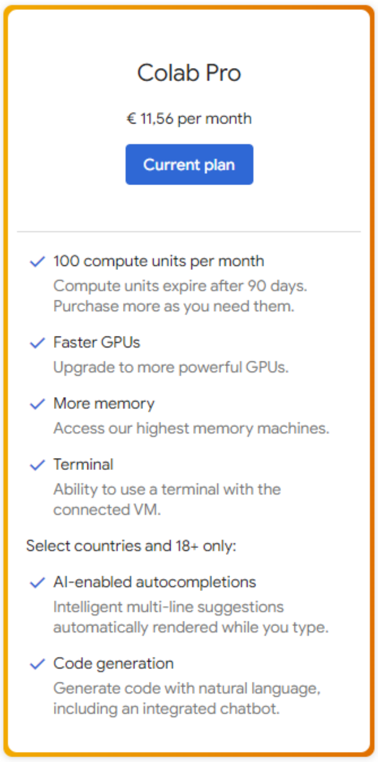
2.2.1.1 Google Colab

Introduction

"Colab is a hosted Jupyter Notebook service that requires no setup to use and provides free access to computing resources, including GPUs and TPUs. Colab is especially well suited to machine learning, data science, and education." [9] Colab is a program created by Google, it stands out as an exceptional tool for machine learning projects due to its compatibility with popular libraries and frameworks used in machine learning, such as Pandas, scikit-learn and matplotlib.

Advantages

Among the primary advantages of Google Colab are its capabilities for real-time collaborative writing, cloud-based storage integration, and access to powerful GPU resources. The platform's real-time collaborative writing feature allows for simultaneous code development and analysis by multiple users at the same time. As a Google product, Colab is compatible with Google drive, enabling the storage and retrieval of large datasets directly within the cloud environment. This integration proved crucial for the project, given the substantial size of the dataset. While the free version of Colab offers access to standard GPUs, Colab Pro extends access to more powerful NVIDIA V100 or A100 Tensor Core GPUs, providing the necessary computational power for intensive data processing and model training.


Colab Pro

€ 11,56 per month

Current plan

✓ 100 compute units per month
Compute units expire after 90 days. Purchase more as you need them.

✓ Faster GPUs
Upgrade to more powerful GPUs.

✓ More memory
Access our highest memory machines.

✓ Terminal
Ability to use a terminal with the connected VM.

Select countries and 18+ only:

✓ AI-enabled autocompletions
Intelligent multi-line suggestions automatically rendered while you type.

✓ Code generation
Generate code with natural language, including an integrated chatbot.

Figure 2.2.1: Colab Pro features [10]

Sharing and Collaboration features

The process of saving and sharing notebooks in Google Colab are highly user-friendly. "Colab notebooks are stored in Google Drive, or can be loaded from GitHub. Colab notebooks can be shared just as you would with Google Docs or Sheets." [11]. When a notebook is instantiated within Google Colab, the creator of the notebook can share it with others, assigning them with a role of either editor, viewer or commenter. Each role has their own permissions.

Challenges and Limitations

Google Colab's merge conflict management is very poor. Occasionally, simultaneous edits by multiple users can result in failure to save the notebook, triggering an error message with the option to show difference. However, this comparison only delineates the changes made by each contributor without offering a method to resolve it. Only allowing the user to save their changes, which results in the deletion of the other person's changes.

In response to these limitations the team has opted a strategy of creating a notebook for each model. This approach both prevents overwriting the contributions by other team members and engenders a more organized and efficient folder/model structure, simplifying the process of locating and accessing the desired model. Nevertheless, this approach does not fix collaboration within the same notebook; however, with these modifications, such collaboration was no longer necessary.

2.2.2 Documentation Tools

2.2.2.1 Atlassian

Atlassian is a software corporation that specializes in developing applications for software developers and project managers [12]. Among these applications are Jira, Confluence, BitBucket, and Trello. for this project, the team has employed Jira and Confluence for purposes such as issue tracking, the storage and management of documentation and comprehensive sprint management.

2.2.2.2 Jira

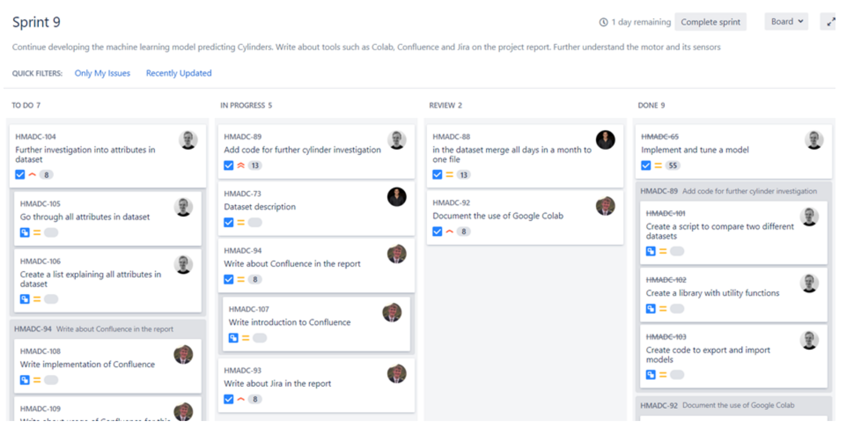
Introduction

Jira is the first ever tool created by Atlassian. Its main purpose is providing task management boards for teams to manage their work in a structured and easy way. Some of its other features are time sheets, sprint tracking and reporting.

Scrum board

The boards functionalities made it possible to initiate sprints, create and manage issues and log time. The HMADC board was structured into three columns: 'To Do', 'In Progress', and 'Done'. To enhance the quality of issue tracking, an additional column titled "Review" was added, positioned prior to the 'Done' column. This arrangement provides a more thorough evaluation of tasks, acknowledging the possibility that items perceived as completed might require further refinement following a peer review. It is a good practise to not move issues after they have been classified as completed.

In the picture above, the board with distribution of issues across various stages of completion is shown. Some issues have sub-tasks, which represent more detailed activities for the respective issue. Furthermore, tasks are categorized by type and priority. The blue icon with a check mark signifies a standard task, whereas a



Sprint 9

Continue developing the machine learning model predicting Cylinders. Write about tools such as Colab, Confluence and Jira on the project report. Further understand the motor and its sensors

TO DO 7

HMADC-104 Further investigation into attributes in dataset 8

HMADC-105 Go through all attributes in dataset

HMADC-106 Create a list explaining all attributes in dataset

HMADC-94 Write about Confluence in the report

HMADC-108 Write implementation of Confluence

HMADC-109

IN PROGRESS 5

HMADC-89 Add code for further cylinder investigation 13

HMADC-73 Dataset description

HMADC-94 Write about Confluence in the report 8

HMADC-107 Write introduction to Confluence

HMADC-93 Write about Jira in the report 8

REVIEW 2

HMADC-88 in the dataset merge all days in a month to one file 13

HMADC-92 Document the use of Google Colab 8

DONE 9

~~HMADC-65~~ Implement and tune a model 55

HMADC-89 Add code for further cylinder investigation

~~HMADC-101~~ Create a script to compare two different datasets

~~HMADC-102~~ Create a library with utility functions

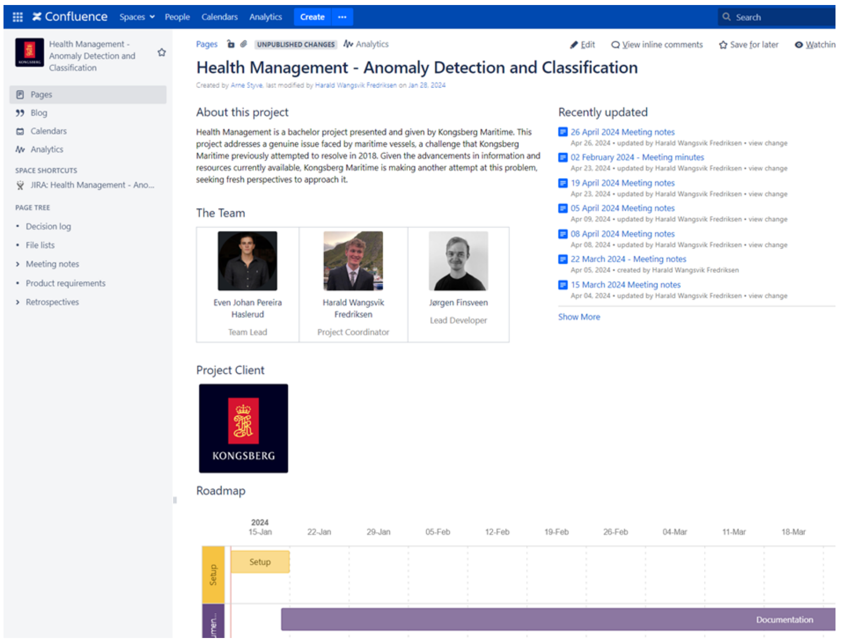
~~HMADC-103~~ Create code to export and import models

HMADC-92 Document the use of Google Colab

Figure 2.2.2: Jira scrum board
[13]

blue icon featuring two boxes represents a sub-task. Adjacent to task identifiers, the coloured lines next to the type icon symbolises the priority of a task, red lines indicate high priority, yellow lines indicate medium priority and blue are low priority.

2.2.2.3 Confluence



Health Management - Anomaly Detection and Classification

Health Management - Anomaly Detection and Classification

Created by Arne Styve, last modified by Harald Wangsvik Fredriksen on Jan 28, 2024

About this project

Health Management is a bachelor project presented and given by Kongsberg Maritime. This project addresses a genuine issue faced by maritime vessels, a challenge that Kongsberg Maritime previously attempted to resolve in 2018. Given the advancements in information and resources currently available, Kongsberg Maritime is making another attempt at this problem, seeking fresh perspectives to approach it.

The Team

Even Johan Pereira Haslerud	Harald Wangsvik Fredriksen	Jørgen Finsveen
Team Lead	Project Coordinator	Lead Developer

Project Client

Recently updated

26 April 2024 Meeting notes
Apr 26, 2024 • updated by Harald Wangsvik Fredriksen • view change

02 February 2024 - Meeting minutes
Apr 23, 2024 • updated by Harald Wangsvik Fredriksen • view change

19 April 2024 Meeting notes
Apr 23, 2024 • updated by Harald Wangsvik Fredriksen • view change

05 April 2024 Meeting notes
Apr 09, 2024 • updated by Harald Wangsvik Fredriksen • view change

08 April 2024 Meeting notes
Apr 08, 2024 • updated by Harald Wangsvik Fredriksen • view change

22 March 2024 - Meeting notes
Apr 05, 2024 • created by Harald Wangsvik Fredriksen

15 March 2024 Meeting notes
Apr 04, 2024 • updated by Harald Wangsvik Fredriksen • view change

Figure 2.2.3: Confluence main page
[14]

"Confluence is a team workspace where knowledge and collaboration meet. Dynamic pages give your team a place to create, capture, and collaborate on any project or idea. Spaces help your team structure, organize, and share work, so every team member has visibility into institutional knowledge and access to the information they need to do their best work." [15]. The platform is notably flexible, offering extensive customization options to tailor the user experience to specific needs. It also has high compatibility with Jira, enabling seamless integration and cohesive documentation across these platforms.

2.3 Work Methodology

2.3.1 SCRUM

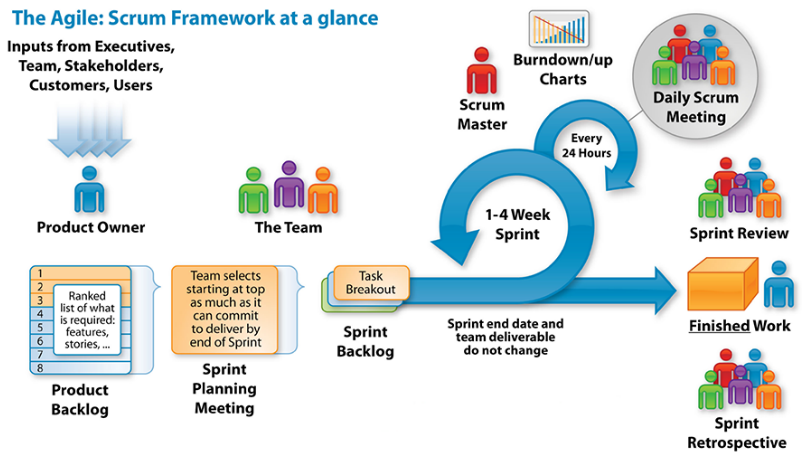

The Agile: Scrum Framework at a glance
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint end date and team deliverable do not change
Sprint Review
Finished Work
Sprint Retrospective

Figure 2.3.1: The Agile: Scrum Framework at a glance [16]

2.3.1.1 Introduction

“Scrum is an agile project management framework that helps teams structure and manage their work through a set of values, principles, and practices.” [17]. The methodology is often used during software development to help developers continuously improve their work and actively reflect on the project execution. This is achieved through systematic implementation of regular sprints, structured meetings, and strategic planning.

2.3.1.2 Sprints

When working in a scrum framework you work with sprints, which are “time-boxed periods when a scrum team works to complete a set amount of work” [18]. The sprint timeline encompasses four main phases: Sprint Planning, Sprint Backlog, Sprint Retrospective, and Sprint Review. Throughout the duration of the sprint, developers works with designated tasks set for completion within that specific sprint cycle. Subsequently, the team review their methodologies, with a focus on identifying areas for improvement and enhancing future performance.

2.3.1.3 Planning

"Sprint planning is an event in scrum that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved. Sprint planning is done in collaboration with the whole scrum team." [19]. Based on this objective there are distributed tasks/issues that are meant to be completed within the time period of the sprint. The list of these tasks is called the sprint backlog.

2.3.1.4 Product backlog

In the planning stage of the project, the team created an extensive list of tasks anticipated to be necessary in later stages, which were then added into the project backlog. "The Product Backlog is an emergent, ordered list of what is needed to improve the product. It is the single source of work undertaken by the Scrum Team." [20, p. 10]. For each sprint, tasks are selected from this backlog in alignment with the specific objectives set for that sprint. Additionally, as the project progresses and new tasks emerge, these are added to the backlog.

2.3.1.5 Sprint backlog

"The Sprint Backlog is composed of the Sprint Goal (why), the set of Product Backlog items selected for the Sprint (what), as well as an actionable plan for delivering the Increment (how).

The Sprint Backlog is a plan by and for the Developers. It is a highly visible, real-time picture of the work that the Developers plan to accomplish during the Sprint in order to achieve the Sprint Goal. Consequently, the Sprint Backlog is updated throughout the Sprint as more is learned. It should have enough detail that they can inspect their progress in the Daily Scrum." [20, p. 11].

2.3.1.6 Review

Towards the end of each sprint, the team review the outcome of the sprint. "The purpose of the Sprint Review is to inspect the outcome of the Sprint and determine future adaptations." [20, p. 9]. This meeting provides a platform for the development team and client to discuss what was accomplished during the sprint, and to present work done to the client. This input aids the team in guiding the focus for the subsequent sprint and to identify areas for enhancement in the team's work.

2.3.1.7 Retrospective

At the conclusion of each sprint, the team engages in a review, not of the result, but of the process of the sprint. This review, known as the retrospective, is an introspective exercise centred on the team's collaboration and execution methodologies. "The purpose of the Sprint Retrospective is to plan ways to increase quality and effectiveness." [20, p. 10]. Moreover, the team considers the previous sprint's scope – evaluating whether it was overly ambitious, undersized, or balanced. This assessment is used to plan the forthcoming sprint, ensuring a balances and achievable set of objectives.

2.3.1.8 Daily stand-up

The daily stand-up is a brief, internal meeting that includes all members of a project team. This meeting is designed to be concise, typically lasting no loner than five minutes. Its purpose is to ensure that all team members are kept up to date on current progress and any deviations or modifications to the project plan. The meeting is intended to be purely informative; therefore, extensive discussions are to be avoided.

2.3.1.9 Roles

"Scrum has three roles: product owner, scrum master, and the development team members." [21]. The product owner is tasked with the prioritization of tasks within the backlog based on the costumer and business requirements. The scrum master's role is to ensure that the team is following the SCRUM methodology. The Development Team consist of professionals who are responsible for the execution of tasks, delivering potentially releasable increments of the product at the conclusion of each sprint.

2.4 Data Analysis

2.4.1 Data Attributes

Data is an atomic unit of information. Data is meaningless until context is added to it. When data comes with context, it is possible to gather insight from it. An important way context is added to data is when data is grouped and labeled such that one may view data as assignment of a property. Different groups of data are known as attributes.

Attributes represents different properties of the object. Attributes can be of different nature, and therefore, it is convenient to define attribute categories. [22, p. 21] defines the following three categories:

***Continuous**: A continuous attribute is on which, if it can take values a and b, then it can also take any value between a and b.*
***Discrete**: A discrete feature is one where if it can take a value a, then there is a positive minimum distance to the nearest other value b it can take.*
***Binary**: A binary feature is a discrete feature which can only take two values. Usually these are denoted 0 (false) or 1 (true).*

Additionally, attributes can be distinguished into four different types of attributes. [22, p. 21] defines the these attribute types as follows:

***Nominal**: If the variable is not ordered and only uniqueness matters. An example is the country of origin or the id.*
***Ordinal**: If the variable is ordered (smaller, larger). An example is a safety rating.*
***Interval**: If the variable is ordered and the relative magnitude of the variable*

has a physical meaning.

***Ratio**: If the value 0 of the variable has a specific, physical meaning. I.e. it makes sense to say one value of the variable is "twice as large" as another.*

A dataset is likely to contain several attributes which belong to different categories and types. When exploring and analyzing a dataset, it is important to recognize its attributes and the distinction between them.

2.4.2 Types of Datasets

"A dataset can be thought of as standardized measurements of objects of the same basic type." [22, p. 19]. Datasets are often on a format which resembles the object-relational database model, and consists of columns representing attributes of the object, and rows representing different instances i.e. it can be described as an $N \times M$ matrix where N is the number of instances and M is the number of features.

There are however several types of datasets. The one mentioned above is known as a record dataset. Other types are relational datasets, which is a collection of data objects and is represented as a graph, and ordered datasets, which is an ordered collection of data objects, and is represented as a time-series sequence.

2.4.3 Data Quality

Datasets are often incomplete or contains data which does not represent the phenomenon in a proper way. One may regard the presence of such issues as the quality of the data. Data is of high quality if it is consistent and capable of representing the phenomenon they correspond to in an accurate manor. Some examples of quality problems of data are:

Noise: Data is technically correct, but it is distorted. Noise in datasets could indicate that there are some issues with the measurement accuracy, or that there is an interference from other sources. In order to deal with noise in data, one can attempt to remove it through filtering.

Outliers: Some of the data is significantly different from the other values. This can occur due to measurement errors, but it does not necessarily has to be a mistake. It is important to consider whether it is a measurement error or if it is a natural phenomenon. In order to handle unwanted outliers, one can identify and correct them. A common way to correct outliers in data is to replace the value with the mean value of the data of the same property.

Missing values: Datasets are often incomplete due to missing values. This could occur due to measurement errors or if the data attribute is not applicable to that particular instance. Ways to handle missing values in datasets include excluding incomplete instances or assign an estimated value in its place.

When training a ML model, it is important to make sure to feed it with data of high quality. In order for the model to learn, predict, and classify data, it is important that the data it is trained on is as accurate as possible, since the quality of the input will affect the accuracy and relevance of the output. The process of cleaning a dataset and resolve quality issues in it is known as data preprocessing.

2.4.4 Preprocessing

"Data mining is a methodology in computer science for discovering meaningful patterns and knowledge from large amounts of data. However, before a data mining model can be applied, the raw data must be preprocessed to ensure that it is in a suitable format for analysis. Data preprocessing is an essential step in the data mining process and can greatly impact the accuracy and efficiency of the final results." [23, paragraph 1].

Preprocessing can be considered to be an art in itself. The necessary measures needed to ensure that a dataset has good quality depends on the dataset itself, what it will be used for, and how it will be used. It is in other words a subjective task. A convenient strategy could be to ask the following questions:

1. What will the dataset be used for?
2. Which of the attributes in the dataset are relevant?
3. Is the dataset readable and easy to understand? (e.g. does the attributes have a meaningful label?)
4. Are there any missing values in the dataset?
5. How should outliers be classified and handled?
6. Should noisy data be filtered?
7. Are any of the attributes completely independent? (e.g. are any attributes completely irrelevant or random?)
8. Is there any imbalance in the data? (e.g. duplicate instances, uneven ratio between different attribute values?)
9. Are there any text attributes which could be represented by a numeric value? (e.g. replacing an enum attribute $A_1 = \{"low", "medium", "high"\}$ by an ordinal attribute $A_2 = \{1, 2, 3\}$)
10. Should a new attribute be introduced?
11. Could some attributes be merged together? (e.g. replacing attribute *height* and *length* with attribute *area* = *height* x *length*)
12. Should the data be scaled or normalized?
13. Should the dataset be projected into lower dimensional subspace? (e.g. using Principal Component Analysis to reduce $\vec{v_1} = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix}$ into $\vec{v_2} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$)

This line of questions is an attempt at proposing a set of general guidelines which is applicable in most situations. By asking the above questions and acting upon them, one can be reasonably confident that the dataset is properly preprocessed and holds good quality. One should note, however, that preprocessing is a continuous process, and that it may be necessary to conduct more preprocessing in several iterations depending on the project.

2.4.5 Scalers

2.4.5.1 Purpose

Scalers are a technique used to normalize the range of independent variables or features of data. Independent variables come in different types, including numerical ones like age *(ranging from 0 to 100 years)*, *salary (in the range of thousands)*, dimensions *(with decimal points)*, and many more. We don't want our machine learning model to mistake a feature with a larger magnitude as being better than others. Scaling in machine learning helps to bring all independent variables to the same range, for example, centered around a particular number (like 0) or in the range (0,1), depending on the scaling technique used [24].

2.4.5.2 Min-Max Scaler

This technique scales a feature or observation value between 0 and 1 using min-max scaling.

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{2.1}$$

In Formula 2.1, x is the feature, and *min()* and *max()* are the minima and maximum values of the feature [24].

2.4.5.3 Standard Scaler

To standardize a feature value such that it has a distribution with a mean value of 0 and a variance of 1, we can use a formula called feature scaling. Formula 2.2 describes how an attribute is standardized:

$$x' = \frac{x - \bar{x}}{\sigma} \tag{2.2}$$

Here, σ represents the standard deviation of the feature vector, and $\bar{x}$ represents the average of the feature vector. This process of scaling is important to ensure that the features have a comparable scale, which is necessary for certain machine learning algorithms to work efficiently [24].

2.4.5.4 Robust Scaler

RobustScaler is a technique used to scale features in a way that is robust to outliers. This algorithm removes the median and scales the data based on the quantile range, which is typically the Interquartile Range (IQR), [25]. Formula 2.3 describes this scaling method [26].

$$x_{robust} = \frac{x - median(x)}{IQR(x)} \tag{2.3}$$

2.4.5.5 Normalization Scaler

This normalization will create the distribution of features between [-1, 1] by dividing by the standard deviation [27].

$$x' = \frac{x - mean(x)}{max(x) - min(x)} \tag{2.4}$$

Formula 2.4 describes how the normalization works. Here x' denotes the normalized feature.

2.4.6 Correlation

Correlation is an important concept in statistics. The correlation-coefficient R is a measure of how much two datasets relates to each other. The correlation-coefficient is a number such that $-1.0 \leq R \leq 1.0$. An R close to 1 indicates a strong, positive relationship between the sets, while an R close to -1 indicates a strong, negative relationship. If $R \approx 0$ it indicates that there is no relationship present.

$$R = \frac{S_{XY}}{S_X S_Y} = \frac{\sum_i \left[(X_i - \bar{X})(Y_i - \bar{Y})\right]}{\sqrt{\left[\sum_i (X_i - \bar{X})^2\right]\left[\sum_i (Y_i - \bar{Y})^2\right]}} \tag{2.5}$$

The correlation-coefficient, R, can be calculated using Formula 2.5. Here, X_i and Y_i denotes the value at index i in the set of X and Y, while $\bar{X}$ and $\bar{Y}$ denotes the mean value of X and Y respectively.

While correlation is a good indicator to use in statistics and ML, it is important to know that **correlation does not equal causality**. There is always a possibility that there is no real relationship between two datasets even though they have a strong correlation.

2.4.7 Data Profiling

Data profiling is the process of studying and evaluating data to gather statistical information about its quality. It begins by examining the available data and its attributes. Data scientists identify relevant data sets for the problem they are trying to solve, take stock of important features, and create a hypothesis of attributes that could be useful for the proposed analytics or machine learning task, according to [28].

2.4.8 Data Cleansing

Data cleansing is a way to ensure high-quality data is to eliminate bad data, fill in missing data, and verify the suitability of the raw data for feature engineering, according to [28].

2.4.9 Feature Engineering

"Feature engineering is the process of creating new features or transforming existing features to improve the performance of a machine-learning model. It involves selecting relevant information from raw data and transforming it into a format that can be easily understood by a model. The goals is to improve model accuracy by providing more meaningful and relevant information" [29].

2.5 Machine Learning

Machine Learning (ML) is according to [30, paragraph. 1] a smaller branch of computer science and artificial intelligence (AI), as illustrated in the Figure 2.5.1. Essentially, machine learning focuses on using the data and algorithms to enable AI to imitate the way that humans learn, gradually improving its accuracy.

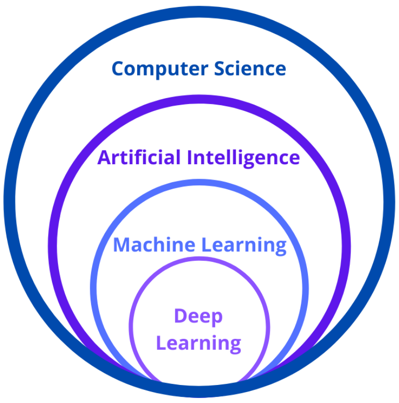


Figure 2.5.1: Machine learning concept [31]

2.5.1 Supervised v.s. Unsupervised Learning

Supervised and unsupervised learning are two distinct approaches to artificial intelligence and machine learning. They differ primarily based on the type of data they use. In supervised learning, algorithms are trained using labeled data to make predictions and improve accuracy over time. The labels in the data serve as guides for the algorithm, providing expected outcomes that help it learn the correct response according to [32].

On the other hand, unsupervised learning uses unlabeled data and allows algorithms to explore the data independently. This method enables algorithms to identify underlying patterns without predefined answers or guidance according to [32].

2.5.2 Neural Networks

"A neural network is a machine learning program, or model, that makes decisions in a manner similar to the human brain, by using processes that mimic the way biological neurons work together to identify phenomena, weigh options and arrive at conclusions." [33, paragraph. 1]. An NN consists of artificial neurons, usually referred to as nodes, which appears in layers as illustrated in Figure 2.5.2.

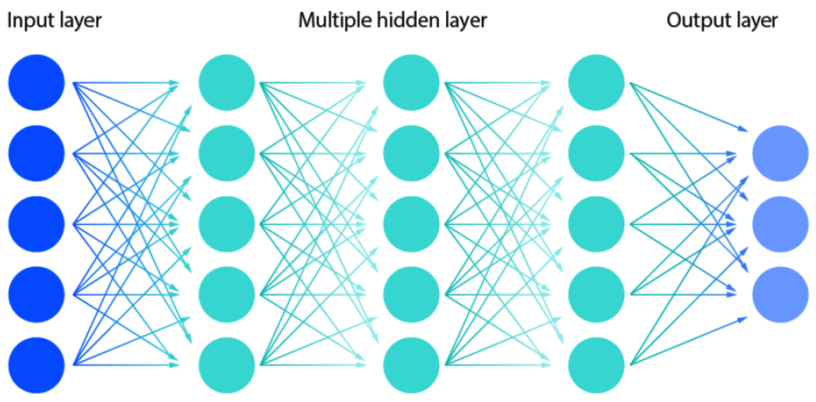


Figure 2.5.2: Neural network illustration
[34]

Each node is connected to other nodes in the network, and can be regarded as a mathematical function which activates the neuron on a given condition when receiving data. The condition is usually a threshold value. The neurons also holds a weight variable which is adjusted when receiving data in order for the model to simulate a learning process. As [33, paragraph. 2] states, a neural network relies on training data in order to learn from it and improve its accuracy.

The weight is a crucial component for the learning mechanism. It's function in a neural network can be described as follows: "When a neural net is being trained, all of its weights and thresholds are initially set to random values. Training data is fed to the bottom layer — the input layer — and it passes through the succeeding layers, getting multiplied and added together in complex ways, until it finally arrives, radically transformed, at the output layer. During training, the weights and thresholds are continually adjusted until training data with the same labels consistently yield similar outputs." [35, paragraph. 9].

2.5.3 Recurrent Neural Networks

"Recurrent Neural Network(RNN) is a type of Neural Network where the output from the previous step is fed as input to the current step." [36, paragraph. 1]. It can be considered to be a specific NN which is able to propagate the models output as a new input. The most important feature which separates an RNN from a traditional NN is that the RNN has a memory state, which enables the model to somewhat recall information about a sequence of data as it traverses through the network. This makes it possible for an RNN to learn historical dependencies in sequential data to a certain degree.

RNN architectures are widely used due to its capabilities, and can be applied to different ML tasks such as natural language processing, speech recognition, and anomaly detection.

2.5.4 The Vanishing Gradient Problem

As stated by [37, paragraph. 2], the vanishing gradient problem occurs "when the gradients of the loss function with respect to the weights of the early layers become vanishingly small". The consequence of this is that the early layers of a NN receives a bare minimum of weight updates during back-propagation, which eventually leads to a convergence or complete stagnation. This can in some cases result in a model being unable to output anything but a constant value regardless of the input it receives.

[37, paragraph. 2] states that the vanishing gradient problem occurs due to a variety of factors such as the choice of activation functions and optimization algorithms. The ML models which are most vulnerable to the vanishing gradient problem are NN's which utilizes back-propagation, leaving this a common issue for RNN architectures. The vanishing gradient problem can be detected by inspecting the weights of a model during training, where the absence of weight updates is a strong indication of vanishing gradient.

2.5.5 Long Short-Term Memory Models

The Long Short-Term Memory model is an improved version of traditional RNN models. "LSTM is well-suited for sequence prediction tasks and excels in capturing long-term dependencies. Its applications extend to tasks involving time series and sequences." [38, paragraph. 1].

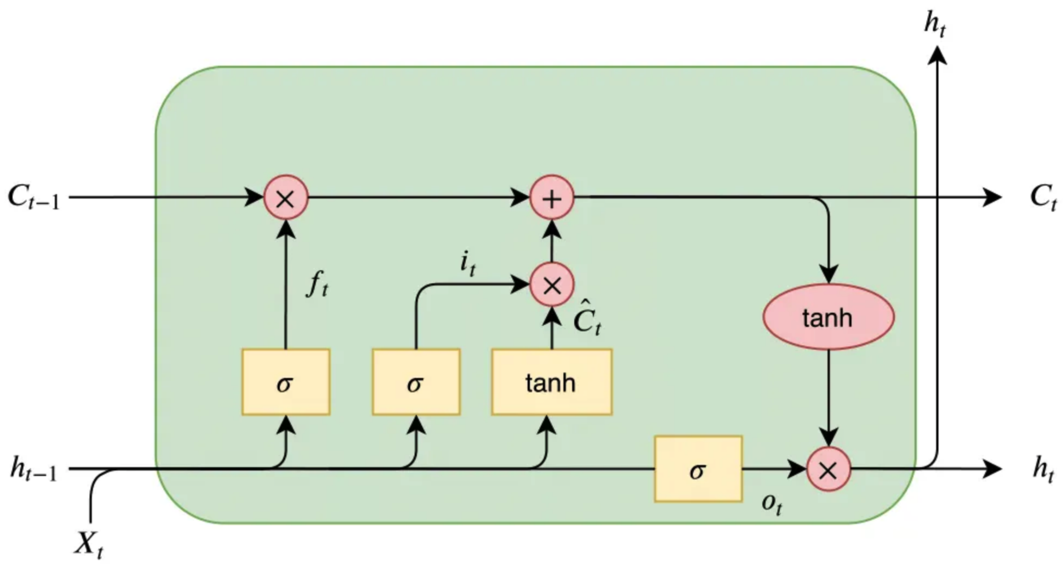


Figure 2.5.3: LSTM illustration
[39]

The illustration in Figure 2.5.3 represents a memory block. These memory blocks, or cells, are chained together in the LSTM architecture. "Information is retained by the cells and the memory manipulations are done by the gates." [38, paragraph. 4].

LSTM has several advantages to traditional RNNs. The most notable are its ability to capture long-term dependencies in data, its ability to process sequential data in both directions, and its insensitivity to the vanishing gradient problem. These advantages does however come at the cost of a more complex model which requires more computational resources and longer training times.

2.5.5.1 Bidirectional LSTM

A bidirectional LSTM is an enhancement of a traditional LSTM design. [38, paragraph. 3] states that a bidirectional LSTM has the ability to process sequential data in both forward and backward directions, and that this enables the model to learn even longer ranged dependencies in the data than a traditional LSTM, which would only be able to process sequential data in only the forward direction.

2.5.5.2 Gates

2.5.5.2.1 Forget Gate The forget gate is used to remove information which is no longer relevant in the cell state. This gate makes it possible for the model to discard information learned from patterns in long-term and short-term trends in the data. "Two inputs x_t (input at the particular time) and h_{t-1} (previous cell output) are fed to the fate and multiplied with weight matrices followed by the addition of bias. The resultant is passed through an activation function which gives a binary output." [38, paragraph. 5].

2.5.5.2.2 Input Gate The input gate is responsible for receiving and adding information to the cell state. "First, the information is regulated using the sigmoid function and filter the values to be remembered similar to the forget gate using inputs h_{t-1} and x_t. Then, a vector is created using *tanh* function that gives and output from -a to +1, which contains all the possible values from h_{t-1} and x_t. At last, the values of the vector and the regulated values are multiplied to obtain the useful information." [38, paragraph. 6].

2.5.5.2.3 Output Gate The output gate is responsible for extracting information from the current cell state and returning it as the output. "First, a vector is generated by applying *tanh* function of the cell. Then, the information is regulated using the sigmoid function and filter by the values to be remembered using inputs h_{t-1} and x_t. At last, the values of the vector and the regulated values are multiplied to be sent as an output and input to the next cell." [38, paragraph. 7].

2.5.6 Loss Functions

The loss function, also referred to as the error function, is a crucial component in the machine learning that quantifies the difference between the predicted outputs of a machine learning algorithm and the actual target values.

The resulting value, the loss, reflects the accuracy of the model's predictions. During training, a learning algorithm such as the Mean Squared Error is used to constantly validate the model's accuracy and adjust the weights.

As stated by [40, paragraph. 1], the loss function typically decreases when the model is training as it learns to fit the training data. Furthermore, that a decreasing loss function does not necessarily guarantee that the model gets better at generalization, but rather that the model is overfitting (See section 2.5.10.3).

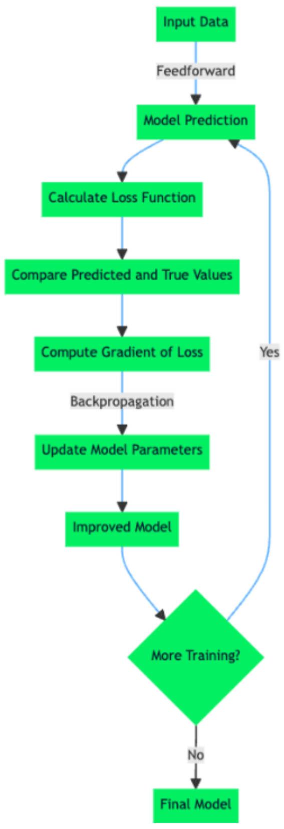


Figure 2.5.4: Mechanics of loss function

[41]

2.5.7 Validation Loss

"Validation loss is a metric used to assess the performance of a deep learning model on the validation set." [42]. Furthermore, [42, paragraph. 3] states that the purpose is to monitor the model's performance and to identify overfitting.

[40, paragraph. 1] states that during training, the validation loss may show different patterns indicating how the model fits with the data. If the validation loss decreases, it could indicate that the model is underfitting, while increasing validation loss could indicate overfitting. A stable validation loss that the model fits well with the data. It is also stated by the same reference that in terms of detecting overfitting, that if the validation loss suddenly starts to increase after a certain point may be an indication of overfitting, prompting adjustments in the model's complexity or regularization.

If the distance between the loss and validation loss is narrowing, it could be an indication that a good fit is achieved. But if the loss starts to be smaller than validation loss, it could indicate that the model has started overfitting instead.

2.5.8 Anomaly Detection

[43, paragraph. 1] states that anomaly detection is the process of identifying rare occurrences or observations that deviate significantly from the majority of the data and do not follow a well-defined notion of normal behavior. This technique finds its application in various domains such as cyber security, medicine, machine vision, neuroscience, law enforcement and financial fraud detection, among others.

2.5.9 Classification

[44, paragraph. 3] states that classification is a type of supervised machine learning technique where a model is trained to correctly label input data. After being fully trained on a set of training data, the model is evaluated on test data to ensure its accuracy before being used to predict labels for new, unseen data.

Classification can be used for several purposes, e.g. predicting how the weather will be based on environmental data, classifying heart diseases based on blood pressure, heart rate, etc. and classifying what species a flower belongs to based on the length of its petals and sepals.

2.5.10 Generalization

[45, paragraph. 24] states that generalization refers to the ability of a model to perform well on new data, which is crucial if we want to use the model in the real world. On the other hand, a model that doesn't generalize well may perform well on the data it was trained on but may not make accurate predictions on new data. This is a significant problem because it means the model may not be useful on practice.

2.5.10.1 Bias

[46, paragraph. 3] states that a bias in machine learning is simply defined as the inability of the model, due to differences or errors occurring between the model's predicted value and the actual value. These differences are known as error or bias error due to bias. Bias is a systematic error that occurs based on wrong assumptions in the machine learning process.

2.5.10.2 Variance

Based on the information from [46, parapgraph. 6] it states that the variance measures the spread of data from its mean. It refers to the change in a predictive model's performance when trained on different subsets of the training data.

2.5.10.3 Overfitting

Overfitting is a common problem that occurs when a model becomes too complex and starts to fit the training data to closely. This means according to [45, paragraph. 3] that the model may not perform well on new, unseen data because it has essentially memorized the training data instead of truly learning the underlying patterns or relationships. In other words, the model is too good at learning from the training data but not so good at generalizing to new data.

2.5.10.4 Underfitting

The opposite of overfitting in machine learning is underfitting. According to [45, paragraph. 5] underfitting refers to a situation where the model is too simple for the task at hand. The model lacks the necessary complexity to capture the underlying patterns in the data.

2.5.11 Accuracy Scores

2.5.11.1 Data Visualization

Data visualization means showing data or information using graphs, charts, or other pictures. It helps us understand how data is related by using images. Turning information into pictures makes it simple to spot patterns, trends, and outliers compared to scanning rows of data on a spreadsheet. Since the aim of data is to uncover insights, visualized data is way more useful, according to [47].

2.5.11.2 Mean Squared Error

[48] states that Mean Squared Error is a metric for evaluating the performance of predictive models. It measures the averages squared difference between the predicted and the actual target values withing a dataset. The primary objective of the MSE is to assess the quality of a model's predictions by measuring how closely they align with the ground truth.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.6}$$

Formula 2.6 above denotes how the Mean Squared Error is calculated. Here, Y_i denotes the real value of an attribute, while $\hat{Y}_i$ denotes the predicted value.

2.5.11.3 R-squared

According to [49], r^2, also known as the coefficient of determination, is a statistical measure used in machine learning to assess the quality of a regression model. Essentially, it determines how well the model fits the data by measuring the proportion of variance in the dependent variable that is explained by the independent variables.

The r^2 score is essentially a number between 0 and 1, where a score of 1 means that the regression fits perfectly with the data. It is also possible to get a negative r^2 score. This happens if the model fits so poorly with the data, that a horizontal line would fit better than the model.

2.6 Ship Engines

2.6.1 Four-stroke Engines

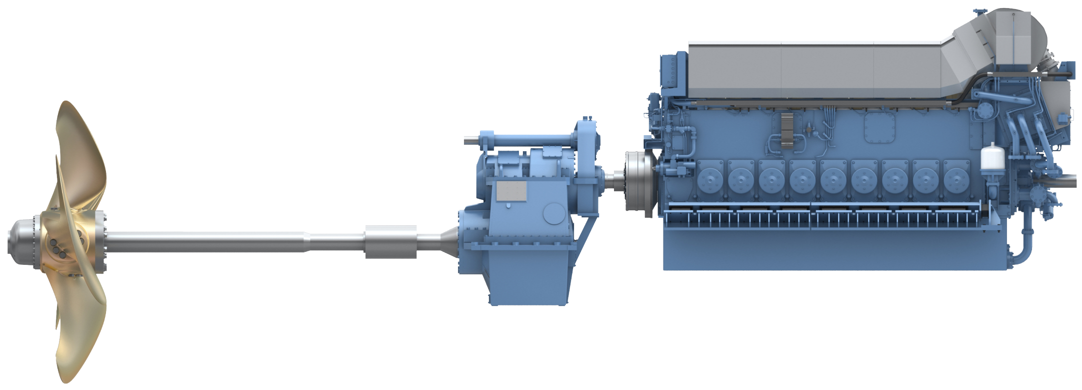


Figure 2.6.1: Rolls-Royce propulsion system
[50]

Combustion engines usually follow either the four-stroke- or the two-stroke principle. Most ships, except the smallest and the largest, has four-stroke engines. [51, p. 36] states that in the four-stroke principle, combustion happens for each fourth stroke. One stroke represents the pistons movement from the upper position to the lower position in the cylinder or vice versa.

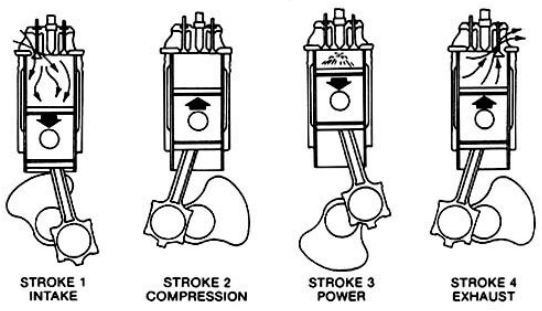


Figure 2.6.2: The four-stroke cycle
[52]

As illustrated in Figure 2.6.2, the four phases of the cycle represents intake, compression, power, and exhaust. The process happening at each phase can be described as follows:

1. **Intake**: The piston moves down, allowing oxygen to enter the cylinder.
2. **Compression**: The piston moves up, compressing the air which increases the temperature inside the cylinder.

3. **Power**: Fuel is injected. The fuel is ignited by the hot temperature inside the cylinder, causing an explosion which pushes the piston downwards.
4. **Exhaust**: A valve is opened while the piston moves upwards, allowing exhaust gas to be pushed out of the cylinder.

2.6.2 Cylinders

The cylinders are the holes in which the pistons are placed. They are drilled into the engine block. A piston is placed inside a cylinder with just enough room for it to be able to move up and down. The pistons connecting rod is attached to the crankshaft, which rotates when the pistons move up and down. The temperature inside the cylinders are high while the engine is running, due to the combustion process taking place. As stated by [51, p. 30], the piston has a smaller diameter than the cylinder, ensuring that the piston does not get stuck when the temperature rises and the metal expands.

2.6.3 Engine Power

"Engine power is the power that an engine can put out. It can be expressed in power units, most commonly kilowatt or horsepower." [53]. The engine power is a measure which affects many attributes such as the speed of the vessel as well as how much load it can carry. The power output of the engine is a dynamic variable which changes depending on how much throttle is used as well as the amount of oxygen and diesel injected into the cylinders.

2.6.4 Lube Oil System

As stated by [54], the lube oil system provides oil to the bearings, gears, and other components at the designated pressure, temperature, and quality. The system comprises an oil tank, pump, cooler, filter, and heater. Its purpose is to ensure that all moving components are lubricated which prevents them from getting stuck. Another feature of the lube oil is transporting heat out of the system.

2.6.5 Fuel Oil System

The fuel oil system is responsible for providing fuel to the engine. The fuel is pumped up from the fuel tank and injected in appropriate doses into the cylinders in the engine. A filter is also present in order to filter out impurities as well as reducing the pressure.

2.6.6 Low-Temperature Cooling System

A low-temperature cooling system is utilized for low-temperature zone machinery, which is directly linked to the primary seawater central cooler. As a result, its temperature is lower than that of the high-temperature (H.T.) circuit. The L.T. circuit encompasses all auxiliary systems, as stated by [55].

2.6.7 High-Temperature Cooling System

The high-temperature cooling system within the central cooling system primarily encompasses the jacket water system of the main engine, where operating temperatures are significantly elevated. This system effectively manages the high-temperature water by utilizing the low-temperature freshwater system for cooling, as stated by [55].

2.6.8 Platform Supply Vessels

"A platform supply vessel (PSV) is a ship specifically designed to supply offshore oil and gas platforms and other offshore installations. They typically range from 50 to 100 metres (160 to 330 ft) in length and are distinguished by the large open deck area used to store supplies and house equipment and to allow for efficient loading and offloading." [56, paragraph. 1].

Figure 2.6.3: UT 7400 - Platform supply vessel [57]

KM has implemented their current CM system on many PSVs originating from different countries and with different objectives. An example of a PSV is shown in Figure 2.6.3. The ships are usually of medium size and consists of either one or two four-stroke, inline diesel-engines. Other notable machines usually present in PSV ships are thrusters, winches, generators, and naturally, the propulsion system.

2.7 Software

2.7.1 Programming

2.7.1.1 Python

Python is one of the most popular programming languages in the world. It contains many features which makes it useful for many sorts of work such as computer science, web-service, ML, and game development. Python is an object oriented programming language with a simple syntax, making it easy to write powerful programs. Like Java, "Python is an interpreted language, which means the source code of a Python program is converted into bytecode that is then executed by the Python virtual machine." [58].

The programming language is one of the most used programming languages for ML and computer science in the world. Due to its many features, its readability, its large community, and support, it is considered to be one of the easiest languages to learn. Python is a suitable choice for ML development, featuring a large portfolio of frameworks and libraries which simplifies the process such as Pandas, NumPy, TensorFlow/Keras, and Scikit-learn.

2.7.1.2 Jupyter Notebook

"JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality." [59, paragraph. 1]. Jupyter Notebooks utilizes the .ipynb file-format which is based on cells in which text can be written, or code such as a Python script which can be executed.

2.7.2 Frameworks & Libraries

2.7.2.1 Pandas

"Pandas is a software library written for the Python programming language for data manipulation and analysis." [60, paragraph. 1]. The library is widely used for dataset operations, offering services such as import/export, statistical methods, data manipulation, plotting, etc. A dataset is either loaded or created in the program and stored as a data structure known as a *pandas.DataFrame.*

2.7.2.2 NumPy

"NumPy is an open source project that enables numerical computing with Python. It was created in 2005 building on the early work of the Numeric and Numarray libraries." [61, paragraph. 1]. It offers a vast amount of functions used for computational science, including functions for statistical computing, mathematical analysis, and quantum computing. In terms of ML, "NumPy forms the basis of powerful machine learning libraries like scikit-learn and SciPy. As machine learning grows, so does the list of libraries built on NumPy." [62, Ecosystem].

2.7.2.3 TensorFlow/Keras

TensorFlow is a library used for creating ML models [63]. It offers support for several programming languages, including Python, JavaScript, and C++. TensorFlow has an API which utilizes Keras [64], making it possible to make a large variety of customized ML models. Together, TensorFlow and Keras offers a large amount of functions and data structures, making it possible to implement ML models such as RNN, LSTM, and traditional NNs.

2.7.2.4 Scikit-learn

Scikit-learn is a Python library used for ML [65]. It is built on NumPy, SciPy and matplotlib, and offers several functions for classification, regression, clustering, data preprocessing, and more. It also contains statistical functions useful for assessing a ML model's accuracy such as MSE and R-squared.

2.8 Finance

2.8.1 Compound Annual Growth Rate

The compound annual growth rate (CAGR) is a metric in business, economics, and investing that quantifies the average annual growth rate of an investment over a specified period, assuming the profits are reinvested at the end of each year, state by Wikipedia [66].

$$\text{CAGR}(t_0, t_n) = \left(\frac{V(t_n)}{V(t_0)}\right)^{\frac{1}{t_n - t_0}} - 1 \tag{2.7}$$

In Formula 2.7, the $V(t_0)$ denotes the initial value, $V(t_n)$ denotes the end value, and $t_n - t_0$ denotes the number of years.

CHAPTER
THREE

METHODS

This chapter contains a review of how the project was conducted and the approaches made along the way. The review includes planning and research, administrative work, data preprocessing, and model training.

3.1 Planning

3.1.1 Research

3.1.1.1 Choosing Google Colab

In addition to Colab's powerful tools, some team members also had prior experience using Google Colab for machine learning purposes. This made it a great candidate as the model development platform. The supervisors support of Google Colab further built on its selection as the preferred programming language.

3.1.1.2 Machine learning algorithms

When researching ML algorithms, the first step was to exclude the algorithms that were not relevant for the situation. In the beginning of the research, the approach were based on existing knowledge of ML. Existing knowledge being former subjects from studies and online courses. Using this knowledge, the development team researched possible methods and algorithms, and discussed these with the supervisors and client. Processing this feedback to create even better plans until the team achieved a plan detailed enough to begin the machine learning process.

3.1.1.3 Data mining

Due to the substantial size of the dataset, extensive research into data mining techniques was essential to effectively sort through and find relationships in the dataset. initially, the group received an information sheet and a detailed presentation from KM to learn the meaning of the variables in the dataset. From this, the group used their knowledge of motor systems and relevant studies to analyse the dataset. Additionally, correlation heatmaps were created to both support and discover new insights and information about the dataset. Then, the research was discussed with KM who provided valuable feedback that guided further investigation. This iterative process of consultation and analysis continued until a conclusion that was well-supported by the data was achieved.

3.1.2 Problem Domain

3.1.2.1 Introduction to Problem Domain

Before beginning development, it was crucial to get a better understanding of the problem domain. Initially, the team got some information in the project proposal. But to further understand the problem domain, the team was introduced to the system, and the impact of anomaly detection by KM.

3.1.2.2 Concerns for the project

When accepting this project as a bachelor's project, it was disclosed that KM had previously attempted to solve this issue in 2016, an effort which ended in failure. Therefore, there was concern that this attempt might also not succeed. One contributing factor to the 2016 failure was insufficiency of the hardware available at the time to develop an acceptable model. Although computer technology has significantly advanced since then, there remained a risk that the hardware might still be too poor to create this model.

In development of a machine learning model, there is always the inherent risk of producing a model that is either inaccurate or biased. Such a model could lead to reduced quality, delayed progress, and failure to meet project objectives. To mitigate these risks, extensive research and model experimentation was undertaken in the planning phase.

3.2 Project Structure

3.2.1 Setting up the environment

The primary challenge encountered in this project was managing the substantial size of the dataset and how to store it. Initially, an attempt was made to upload the dataset directly to the google Colab files folder. However, this approach required that it was uploaded for every connection to the Colab file. Because of the datasets size, this uploading process took too long and was not efficient enough.

Subsequently, the team explored storing the dataset on Microsoft OneDrive and establishing a connection to OneDrive from the Colab terminal. This method, while faster than the latter, also required setup for every connection to the Colab file. Setting up this connection took way to long and was not reliable as it sometimes did not work correctly.

As it turned out, the most effective solution was storing the dataset in Google Drive and then mounting the drive folder in Colab as shown in Listing 3.1.

```python
from google.colab import drive
drive.mount('/content/drive')

PATH = {
    'start': 'drive/MyDrive/dataset/',
    'ship':  '13218' + '.parquet/',
    'year':  'year=' + '2023/',
    'month': 'month=' + '3/',
    'day':   'day=' + '9' + '.parquet'
}

path = PATH['start'] + PATH['ship'] + PATH['year'] + PATH['month']
     + PATH['day']

df = pd.read_parquet(path)
df
```

Listing 3.1: Access Google Drive and define dataset

This solution was easy to implement and very efficient, and therefore proved to be the best method. The initial oversight of this approach was the datasets size, which exceeded the available storage capacity within a team member's Google Drive. After realising that the Google Drive approach was the best solution, all members undertook measures to ensure enough space in Google Drive.

3.3 Problem Approach

In order to implement an anomaly detection system, it was decided that the best strategy would be to group attributes by the system they belonged to, then training a model for each group. The group assumed that this would be a better solution than making a single, large model for all attributes since it would make the implementation easier and possibly result in predictions with higher accuracy. The group assumed that a general model for all attributes would not be as accurate as a model designed for a specific group of attributes. Therefore, the group defined five groups of attributes. These are displayed in Table 3.3.1 and Figure 3.3.1:

Abbreviation	Attribute Group
EX	Cylinder exhaust temperature
HT	High temperature cooling system
LT	Low temperature cooling system
LO	Lube oil system
DO	Fuel oil system

Table 3.3.1: Table of attribute groups and their abbreviations

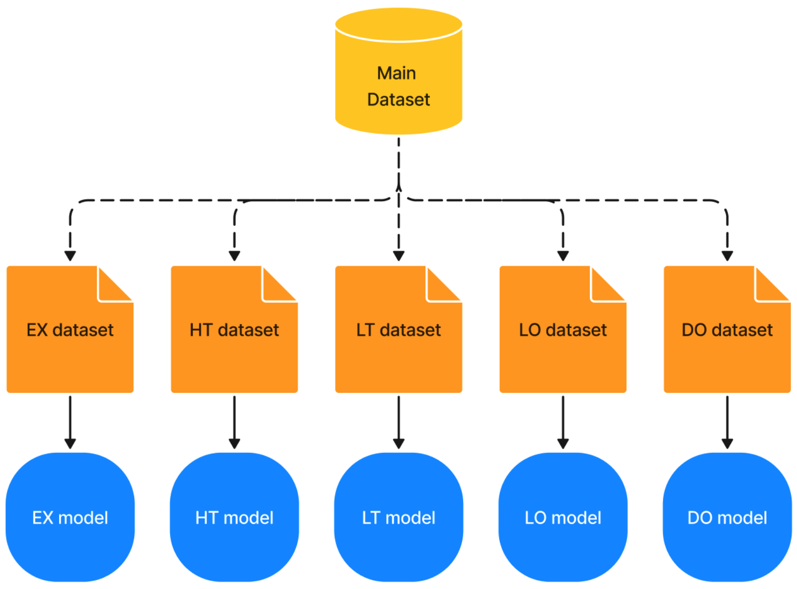


Figure 3.3.1: Illustration of ML system

There are many approaches to anomaly detection. Due to the dataset not containing a label indicating whether there are anomalies present or not, it would be an unsupervised ML problem if one were to approach the problem by training a model to identify anomalies directly. Therefore, the group resorted to an indirect method instead.

The idea was that instead of having a model detect anomalies in an attribute directly from the dataset, it should be able to predict an attribute based on other, related attributes. Given that the model could predict an attribute accurately when there are no anomalies in the data, one could detect an anomaly by comparing the actual values of an attribute with the model's predictions of the same attribute. If the accuracy is below a given threshold, it would indicate that there was an anomaly in the attribute. This approach is illustrated in Figure 3.3.2:

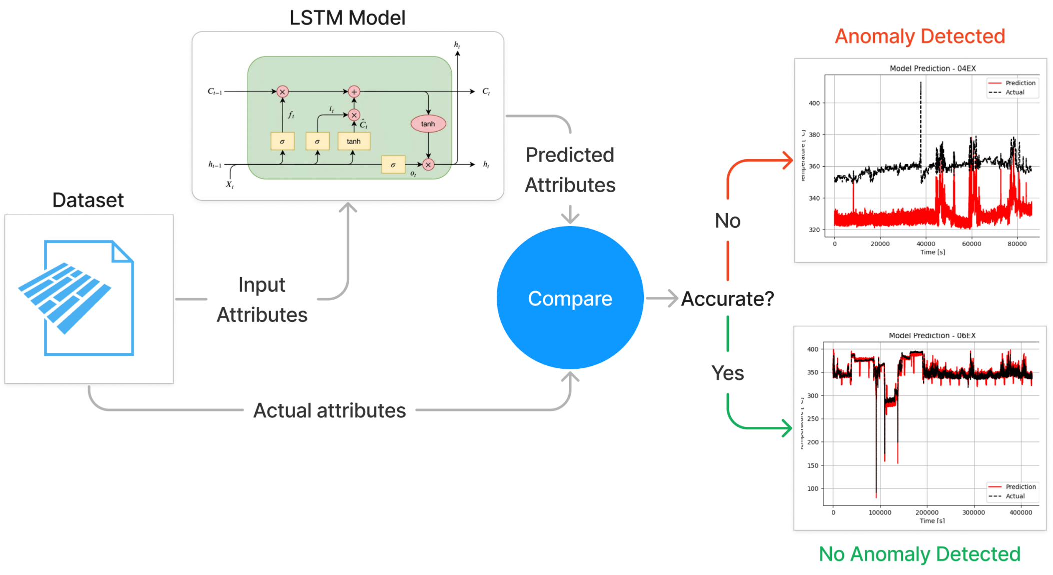


Figure 3.3.2: Illustration of anomaly detection approach

This was considered to be the best approach, and the group decided to use this formulation as a hypothesis which could be investigated following the hypothetical-deductive method (Section 2.1) by attempting to falsify the hypothesis by assessing the results which culminated from the implementation and training of the ML models. By predicting attributes which already exists in the dataset, the problem could be solved using supervised ML, and it would also be easier to train the model and verify its accuracy.

3.4 Execution

3.4.1 Documentation

In the development of a bachelor's thesis, documentation of findings and methodology is crucial for maintaining a clear record and understanding of the process and effectiveness of various approaches. The project management tools Jira and Confluence, were primarily used for this purpose. Jira facilitated issue and sprint management, as well as time logging, while Confluence was utilized to document sprint retrospectives and meeting notes.

Research notes were written and stored in Microsoft Word documents. this was the easiest and best way to make notes while conducting research. The practise was essential for logging findings, and made it possible to recall research details and better understand the reason behind certain choices.

3.4.2 Sprint management

The project management methodology employed is SCRUM, a framework that structures the project into a series of weekly sprints. Each sprint commences with a planning session, during which the team selects items from the product backlog, defines the sprint goal, and outlines the tasks required to achieve this goal. This planning phase ensures that the team has a clear direction and set objectives for the upcoming week. Throughout the sprint, daily stand-up meetings are held. These brief, focused meetings allowed team members to discuss their progress, and identify any troubles.

The sprint concludes with a sprint review and retrospective. In the sprint review, the team demonstrated the completed work to the client from KM and/or to the supervisor, gathered feedback, and discussed any changes or further work on the ML model. Then the team reflected on the sprint and the teams working style, writing down what went good and what could have been done better. These points were then considered and improved on for the next sprint.

3.4.3 Workflow

As mentioned in sprint management, the team operated on a weekly sprint basis, with each sprint defined by a specific completion goal. These sprints were composed of numerous smaller objectives that collectively aimed to achieve the desired outcome. The team selected relevant issues from the product backlog and moved them into the scrum board, both of which were hosted in Jira.

Each team member selected an issue, assigned it to themselves, and moved it from the "To Do" to the "In Progress" column. Once a team member felt confident that the issue was completed, they moved it to the "Review" column. Another team member then reviewed their work, and if they were satisfied, they moved the issue to the "Done" column, thereby ending the life-cycle of the current issue. If absolutely necessary, the issue could be moved back to "In Progress" or "Review". However, in most cases, a new issue was created to address any required changes or further improvements in the completed issue.

3.4.4 Repository

As the development environment selected for the project was Google Colab, the associated Colab files were saved in Google Drive. These files were placed in a shared folder, accessible to the entire development team. Additionally, all functioning Colab files were backed up in a repository on Github, ensuring data integrity and availability. A down-side of this solution was that there were no way to subdue the project to more traditional version control systems such as GIT, as this was done in Colab by automatically replacing the entire file rather than the internal blocks of a file upon changing them.

3.4.5 Meetings

The development team conducted weekly sprint meetings every Friday, alternating between the supervisor and the project client. This pattern persisted throughout the duration of the project, with exceptions made for holidays and other specific cases. Most meetings were held in person although some took place via Microsoft

Teams meeting platform. Each meeting designated a secretary from the development team to take notes during the meeting.

The meetings often started with an update on the progress made since the previous meeting, followed by feedback on that progress. Then, the development team presented prepared questions for discussion. The meetings concluded with directives to be completed or tested before the next meeting.

The development team usually planned which day they were to work on the project in order to have the members work together rather than individually. At the days in which the group worked together with the project, a daily stand-up meeting was conducted. This type of meeting were usually short and executed in person. The typical agenda included what each member planned to work with that day, suggestions, feedback, and status updates.

3.5 Data Collection

The dataset utilized for this thesis was provided by Kongsberg Maritime, derived from a platform supply vessel, specifically designed to support offshore oil and gas platforms, as explained in Section 2.6.8. The vessel is equipped with two diesel engines, labeled 13218 and 13219. The data collection is from 17 January 2023 to 3 January 2024 for both of the engines.

The dataset contained sensor measurements of different components in the engine such as temperature, the pressure of fluids, the temperature of gasses, the RPM of the crankshaft, turbocharger, status codes, etc. These measurements were logged for each second throughout the day, resulting in datasets divided into days. The dataset was structured as a large table and each row was marked with a timestamp. Per definition (see Section 2.4.2), the datasets were therefore considered to be an ordered dataset.

Upon receiving the dataset, the group found it convenient to find a way to merge several datasets, making a single dataset spanning over e.g. a month. This made it possible to perform statistical analysis of data over a longer time span, which would give a more accurate representation of the nature of the data and long-term dependencies.

3.6 Data Preprocessing

3.6.1 Data profiling

The first step in the data preprocessing involved a detailed examination of the dataset to assess its quality and structure. During this phase, the group discovered numerous inequalities, including a significant number of difficult-described attributes, the absence of some values, as well as the large size of the datasets.

The option to utilize either Google Colab or the IDUN cluster [67] at NTNU for managing this dataset was considered. Ultimately, Google Colab was chosen due to the group's familiarity with the platform and its cost-effectiveness.

Upon encountering many unknown attributes in the dataset such as *01EX*, *06HT*, and *10LT*, the group consulted the client at KM. The provision of a tag description for these variables proved to be very valuable, as it facilitated the recognition and understanding of the dataset's various components.

During profiling, several quality issues were identified in the dataset, where the most significant were:

- **Missing values:** There were several instances of missing data across multiple attributes.
- **Unnecessary Attributes:** Some data fields were deemed extraneous and irrelevant for our analysis.
- **Non-Chronological Timestamps:** The timestamps of sensor readings were not always in chronological order, which could potentially make time series analysis unnecessarily complicated.

3.6.2 Data cleansing

3.6.2.1 Sorting and Formatting

As seen in Listing 3.2, the initial action performed on the dataset was making sure the data were sorted in chronological order. Furthermore, the *time* column was formatted into a standard DateTime format and set as the index column, which facilitated time-series analysis and other time-based operations:

```
# Implemented in utility script (see Appendix F)
df = df.sort_values(by='time')
df['time'] = df['time'].dt.strftime('%Y-%m-%d %H:%M:%S')
df = df.set_index(pd.DatetimeIndex(df['time']))
```

Listing 3.2: Data sorting and formatting

3.6.2.2 Decreasing the number of attributes

As mentioned, the dataset contained several attributes that either were irrelevant to the problem at hand, had non-numerical data types representing status codes, or contained a fixed attribute value that never changed. Some examples of such attributes are *Message counter*, *Governor clutch status input*, and *Governor start/max fuel limit active* which were boolean values. It was the intention to focus the analysis on quantitative insights, where only attributes of the continuous category and the ratio attribute type (see Section 2.4.1). This step simplified the dataset by excluding the columns that were not relevant for statistical analysis and machine learning modeling.

```python
def produce_modified_dataset(df: pd.DataFrame, x_attr: list,
    y_attr: list) -> tuple:

    for column in df.columns:
        if len(set(df[column].values)) == 1:
            if any(element == column for element in x_attr):
                x_attr.remove(column)
            if any(element == column for element in y_attr):
                y_attr.remove(column)

    numeric_columns = df.select_dtypes(include=[np.number])
    controlled_parameters = list(set(x_attr) & set(numeric_columns
    .columns.tolist()))
    controlled_parameters.sort()
    columns = controlled_parameters + y_attr

    return df[columns], x_attr, y_attr, controlled_parameters
```

Listing 3.3: Excluding irrelevant columns in the dataset

By implementing this code the total number of columns was reduced from 101 to 64. This made it easier to focus on the attributes which were subject to further investigation. Listing 3.3 above is a function from the utility script (see Appendix F)

3.6.2.3 Handling Missing Values

As mentioned in Section 2.4.3, ensuring high-quality data involves identifying and addressing missing values present in the dataset. Missing values were identified for each column using the script in Listing 3.4:

```python
# Locating missing values for each column
droping_list_all = []
for j in range (0,64):
  if not df.iloc[:,j].notnull().all():
    droping_list_all.append(j)
droping_list_all
```

Listing 3.4: Identifying missing values

For each column, the missing values were replaced with the mean value of the same column, ensuring that no data point was left blank. This was done as follows in Listing 3.5:

```python
# Filling missing values with mean column value
for j in range (0, 64):
  df.iloc[:,j]=df.iloc[:,j].fillna(df.iloc[:,j].mean())
```

Listing 3.5: Missing values columns filled with mean

3.6.2.4 Verification of data

After addressing and resolving important issues in the dataset, the group inspected the dataset's structure, the data types present in the different attributes, and to confirm the absence of missing values. This was done using the script in Listing

3.6. This confirmed that the dataset was adequately preprocessed and ready for further analysis:

```python
# Returns the first n rows for the object
df.head()

# Prints information about the DataFrame
df.info()

# Returns a series with the data type of each column
df.dtypes

# Returns the single integer value representing the total number
   of elements in the DataFrame.
df.shape

# Returns description of the data in the DataFrame
df.describe()

# Shows the columns labels of the data frame
df.columns

# Counting the number of missing values
df.isnull().sum()
```

Listing 3.6: Verification of data

These operations were performed to ensure that the dataset was properly cleaned, organized, and prepared for analysis and modeling.

3.7 Feature Engineering

3.7.1 Attribute Selection

3.7.1.1 Selection Criteria

In developing accurate predictive models for the different subsystems such as the Cylinder Exhaust Gas Temperature, High-Temperature Cooling System, Low-Temperature Cooling System, Lube Oil System, and Fuel Oil System, the selection of attributes was crucial. The approach involved selecting the attributes which were most relevant for predicting the behavior of each system under different operational conditions. This approach was based on both empirical data analysis and theoretical considerations to ensure accuracy and relevance.

The primary criteria for selecting attributes were based on two factors:

- **Correlation Analysis:** Correlation analysis was implemented to identify strong correlations between various sensors. Attributes with high correlation coefficients were considered as they were likely to influence or reflect the system's state. Meeting these criteria alone was however not sufficient for determining which attributes to include in the models.

- **System Knowledge/Expertise from Kongsberg Maritime:** Insights from the client at Kongsberg Maritime helped validate the relevance of selected attributes. This ensured that the attributes selected had some kind of relevance and connection to each other. In several cases, it was possible to isolate a target component subject to condition monitoring by using attributes that were directly affected by the component.

3.7.2 Attributes for Different Systems

3.7.2.0.1 Cylinder Exhaust Gas Temperature (EX) The dataset contains sensor measurements of an engine that has 8 cylinders. The most common anomaly to appear in the EX system is that one of the cylinders experiences a different workload than the other cylinders, which can be identified by inspecting the exhaust gas temperature exiting each of the cylinders.

In an optimal situation, the exhaust gas temperature of all cylinders will fluctuate in complete harmony with each other, meaning that the correlation coefficient between the cylinders should be close to 1. To detect if a cylinder has anomalous behavior, one could therefore try to predict each attribute *01EX*, *02EX*, ..., *08EX*.

To make a model able to predict these attributes, it will need at least one attribute as input which can explain what the expected exhaust gas temperature for each cylinder should be. The attribute that best represents this dependency is *87XS*, which represents the engine's power percentage. Other attributes such as *21EL* and *25EL* representing the RPM of the crankshaft and the turbocharger respectively were also considered, but after consulting KM it was decided that this would likely be a disadvantage since these two attributes usually have a constant RPM which is not affected by the load or speed of the engine.

The relationship between the selected attributes and *25EL* is depicted in the correlation heatmap in Figure 3.7.1 below. As it turns out that *25EL* in reality is not related to the EX system, the high correlation coefficients for that attribute underline the fact that correlation does not equal causality.

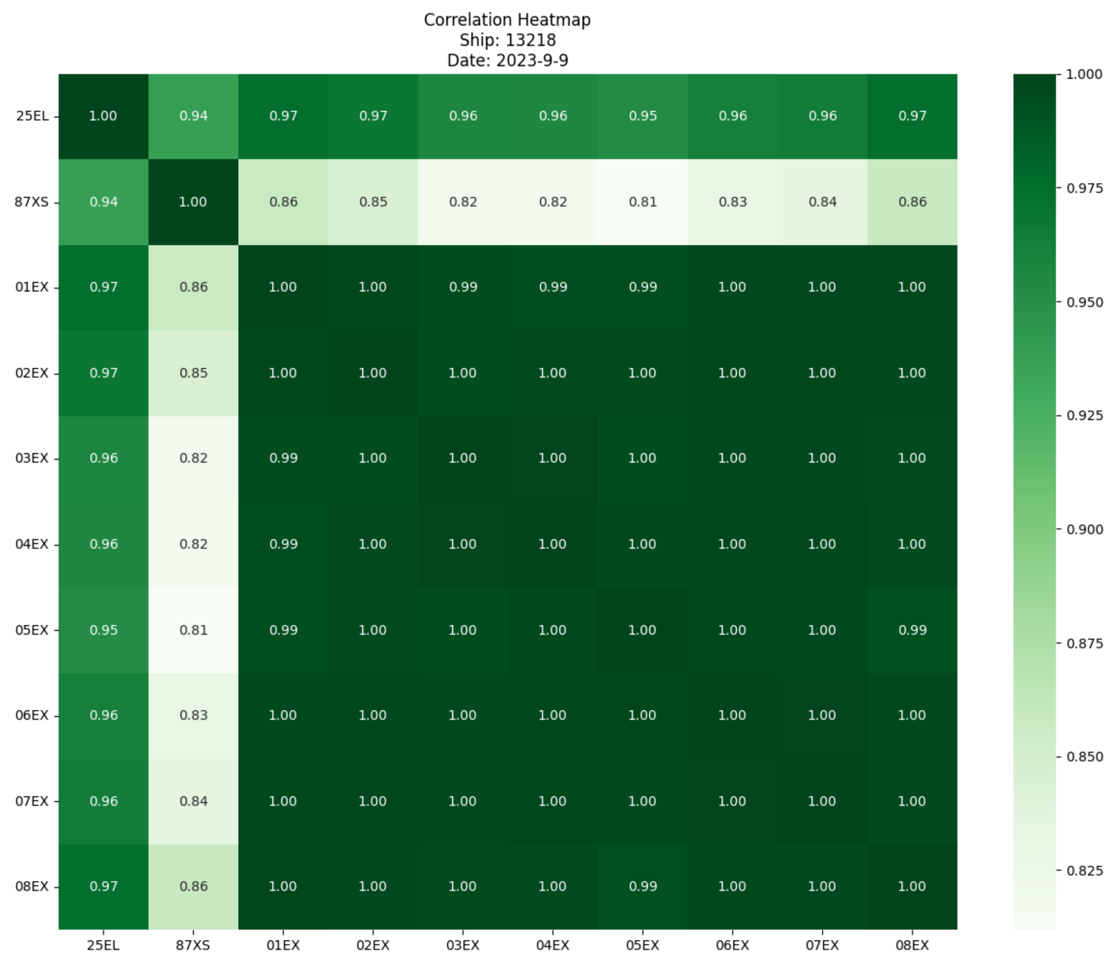
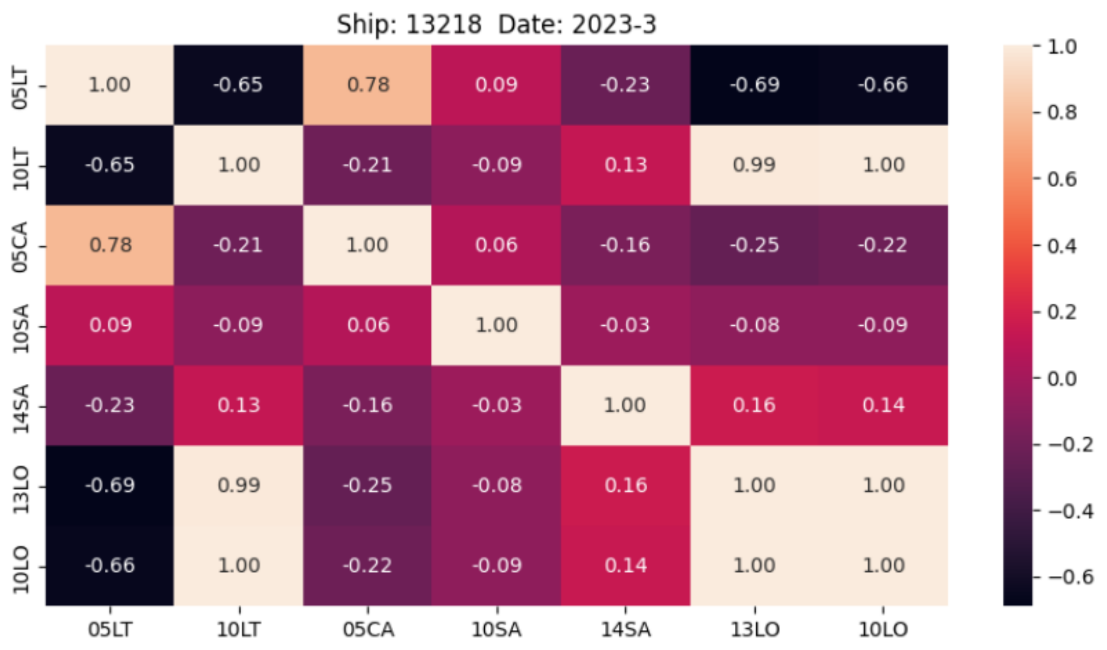


Figure 3.7.1: Correlation heatmap for cylinders

3.7.2.0.2 Low-Temperature Cooling System (LT) The LT system is responsible for cooling parts of the engine by air which again utilizes water. In order to detect anomalous behavior in LT, the idea was to compare the water temperature before, *06LT*, and after, *10LT*, going through the LT system. The attribute describing the engine power percentage, *87XS*, were added alongside *06LT* as inputs to a model which were supposed to predict the temperature of *10LO*. *87XS* is relevant since the activity level of the engine greatly affects the internal temperature of the engine.

The selected attributes alongside some other attributes and the correlation between them can be inspected in Figure 3.7.2 below.



	05LT	10LT	05CA	10SA	14SA	13LO	10LO
05LT	1.00	-0.65	0.78	0.09	-0.23	-0.69	-0.66
10LT	-0.65	1.00	-0.21	-0.09	0.13	0.99	1.00
05CA	0.78	-0.21	1.00	0.06	-0.16	-0.25	-0.22
10SA	0.09	-0.09	0.06	1.00	-0.03	-0.08	-0.09
14SA	-0.23	0.13	-0.16	-0.03	1.00	0.16	0.14
13LO	-0.69	0.99	-0.25	-0.08	0.16	1.00	1.00
10LO	-0.66	1.00	-0.22	-0.09	0.14	1.00	1.00

Figure 3.7.2: Correlation heatmap for low temperature

3.7.2.0.3 High-Temperature Cooling System (HT) In order to assess the condition of the HT system, the most efficient way were to inspect the temperature of the water which were used to cool the system. If there were to be an anomaly in HT, it would be likely that the system is either too warm or too cold. This could be assessed by comparing the cooling water temperature before going through the system, *05HT*, and the temperature of the water as it exits the system, *06HT*.

The idea was to train a model which were able to predict *06HT* by being fed values for *05HT*. In order to ensure that the model could get a sense of understanding of what factors are responsible for the temperature of the HT system, the attribute *87XS*, representing the engine's power percentage, was also added as input.

Figure 3.7.3 below shows the correlation between the selected attributes:

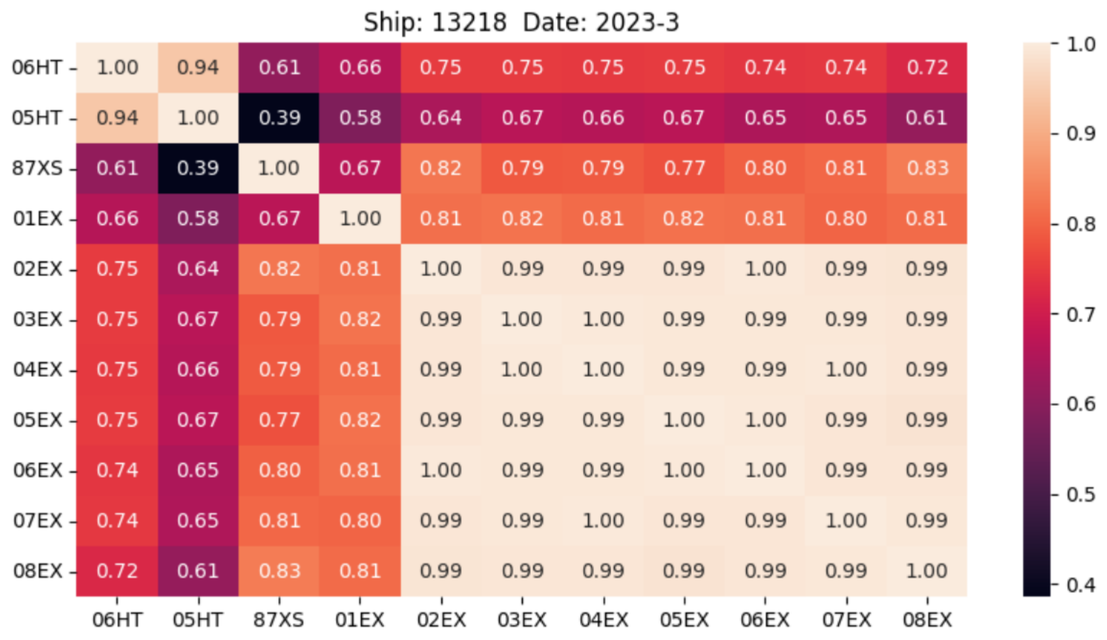


Figure 3.7.3: Correlation heatmap for high temperature cooling system

3.7.2.0.4 Lube Oil System (LO) The most frequent cause for anomalies in the lube oil system is that the filter that the lube oil passes through gets worn out. This is usually recognized by the lube oil pressure getting lower after passing through the filter. In order to detect such an anomaly in the LO system, one could therefore compare the pressure of the lube oil on both sides of the filter, i.e. attribute *13LO* and *10LO* respectively.

To make a model capable of this, the idea was that the model should be able to predict what the pressure of *10LO* should be based on the pressure of *13LO*. The attribute representing the engine's power percentage, *(87XS*, was also added to have a dependency which explains why the fuel oil has the pressure it has at a given time.

The relationship between the mentioned attributes can be assessed by looking at the correlation heatmap depicted below in Figure 3.7.4.

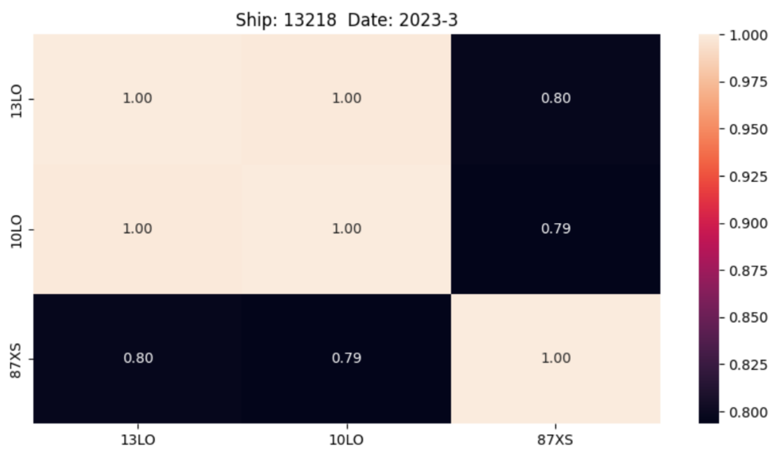


Figure 3.7.4: Correlation heatmap for lube oil system

3.7.2.0.5 Fuel Oil System (DO) The primary component subject to anomalous behavior is the filter which the fuel oil passes through. This filter needs to be replaced as it gets worn out. In order to detect an anomaly in the fuel oil system, indicating that the filter should be replaced, the idea was to predict the attribute *10DO*, which represents the fuel oil pressure after going through the filter. The group chose to use Fuel Oil Pressure Before Filter *13DO* and Engine Power percentage *87XS*. *13DO* was chosen as it highly correlates with *10DO* and since it represents the fuel oil pressure before the filter while *10DO* represents the pressure on the other side of the filter, these two attributes isolate the filter. *87XS* since it describes the activity level of the engine and is therefore highly relevant the the fuel oil pressure.

The relationship between these three attributes can be inspected by viewing the correlation heatmap depicted in Figure 3.7.5.

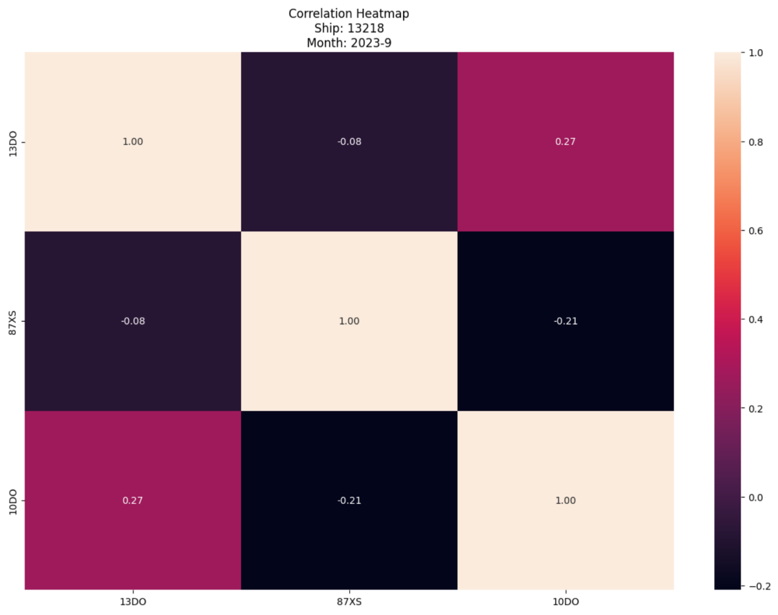


Figure 3.7.5: Correlation heatmap for fuel oil system

3.7.3 Filtering

3.7.3.1 Background

When exploring the dataset and training different models, it turned out that some attributes contained noisy data. As mentioned in Section 2.4.3, noise is considered to be a quality issue that can affect the performance of the models. The group decided to implement filters for models which seemed to be affected the most by noise.

As mentioned in Section 2.4.3, the easiest way to reduce noise in data was to filter it out. The group decided to apply two types of filters in order to filter out noise as well as irrelevant data. Both filters are displayed in Listing 3.7 below.

```
1  # Median filter window size
2  window_size = 355
3
4  if FILTERING:
5    # Filtering by specific condition
6    # See util function definition in Appendix F
7    df = util.remove_rows_at(df, 'df["87XS"] >= 10')
8
9    # Median filter implementation
10   df['87XS'] = df['87XS'].rolling(window_size, min_periods=1).
       median()
11   df['10DO'] = df['10DO'].rolling(window_size, min_periods=1).
       median()
12   df['13DO'] = df['13DO'].rolling(window_size, min_periods=1).
       median()
```

Listing 3.7: Filtering attributes

3.7.3.2 Condition Filtering

The attribute *87XS* represents the engine's power percentage and can be considered to directly describe the activity level of the engine, where *87XS* ≈ 0 means that the engine is not running. The group and the client agreed that a certain threshold should be implemented in order to exclude observations made when the engine was running at minimum power, since those observations would add confusion to the model's logic and since the overall purpose of the anomaly detection project was to detect anomalies which occurred during normal operation.

The filter was implemented by a general function that accepted a string condition and parsed it in order to apply it as a filter. An example of such a condition can be seen in Line 6 in Listing 3.7.3.1. Here, the dataset is filtered to exclude rows where *87XS* has a value below 10. The mean value of *87XS* over the course of a month was approximately 42.

3.7.3.3 Median Filtering

In order to reduce noise in the data, a median filter was applied. The median filter were given a window size which specified the intervals in which a median was calculated. In the case of the DO model which Listing 3.7.3.1 originates from, it was discovered through experimentation that a window size of 355 gave the best results.

The filter were implemented for several models, but not all. Those which received median filtered data were models in which the actual target attributes, the predicted attributes, or input attributes closely related to the target attribute had a noticeable amount of noise. The group decided to be critical when deciding when to implement the median filter in order to preserve the integrity of the dataset.

3.7.4 Scalers

As described in 2.4.5, various scaling techniques were implemented in the model, such as StandardScaler, MinMaxScaler, RobustScaler, and Normalizer to transform the input data. After comparing the results, the group decided to go with the MinMaxScaler as it produced the best outcome. As mentioned in Section 2.4.5.2, this scaler adjusts the feature values to a standard range of [0, 1], which is crucial for RNNs like LSTM, as they are sensitive to the scale of the input data. The scalers were implemented as demonstrated in Listing 3.8.

```python
# Scaling algorithms
SCALERS = {
    'standard':     StandardScaler().fit(dfx),
    'minmax':       MinMaxScaler(feature_range=(0, 1)).fit(dfx),
    'robust':       RobustScaler().fit(dfx),
    'normalizer':   Normalizer().fit(dfx)
}

# Specifying which scaler to use
scaler = SCALERS[ SCALING_ALGORITHM ]       # NB: Scaler must be
    the same as the one used for training
joblib.dump(scaler, f'{SCALER_PATH}{SCALER_NAME}.joblib')
dfx = pd.DataFrame(scaler.transform(dfx))
```

Listing 3.8: Implementation of scalers

3.8 ML Implementation

3.8.1 LSTM Model

When deciding which model to implement, various factors were taken into consideration. The dataset was an ordered collection of sensor measurements in the form of a time series sequence. The data was likely to be affected by both long-term and short-term dependencies, and the main objective was to produce a model which were able to predict a given set of attributes based on another set of attributes. Based on this, it decided that the most suitable option was the LSTM model.

The main idea was to write a script (Listing 3.9) which could act as a protocol for producing models predicting different sets of attributes. This made it possible to recycle the script for each model. In an optimal solution, one could specify only a couple of variables in order to train a model. For each attribute group, one could train a model with a set of input attributes and a set of target attributes.

```python
import numpy as np
import pandas as pd

from keras.models import Sequential
from keras.callbacks import EarlyStopping
from sklearn.model_selection import train_test_split
from keras.layers import LSTM, Dense, Dropout, Bidirectional

# Splitting the dataset into training- and validation sets
X_train, X_test, y_train, y_test = train_test_split(dfx, dfy,
    test_size=TEST_SIZE, shuffle=False)

# Reshaping training set
X_train = X_train.to_numpy().reshape(X_train.shape[0], timesteps,
    num_attributes_x)
y_train = y_train.to_numpy().reshape(y_train.shape[0], timesteps,
    num_attributes_y)

# Reshaping validation set
X_test = X_test.to_numpy().reshape(X_test.shape[0], timesteps,
    num_attributes_x)
y_test = y_test.to_numpy().reshape(y_test.shape[0], timesteps,
    num_attributes_y)

# Define LSTM model
model = Sequential()
model.add(Bidirectional(LSTM(LAYER_1_UNITS, input_shape=(timesteps
    , num_attributes_x), return_sequences=True)))
model.add(Dropout(DROPOUT))
model.add(Dense(num_attributes_y))

# Compile the model
model.compile(optimizer=OPTIMIZER, loss=LOSS_FUNCTION)
```

Listing 3.9: Defining the LSTM model

3.8.2 Early Stopping Mechanisms

The model's were trained with two early stopping mechanisms which would activate if either the loss function or the validation loss would start to increase during training. One could specify a patience level for the mechanisms, which determines how many epochs the mechanism will wait before intervening.

The patience level for both mechanisms were set to five epochs. If a model experienced increasing loss or validation loss for 5 epochs, the early stopping mechanisms would stop the training early, restoring the weights at the epoch before the loss or validation loss started to increase. This was implemented as a means of preventing overfitting, which the loss functions are one of the most reliable indicators for this as stated in Section 2.5.7.

3.8.3 Experimentation

3.8.3.1 Layers & Neurons

In order to increase the model's capability of learning long-term dependencies, the LSTM layers were wrapped inside a bidirectional object. As mentioned in Section 2.5.5.1, bidirectional LSTMs has the ability to process sequences with data in both forward and backward directions, allowing the model to better learn long-term dependencies.

It was also attempted to train the models with several bidirectional LSTM layers. It was discovered that two layers gave the best results. Models with more than two layers were often too complex, which resulted in underfitting and vanishing gradient. A single-layer model were quite accurate when predicting attributes at dates close to the date the model was trained on, but got slightly less accurate when the distance increased. The double-layered model was not as sensitive to this.

The amount of neurons in each layer was also subject to experimentation. The group discovered, depending on how many layers the model had, that increasing the number of neurons to a certain level somewhat tended to improve the trained model's accuracy, but if there were to many neurons combined with several layers, the vanishing gradient problem (see Section 2.5.4) occurred.

3.8.3.2 Activation Functions

The LSTM layers all implemented an activation function. The one which were used were *tanh* which was the default activation function, and also the function which in general were used the most in LSTM projects.

3.8.3.3 Optimization

The group tried compiling the model using different optimizer algorithms. Choosing the correct optimizer algorithm were a subject to experimentation. The ones which were tried were Adam, Adamax, SGD, and RMSprop. The algorithm which gave the best results was RMSprop. It was discovered during experimentation that some of the optimizer algorithms made the model vulnerable to the vanishing gradient problem (see Section 2.5.4).

3.8.3.4 Dropout

The dropout layer was a means of preventing overfitting. The higher the dropout-rate, the higher was the amount of the training results discarded for each epoch. It is normal to set the dropout-rate at around 20%. After testing out different dropout-rates, it was concluded that 20-25% in general gave the best results for the models and were sufficient in terms of overfitting prevention.

3.8.4 Training

The dataset in its entirety were divided into sub-datasets for each day. In order to make the models learn the underlying nature of the data, the group decided to train it on continuous data ranging from an entire month. This would also make use of the LSTM's ability to learn dependencies over time. In order to train it on a month of data, all day-datasets in a month were merged into a single dataset. This was done using a utility script (see Appendix F).

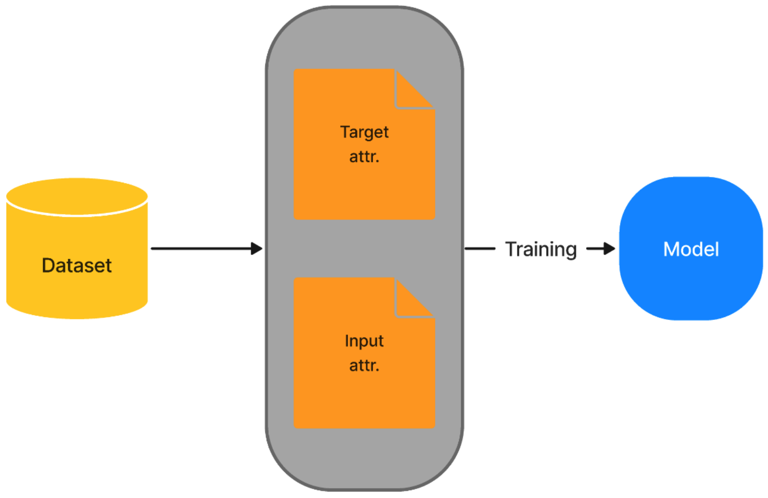


Figure 3.8.1: Illustration of model training

Figure 3.8.1 above illustrates how the dataset is divided into input and target attributes, and that both are put to use when training the model. When the model is trained, it will only receive input attributes, which it will use to predict the corresponding target attributes.

In order to train the models, the dataset was divided into a training set and a validation set. The training proportion was 80% - 20%, which equals approximately 24 and 6 days. The models received both the input- and the target attributes of the training set during training. Upon validation, the models were fed only the input attributes of the validation set. The predictions produced by the models were then compared to the corresponding target attributes. Figure 3.8.1 depicts an illustration of the training phase.

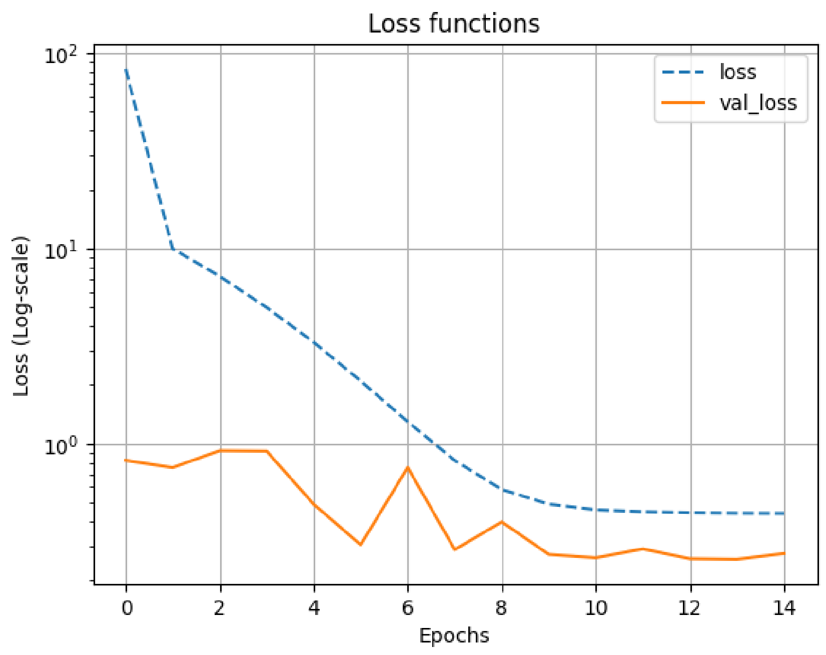


Figure 3.8.2: Functions for loss and validation loss

MSE was used as the loss function. In Figure 3.8.2 one can see that the model is learning due to the loss function were decreasing. Both functions seemed to stagnate at around 10 epochs, indicating that the model had learned as much as it could from the dataset. The distance between the two functions were also decreasing, indicating that the model was not underfitting. All in all, loss-functions such as these indicated that the model was learning properly.

3.8.5 Validation

3.8.5.1 Plots

A way to assess the accuracy of a trained model was to plot the predicted attributes next to its actual values. An accurate model could be recognized if it produced a prediction plot which was similar to the plot of the actual values. Furthermore, the loss functions after training were plotted, which could give insights regarding the training's effect on the model.

Prediction plotting were conducted in two ways. The first was to plot the two graphs next to each other, making it easy to compare the pattern of movement. The other was to plot the two graphs on top of each other. It was considered that the latter often were the best method, since this allowed to accurately compare the graphs by seeing if they overlapped each other.

3.8.5.2 Accuracy Scores

In order to assess the accuracy of the models in a more quantitative way, the group used the predicted attribute values and the actual values to calculate the mean squared error and r^2. An accurate model could be recognized by having a low mean squared error and an r^2 close to 1.

3.9 Detecting Anomalies

In order for the models to be usable for anomaly detection, they needed to be accurate when predicting attributes without anomalous behavior, and be inaccurate when predicting attributes with anomalies. This made testing the models on anomalous data an important part of the model validation.

KM told the group in which datasets there were anomalies, but it was usually possible to locate datasets with anomalies by looking at the correlation between the attributes. An anomaly is indicated if the correlation differs from what it normally is. An example of this can be seen below in Figure 3.9.1:

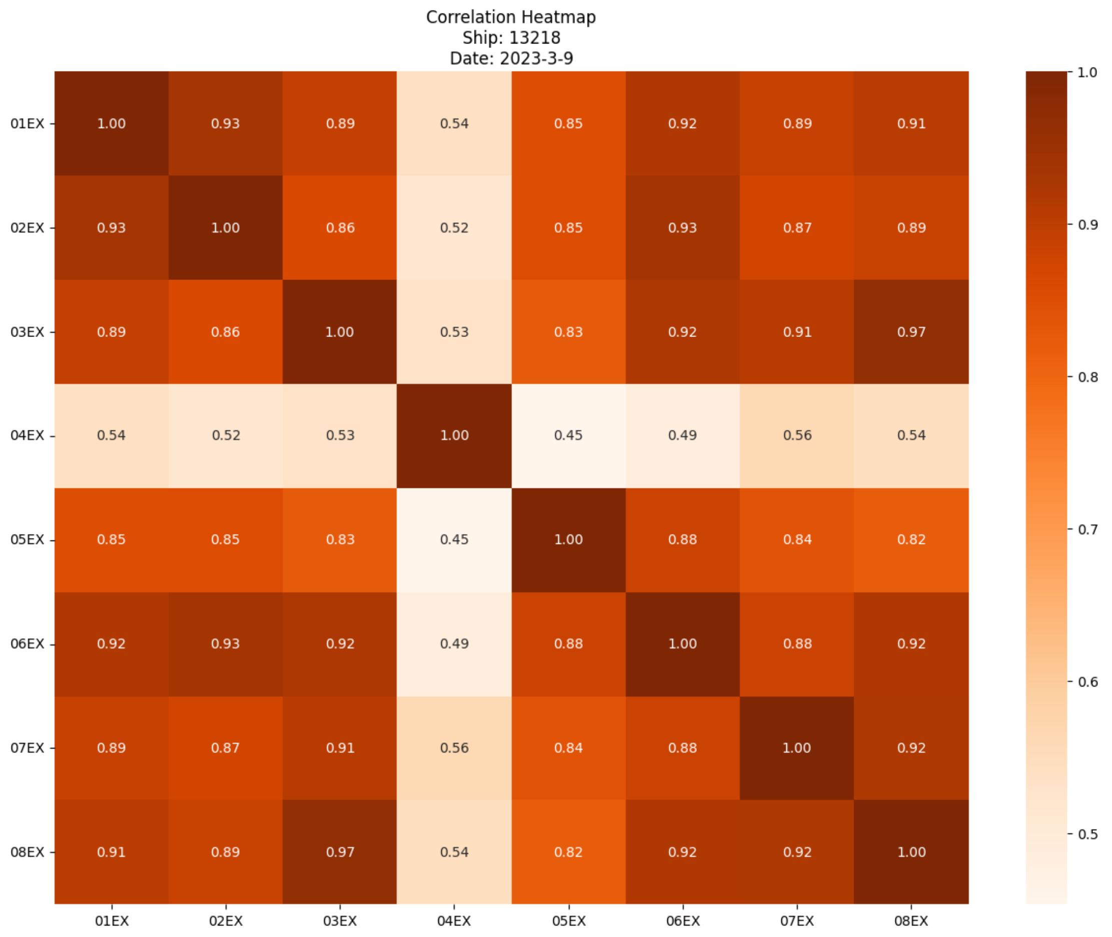


Figure 3.9.1: Correlation heatmap for EX with anomaly in cylinder no. 4

In the above figure, one can see that *04EX* correlates less with the other EX-attributes. In a perfect situation, *01EX*, *02EX*, ..., *08EX* would all have a correlation-coefficient $R = 1.0$ with each other.

The model was fed with input attributes from the anomalous datasets. The model then predicted the target attributes for the corresponding inputs. The results were validated by plotting and calculating accuracy scores.

When testing the models on datasets with anomalies, it was expected for it to be far less accurate than it usually is. The idea was that anomalies could be detected by looking at the models accuracy. This forms the basis for the chosen approach for anomaly detection as mentioned in Section 3.3. In order for the approach to be considered to be an appropriate solution, there will need to be a significant difference between a model's accuracy upon predicting non-anomalous data and the accuracy upon predicting anomalous data.

CHAPTER

FOUR

RESULTS

This chapter presents the results achieved throughout the project in the form of validation of the trained ML models in different perspectives including learning abilities, model accuracy, plots of predictions as well as a demonstration of model predictions upon anomalous data.

4.1 Models

4.1.1 About the Validation

4.1.1.1 Validation Sets

The trained models were validated by having them predict the last 20% of the dataset they were trained on. The dataset in its entirety contained measurements from 30 continuous days, with measurements performed each second. This leaves the validation set to correspond to approximately six days. The total number of observations in the validation set was approximately 410 000.

4.1.1.2 Loss Functions

The functions of loss and validation loss is plotted and used to determine how the models responded to the training it went through. The two functions are plotted as line graphs on a logarithmic scale.

The evaluation of the loss functions is convenient in order to gather insight in whether the models managed to generalize and learn from the data it received as input while training (see Section 2.5.6 and Section 2.5.7). This can be deduced by seeing if the loss function decreases throughout training. It can also indicate whether the data is overfitting (Section 2.5.10.3), underfitting (Section 2.5.10.4) or fitting just right with the data by looking at the validation loss, which should be as flat and stable as possible. Furthermore, the distance between the two functions and the end of the training can also indicate whether the model will be able to perform accurate predictions on new data.

During training of the models, there were early stopping mechanisms at place which would activate if either loss or validation loss started to increase as mentioned in Section 3.8.1. The mechanisms had a patience of 5 epochs, meaning that it if it would stop the training, the model would restore the weights from five epochs before it stopped.

4.1.1.3 Comparison of Means

The comparison of means is conducted by calculating the mean of all values of the predicted attributes and its corresponding actual values respectively. The mean values were then compared to each other by calculating the absolute value of the difference between them.

Small differences between the mean values indicates that the model is able to predict the overall temperature quite accurately. This measure alone is not sufficient grounds to deem the model as precise or not, as it only gives a mere overview of the model's performance. If the model is suffering from the vanishing gradient problem as mentioned in Section 2.5.4, the model would likely produce a mean value similar to the mean of the actual data, but it could in the worst case mean that the model only returns a single value regardless of the input, rendering it useless due to its lack of predicting the fluctuations in the data. In order to validate the model's ability to precisely predict the actual fluctuations of the temperature, one cannot rely on this statistic alone.

4.1.1.4 Accuracy Scores

The accuracy scores used to evaluate the models are Mean Squared Error and r^2. These metrics are calculated by comparing all actual values with their corresponding predicted values of the validation set. As mentioned in Section 2.5.11.2, the MSE is a good measure which describes how much the predicted values deviates from the actual values. A low MSE indicates better accuracy. As mentioned in Section 2.5.11.3, r^2 is a statistical measure of how well the model fits data points. This measure has in other words the ability to describe how well the models are able to predict the fluctuations in data accurately. A r^2 close to 1 indicates better accuracy.

4.1.1.5 Visualization

The models were also validated visually by plotting the predictions alongside or on top of a plot of the actual data in the same time span. This validation method does not provide any quantitative results, but it is a valuable test which allows the developers to quickly assess how accurate the model was at its predictions by comparing the shape and heights of the plotted graphs.

The visual assessment was conducted by plotting the graphs in two ways. The first way was plotting the two graphs directly on top of each other, while the other way was plotting the graphs next to each other in their own sub-plots.

4.1.2 Cylinder Exhaust Temperature

4.1.2.1 Loss Function Analysis

Below in Figure 4.1.1 is a plot of the loss and validation loss of the model during training. The model was set to train for 15 epochs, but it seems that the early stopping mechanism intervened, making the model restore the weights which gave the best results, which in this case would be epoch 7.

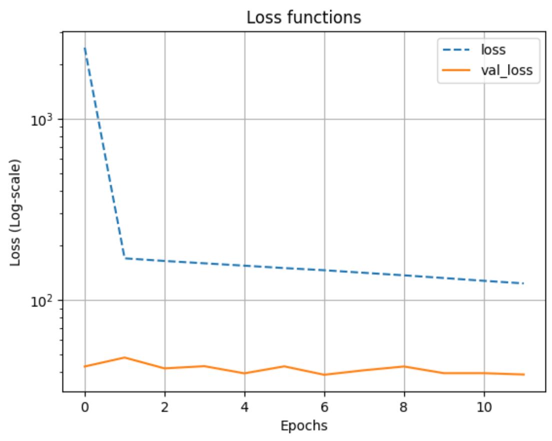


Figure 4.1.1: Loss function for EX

The loss function seems to decrease quite rapidly during the first epoch, and later only slightly decreasing in a more linear motion throughout the remaining epochs. This indicates that the model learned quite much during the first epoch, and then continued to learn a bit more for each epoch after.

The validation loss seems to fluctuate slightly throughout the epochs, but it seems to be remain quite stable and neither decrease nor increase much in the long run. This is a sign that the model has a good fit and is neither under- or overfitting.

All in all, it seems that the model is learning appropriately and generalizes the data quite well.

4.1.2.2 Predictive Performance

As shown in Table 4.1.1, the overall difference between the mean of the predicted and the actual values for the cylinders ranges from 0.143 to 4.948. The exhaust gas temperature for the cylinders are usually fluctuating between 250°C and 400°C.

Attribute	**Actual Mean**	**Prediction Mean**	**Difference**
01EX	350.011	347.838	2.173
02EX	354.548	357.865	3.317
03EX	348.843	343.895	4.948
04EX	332.110	333.444	1.334
05EX	339.986	339.843	0.143
06EX	351.335	350.522	0.813
07EX	341.662	340.669	0.993
08EX	335.679	335.248	0.431

Table 4.1.1: Comparison of mean value of predictions and actual values for EX attributes

The predictions for the first four cylinders are indicated to be not as accurate as the remaining cylinders according to Table 4.1.1. This could be due to minor anomalies in the data or errors made during preprocessing. As mentioned in Section 2.4.4, the quality of the processed data will directly affect the accuracy of the model.

It should, however, be mentioned that a difference of means of 4.948, as in *03EX*, is still considered to be acceptable. Overall, the mean difference is quite low considering the number of predictions made, the range of values, and the volatility of the exhaust gas temperatures, as all these factors increases the sensitivity of this metric.

4.1.2.3 Accuracy Metrics

Below, in Table 4.1.2, the MSE and r^2 score of the model for each cylinder is displayed. The MSE ranges between 25.059 and 63.026. Note that the MSE for the first four cylinders are notably higher than four the remaining cylinders. This observation corresponds with the observations made when evaluating the comparison of means in the above section. The r^2 values does not display this phenomenon although attribute *03EX* has the lowest score, which was also the case in the comparison of means.

Metric	**01EX**	**02EX**	**03EX**	**04EX**	**05EX**	**06EX**	**07EX**	**08EX**
MSE	49.396	50.129	63.026	30.313	25.059	28.509	30.813	32.314
r^2	0.935	0.922	0.828	0.923	0.932	0.941	0.939	0.952

Table 4.1.2: Accuracy metrics for EX

The MSE scores are quite reasonable. Since the actual data fluctuates quite much between the temperatures 250°C and 400°C, it was expected that the MSE scores would be somewhat high, but the observed scores is considered to be satisfactory concerning the fluctuation. The achieved r^2 scores were actually better than expected. As mentioned in Section 2.5.11.3, if r^2 equals 1, it indicates that the independent variables explains all the variance in those that are dependent, so having almost all scores above 0.92 indicates high accuracy of the model's predictions.

4.1.2.4 Visualization

Below, Figure 4.1.2 depicts a plot of the predicted and actual values for *01EX*. The prediction graph seems to follow the fluctuations of the actual values quite well while also hitting the correct temperatures most of the time. The predictions does seem to be a bit lower than the actual values between $t \approx 30000$ and $t \approx 180000$.

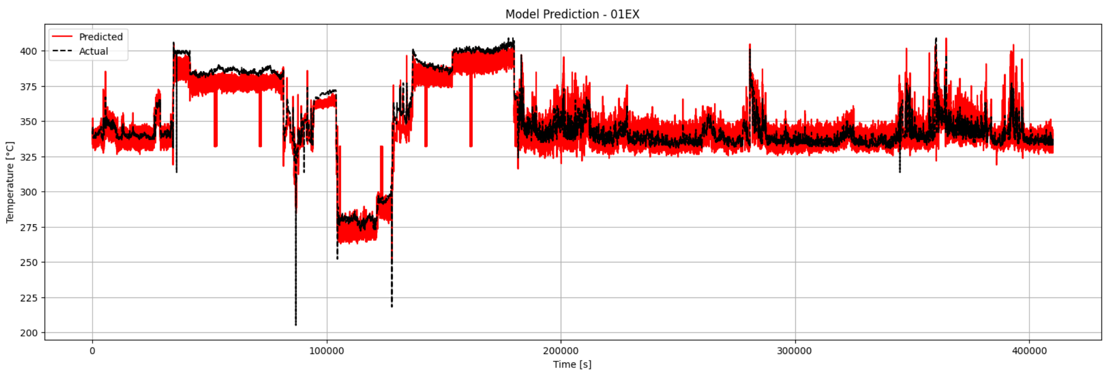


Figure 4.1.2: Predictions for 01EX

Figure 4.1.3 depicts a plot of the predicted and actual values for *05EX*. The prediction graph seems to fit better than what it did for *01EX* by hitting the correct temperatures with almost no exceptions.

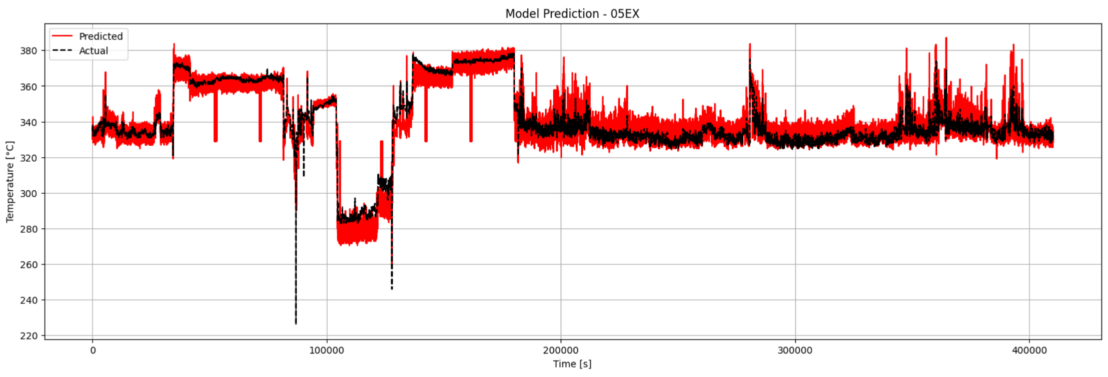


Figure 4.1.3: Predictions for 05EX

Overall, both plots shows that the model has learned to predict the exhaust gas temperatures with quite high accuracy. The model is able to predict the fluctuations quite precisely and also predict the correct temperatures. One may notice that the model does, however, produce some noisy predictions which is visible in both Figure 4.1.2 and Figure 4.1.3 between $t \approx 180000$ and $t \approx 400000$, but one could argue that the actual values mostly lies in the center of the noisy areas, which could indicate that the model only amplifies the noise already present in the actual data.

In the case of the EX system, the model was trained on data which were not filtered. It is possible that the model would produce less noise if a mean filter was applied to this model as described in Section 3.7.3. It was concluded that the model already met the required accuracy levels, and the implementation of a median filter was therefore not prioritized.

4.1.3 High-Temperature Cooling System

4.1.3.1 Loss Function Analysis

The loss functions for the HT model is depicted in Figure 4.1.4. The model was set to train for 15 epochs, but seems to have stopped early at the 10^{th} epoch and restored the best weights which were achieved at the 5^{th} epoch.

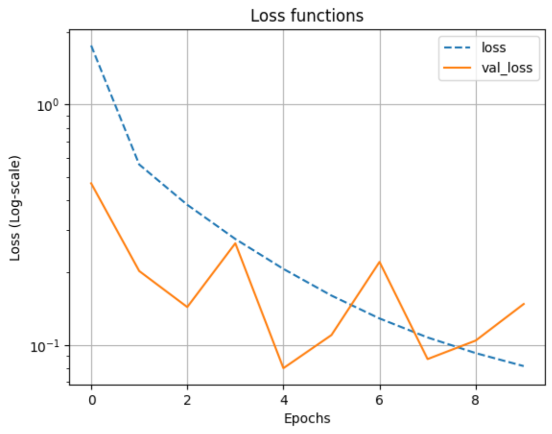


Figure 4.1.4: Loss function for HT

The loss function seems to exponentially decrease throughout training, which indicates that the model learns to fit the data it trains on.

The validation loss seems to have a decreasing trend, but it fluctuates significantly up and down for each epoch. Typically, it would be an indication of overfitting if the validation loss starts to be greater than the loss function, but in this case it was prevented by the early stopping mechanism which restored the weights of epoch 5. Seeing the validation loss between epoch 0 and 5 as a whole, it does not appear to be stable, but it should be noted that the fluctuation is of the log-scale of 10^{-10} which is quite small.

All in all, the model appears to learn from the data, but it shows sign of having the potential to fit better.

4.1.3.2 Predictive Performance

As shown in Table 4.1.3, the model managed to produce a set of predictions for *06HT* whose mean value was quite similar to the mean value of the actual values.

The difference lies at 0.287, which is quite low.

Attribute	Actual Mean	Prediction Mean	Difference
06HT	78.660	78.373	0.287

Table 4.1.3: Comparison of mean value of predictions and actual values for HT

4.1.3.3 Accuracy Metrics

Below, in Table 4.1.4, the accuracy scores are displayed. The MSE is very low which indicates that the model is very precise at predicting the correct magnitude of the temperature in *06HT*. The r^2 score is close to 1, which indicates that the model also is able to predict the fluctuations in the data quite well.

Metric	06HT
MSE	0.080
r^2	0.910

Table 4.1.4: Accuracy metrics for HT

4.1.3.4 Visualization

When comparing the prediction graph for *06HT* next to the graph representing the actual values, (Figure 4.1.5), it is clear to see that the model had a positive outcome of the training. The graphs seems to be very similar in terms of shape, and they also seem to be almost identical in terms of magnitude.

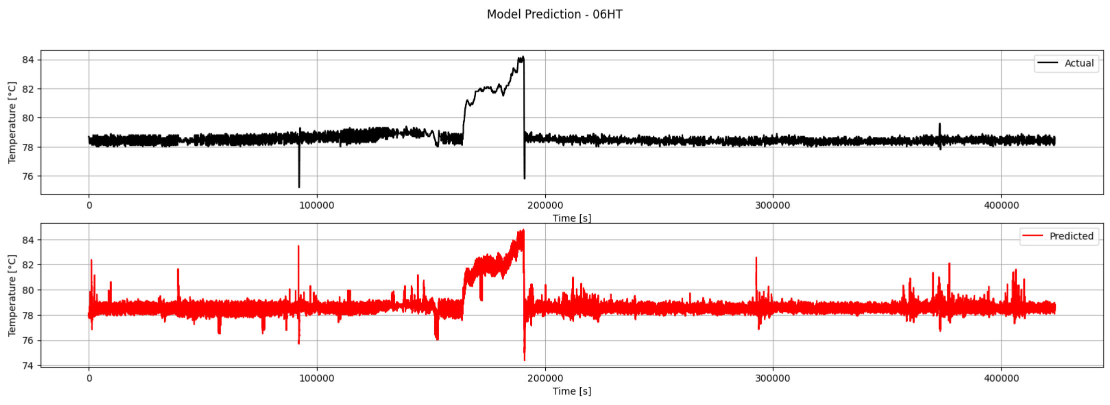


Figure 4.1.5: Predictions for HT

The model was trained on data which implemented a median filter for the input attribute *87XS*. This resulted in an improvement in the overall accuracy, and there were also less noise in the prediction graph. The model seems to be very accurate in its predictions, but it should be noted that it is somewhat naive to determine its accuracy based on these validations alone since the data it predicts is linear most of the time.

Due to the attribute's linear nature, one could argue that the model in fact meets the requirements in terms of accuracy, taking into consideration that the predictions were also accurate at the sudden fluctuation between $t \approx 150000$ and $t \approx 190000$.

4.1.4 Low-Temperature Cooling System

4.1.4.1 Loss Function Analysis

Figure 4.1.6 below depicts the loss functions for the LT model. This model were set to train for 10 epochs, but the early stopping mechanism was activated just before epoch 7, restoring the best weights which were at the 2^{nd} epoch.

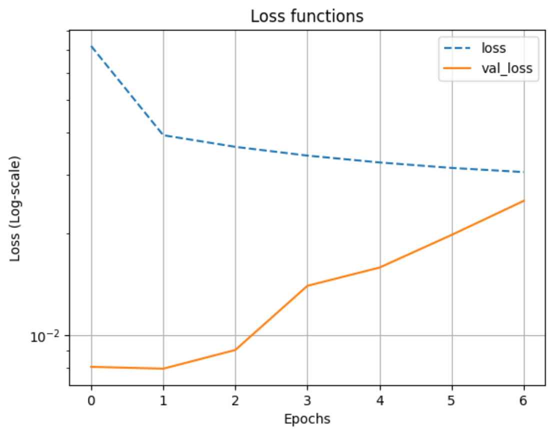


Figure 4.1.6: Loss function for LT

The loss function seems to have a steep decrease in the 1^{st} epoch, and later decreasing in a less steep, but linear fashion. It indicates that the model learned the most during the first epoch, and later learned decreasingly less for each epoch.

The validation loss is actually increasing throughout the training period. This could indicate that the model was about to be overfitted. The log-scale is at 10^{-2} so it not by a lot, but the early stopping mechanism restored the weights at epoch 2, which is likely to have prevented the model to be overfitted.

Overall, the model seemed to learn throughout the training, but it was possible that it were about to be overfitted. The early stopping mechanism made the model restore the weights from the 2^{nd} epoch, which left the model with a rather short amount of training time.

4.1.4.2 Predictive Performance

As shown below in Table 4.1.5, the mean value of the model's predictions of *10LT* is very close to the mean value of the actual values. There is 0.021 in absolute difference between them, which is quite low. One should note, however, that the attribute has a quite low mean value which makes it natural that the difference of means is low as well.

Attribute	Actual Mean	Prediction Mean	Difference
10LT	4.405	4.384	0.021

Table 4.1.5: Comparison of mean value of predictions and actual values for LT

4.1.4.3 Accuracy Metrics

Table 4.1.6 displays the scores for MSE and r^2. Both metrics indicates a remarkably precise model, attaining an exceptionally low MSE of 0.008 and a just as exceptionally high r^2 of 0.997. This affirms that the predictions are in close agreement with the actual values of *10LT*.

Metric	10LT
MSE	0.008
r^2	0.997

Table 4.1.6: Accuracy metrics for LT

4.1.4.4 Visualization

Below, in Figure 4.1.7, one can see the prediction graph next to the graph of the actual values. It comes quite clear that the model was able to accurately replicate the actual behaviour of the cooling system.

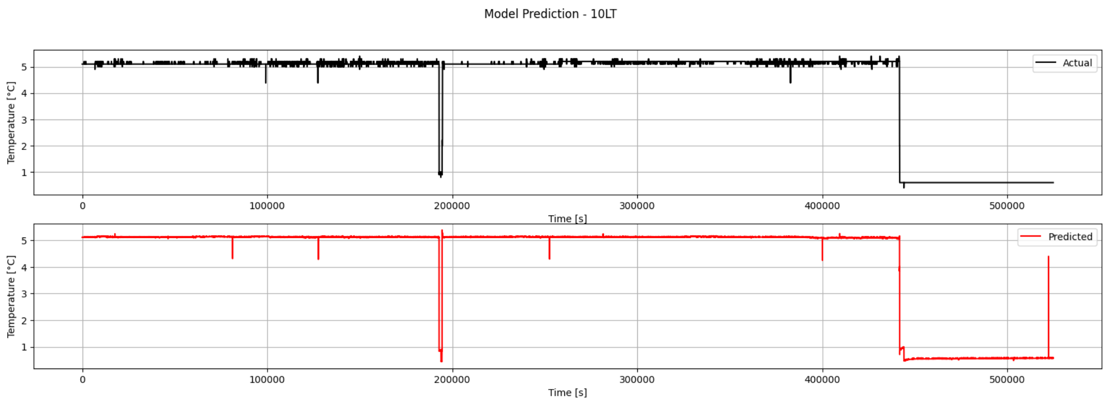


Figure 4.1.7: Predictions for LT

Interestingly, there appears to be less noise in the predicted values compared to the actual values. This could be an indication that the model is generalizing well, but it could also be unrelated. Another likely explanation is that the noise is caused by the actual sensor rather than the attribute itself. The model was trained on data which implemented a median filter for the input attribute *87XS*, which also could be a factor to this phenomenon.

The graphs are almost identical except some unexpected spikes visible in the prediction graph. The model was also able to correctly predict the steep downwards spike at $t \approx 195000$, which is reassuring since it can be difficult to evaluate the model's accuracy when the data it predicts upon is usually almost completely linear. Even when taking the successful prediction of the downwards spike, it is still somewhat ambiguous to say anything about the realistic accuracy of the model, but the developers think it is justifiable to assume that the accuracy meets the requirements given that the prediction graph continues to be precise when the temperature in *10LT* drops such as in $t \approx 195000$ and $t \approx 435000$.

4.1.5 Lube Oil System

4.1.5.1 Loss Function Analysis

The loss functions for the trained model is depicted below in Figure 4.1.8. The model was set to train for 10 epochs, but the early stopping mechanism seems to have stopped the training after the 5^{th} epoch, restoring the weights from the 2^{nd} epoch instead.

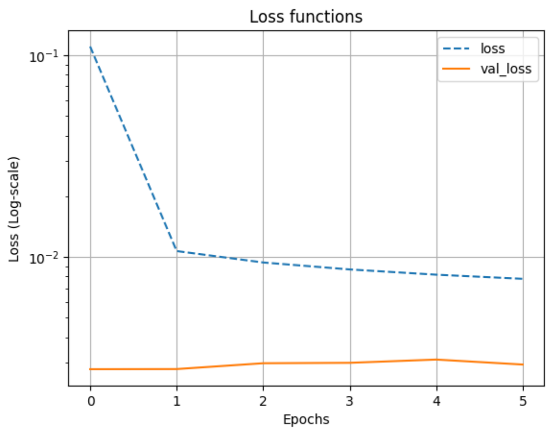


Figure 4.1.8: Loss function for LO

The loss function strongly decreased during the first epoch, and later had a weaker decreasing trend throughout the training. This indicates that the model learned the most in its initial phase, and later learned less and less for each epoch.

The validation loss seems to be almost linear throughout the training, which is a good sign indicating that the model is fitting just right. There is, however, a slight increase in the validation loss, which promptly activated the early stopping mechanism.

Judging by the loss functions, it seems that the model was learning quite well from the data and without overfitting or underfitting. The total training time considering that the early stopping mechanism intervened was rather short at only two epochs.

4.1.5.2 Predictive Performance

Similar to the mean value comparison of *10LT*, this model produces a set of predictions for *10LO* whose mean value is very close to the mean value of the set of the actual values for the same attribute, which is shown in Table 4.1.7.

Attribute	Actual Mean	Prediction Mean	Difference
10LO	4.671	4.688	0.017

Table 4.1.7: Comparison of mean value of predictions and actual values for LO

The difference of the mean values lies at 0.017. The mean of the actual values are quite low at 4.671, making it reasonable to assume that a properly trained model would produce a mean which has a small difference as well.

4.1.5.3 Accuracy Metrics

The accuracy scores displayed in Table 4.1.8 shows that the model achieved an impressively low MSE of 0.003. Furthermore, the attained r^2 equals 0.972. Both scores are very good and further indicates that the model was precise in its predictions for the validation set.

Metric	10LO
MSE	0.003
r^2	0.972

Table 4.1.8: Accuracy metrics for LO

4.1.5.4 Visualization

The prediction graph and the actual graph was plotted side by side as shown in Figure 4.1.9. The graphs appears to be almost identical to each other, with the only visible difference being noise.

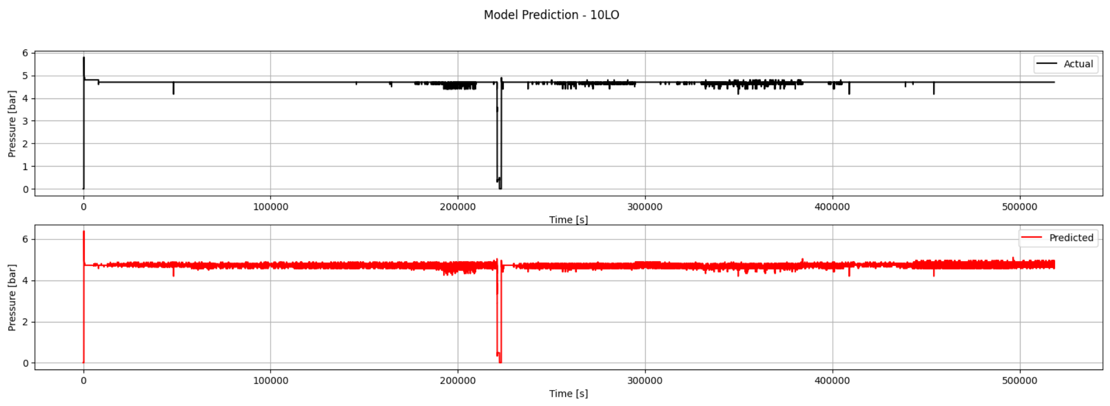


Figure 4.1.9: Predictions for LO

The graph is quite linear, which makes it easy for the model to predict the pressure in *10LO* accurately, but it has proven itself to also be able to predict spikes and fluctuations as seen in $t \approx 0$, $t \approx 50000$, $t \approx 225000$, and $t \approx 45000$. This is a good sign which indicates that the model has learned how the dependencies affect the target attribute.

4.1.6 Fuel Oil System

4.1.6.1 Mean Filtering

While training models for the fuel oil system, it was discovered that there were a lot of noise in the input attribute *13DO* and the target attribute *10DO*. The group decided to train two models for the system, one where a median filter is implemented for the two attributes, and one without. This was done in order to see the effects of the median filter and to emphasis how this would affect the model's ability to accurately predict *10DO*.

For the sake of simplicity, the model which were trained on data with the median-filter will be called the "filtered model" while the model which were trained on unfiltered data will be called the "non-filtered model".

4.1.6.2 Loss Function Analysis

Figure 4.1.10 below depicts the loss functions for the model which were trained on data without the median filter. The model was set to train for 15 epochs, which it seems to have completed (epoch count starts from 0 in the plot).

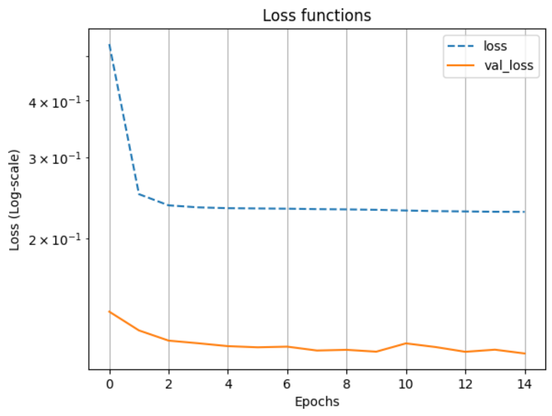


Figure 4.1.10: Loss Function for DO without filter

The loss functions for the filtered model is depicted below in Figure 4.1.11. This model was also set to train for 15 epochs, and like the non-filtered model, the early stopping mechanism did not activate, which resulted in a total training time of 15 epochs.

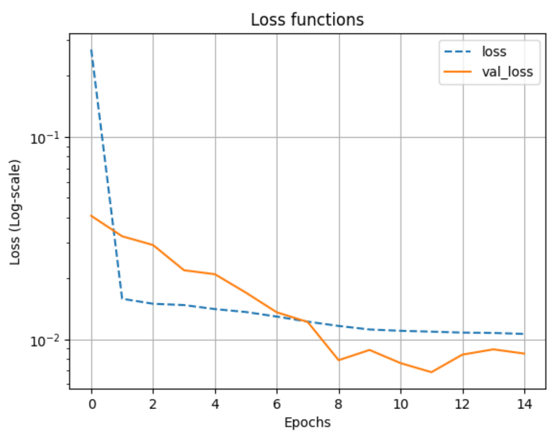


Figure 4.1.11: Loss Function for DO with filter

Due to the plots having different logarithmic scale, it can be somewhat difficult to directly compare the graphs from the two models with each other.

The loss function for the non-filtered model seems to only decline in the earliest epochs. For the rest of the epochs, it seems that there were little to no decline in the loss. This could indicate that the model simply stopped learning from the data after the completing the first three epochs.

On the other side, validation loss for the non-filtered model seems to be quite stable. It had a slight increase in the first epochs but seemed to be quite linear later. It is likely to be so due to the model does not seem to learn much from the later epochs, which would make it reasonable to believe that the validation loss should not change much either.

For the filtered model, the loss function is strongly declining after the first epoch. After, it has a much less steep decline. It indicates that the model learned the most in the earliest epochs, but unlike the non-filtered model, it seems to have learned a bit more and also continued to learn ever so slightly for each epoch throughout training.

The validation loss for the unfiltered model was decreasing throughout the training. While this is often a warning sign that the model tends to underfit, it is to a quite low scale. Furthermore, the validation loss seems to stabilize a bit at the later epochs, which could indicate that the model is not underfitting anymore.

Overall, both models seem to have learned from the data, but mostly in the beginning. It is reasonably to believe that both models had a good fit with the data as well, judging by the validation loss functions. The filtered model seems to outperform the non-filtered model, judging by the fact that it learned more which can be indicated by looking at the logarithmic scale of the two plots, and that the loss and the validation loss functions were quite close to each other in the end, which indicates that the filtered model had a better effect of the training compared to the non-filtered model.

4.1.6.3 Predictive Performance

The filtered and the non-filtered model produced two different sets of predictions. The mean of the predictions alongside the mean of the actual values are displayed below in Table 4.1.9 for the non-filtered model, and in Table 4.1.10 for the filtered model.

Attribute	**Actual Mean**	**Prediction Mean**	**Difference**
10DO	6.931	7.014	0.083

Table 4.1.9: Comparison of mean value of predictions and actual values for DO without filter

Attribute	**Actual Mean**	**Prediction Mean**	**Difference**
10DO	6.937	6.991	0.054

Table 4.1.10: Comparison of mean value of predictions and actual values for DO with filter

The non-filtered model (Table 4.1.9) produced a set of predictions whose mean value were 7.014. The actual mean of the values were 6.931. This left a quite small difference of 0.083. This is a good sign that even without the median filter, the model is still capable of predict the target attribute with somewhat accuracy.

The filtered model (Table 4.1.10) on the other hand produced predictions whose mean value were 6.991. The actual mean were 6.937, resulting in a difference of 0.054. This is slightly lower than same comparison made for the non-filtered model, which indicates that the median filter does improve the accuracy. In order to be certain it will not suffice by only using this statistic to evaluate the accuracy of the two models.

Note that the actual mean is not identical for the two models even though they were trained and tested on equal proportions of the data at the same indexes in the dataset. The reason why these values are not equal is likely due to the median filter applied on the data for the filtered model, which also filters the target attribute *10DO*.

4.1.6.4 Accuracy Metrics

The non-filtered model achieved, as shown in Table 4.1.11 an accuracy score of 0.112 for MSE and 0.115 for r^2. The MSE score is quite low, which indicates that the predictions are overall quite similar to the corresponding actual values in terms of magnitude. The r^2 score, however, is a lot lower than expected. This indicates that the model still is inaccurate in its predictions of the fluctuations in the data.

Metric	10DO
MSE	0.112
r^2	0.115

Table 4.1.11: Accuracy metrics for DO without filter

Below, in Table 4.1.12, is the accuracy scores of the filtered model. The MSE score is remarkably low at 0.008, which is a large improvement compared to the MSE of the non-filtered model. The r^2 score is at 0.587 which is also a significant improvement of the corresponding score of the non-filtered model. The r^2 score is still a bit lower than expected, but it makes up for it to some extent by the low MSE score.

Metric	10DO
MSE	0.008
r^2	0.587

Table 4.1.12: Accuracy metrics for DO with filter

Both models achieves a very good MSE score. On the other hand, they both achieves an r^2 score which is significantly lower than any of the other models made in this project. The mean filter seems to improve the low r^2 score of the non-filtered model by increasing it by almost 500%, as well as also decreasing the MSE score, which further indicates that the mean filter enhances the model's accuracy.

4.1.6.5 Visualization

Below, in Figure 4.1.12, the graph of the predicted values for *10DO* is plotted on top of the corresponding graph of actual values. The noise which naturally lies in the data is easy to identify, but interestingly, the model seems to be able to be somewhat unaffected to the substantial amount of noise, predicting values with much less noise. Furthermore, the prediction graph seems to follow the very center of the actual graph. Assuming that the noise in the actual data deviates equally to both sides, it is likely that the prediction graph actually lines up with the actual data very accurately.

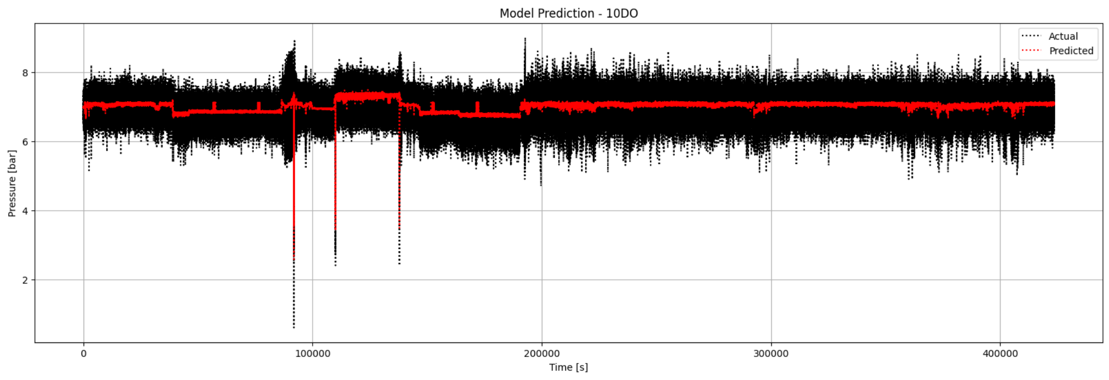


Figure 4.1.12: Predictions for DO without filter

The prediction graph and the actual graph for *10DO* is plotted below each other in Figure 4.1.13. The graphs were plotted below each other in order to make it easier to compare the shape of the graphs. The median filter has seemingly reduced the noise in the data to a minimum, making it easier to see the overall fluctuation of the attribute. The prediction graph has a bit more noise, but it has much resemblance to the actual graph in terms of shape and magnitude.

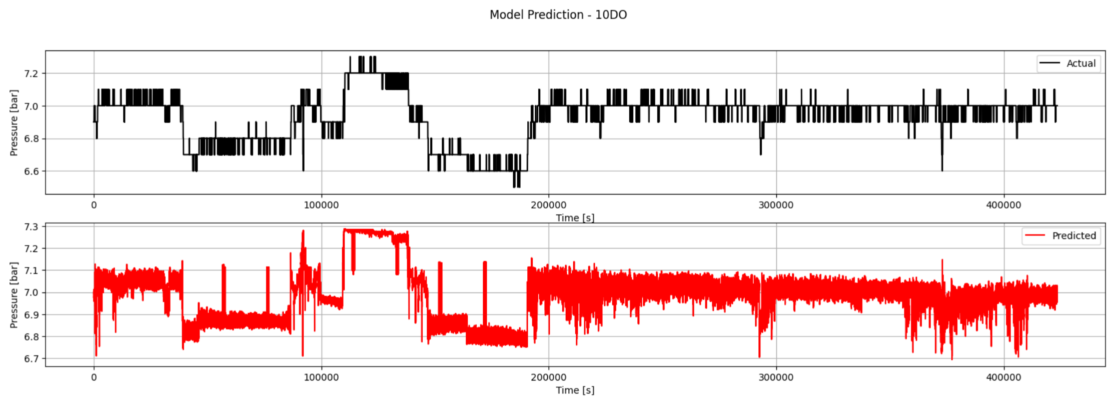


Figure 4.1.13: Predictions for DO with filter

Both models seem to be able to predict the target attribute quite accurately looking at the plots. It was quite unexpected that the non-filtered model would be as accurate as it turned out to be judging by its somewhat poor performance in the accuracy evaluation of MSE and r^2. The filtered model seems to be very accurate, and it would be possible that if the noise in the model prediction was removed as well, the prediction graph would be almost identical to the actual graph.

4.2 Anomaly Detection

4.2.1 Purpose

The EX model were used to test how accurate the predictions would be on an attribute at a time of anomalous behavior. An example of this was in the fourth cylinder, *04EX*, at the 9th of March, 2023. This anomaly is easily identifiable by looking at the correlation heatmap for the EX attributes presented in Figure 3.9.1 in Section 3.9. The dataset contains measurements for a single day, and the whole data set is used for prediction making, which results in approximately 86 000 measurements.

This section will display the results of the EX model's predictions on the said date, comparing the predictions of the healthy cylinder *02EX* with the anomalous cylinder *04EX*. As mentioned in Section 3.3, it was expected that the model would not be able to predict an anomalous attribute with the same level of accuracy compared to predictions made of the same attribute when it's behaving as normal as described in Section 3.9. This hypothesis were subject to validation following the strategies of the scientific method mentioned in Section 2.1 through the results below, where the hypothesis states that the predictions for *02EX* will be accurate, while the predictions for *04EX* will be inaccurate. Should the results indicate that the model is able to accurately predict the values of *04EX*, it would be considered to falsify the hypothesis, thus dismissing the ML approach used for this project.

4.2.2 Predictive Performance

In Table 4.2.1 below, the mean value of the set of predictions for both *02EX* and *04EX* is compared to the actual mean for the corresponding time frames.

Attribute	**Actual Mean**	**Prediction Mean**	**Difference**
02EX	352.019	352.914	0.895
04EX	359.356	329.141	30.215

Table 4.2.1: Comparison of mean value of predictions and actual values for EX with anomaly

Compared to the mean values listed in Table 4.1.1, the difference of the predicted mean and the actual mean for *02EX* is actually smaller when tested on the anomalous dataset, achieving a difference of 3.317 and 0.895. The difference of means for *04EX* is, however, drastically changed for the worse when predicting on the anomalous data, achieving a difference of 1.334 for the non-anomalous *04EX*, and a difference of as much as 30.215 for the anomalous *04EX*. This is a very large difference which clearly stands out compared to the corresponding differences of healthy EX attribute predictions. This strongly indicates that the model is not capable of accurately predict the exhaust gas temperature of the anomalous cylinder, but is still capable of accurately predict for the healthy cylinders at the same time.

4.2.3 Accuracy Metrics

In Table 4.2.2 below, are the accuracy scores MSE and r^2 displayed for *02EX* and *04EX*.

Metric	**02EX**	**04EX**
MSE	24.432	939.281
r^2	0.641	-34.212

Table 4.2.2: Accuracy metrics for EX with anomaly

Compared to the corresponding accuracy metrics achieved by the same model after it was trained (Table 4.1.2), the MSE for *02EX* is lower for the predictions made on the current dataset, achieving a MSE of 24.432, compared to its MSE of 50.129 for the same attribute after training. The r^2 for the same attribute is lower at the current predictions, achieving the score of 0.641 versus the score of 0.922 in Table 4.1.2.

The anomalous attribute *04EX* achieves extremely poor scores (Table 4.2.2) compared to the scores achieved on the same attribute when it was healthy (Table 4.1.2). The MSE is now at the immense score of 939.281 versus the previous score of 30.313 found in Table 4.1.2. This is an incredible high error-rate which clearly indicates that something is wrong with *04EX* in the current dataset. Furthermore, the previous r^2 was at 0.923 which is quite a good score, but now, the score is at -34.212. This result confused the developers quite a bit, due to the r^2 score usually should be a number between 0 and 1 as mentioned in Section 2.5.11.3. Nonetheless, it is safe to say that the achieved r^2 score is a strong indicator of errors in the predictions of *04EX*.

4.2.4 Visualization

Figure 4.2.1 depicts the prediction graphs and the actual graphs for *02EX* and *04EX*.

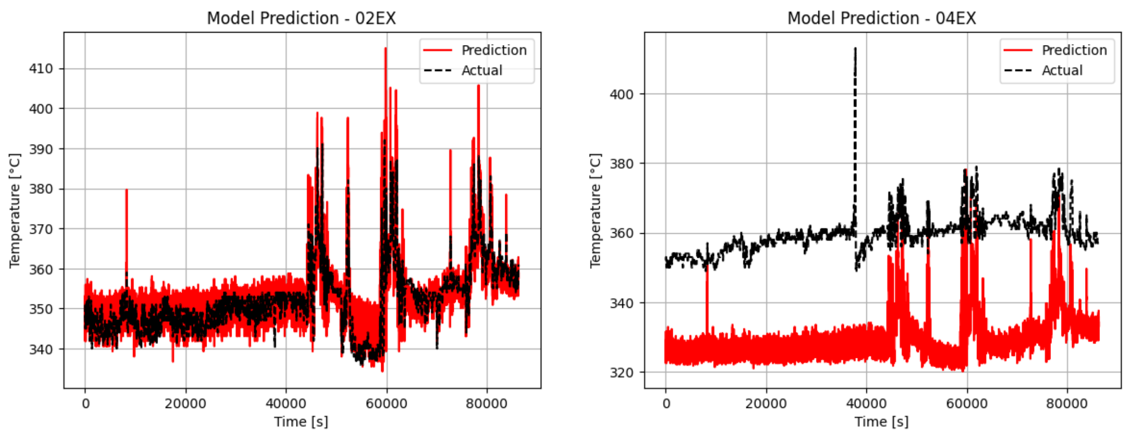


Figure 4.2.1: EX prediction on healthy v.s. unhealthy cylinder

The predictions for *02EX* seems to align well with the actual values. The amount of noise in the predictions seems to be the as it was for *05EX* in Figure 4.1.3. All in all, it seems that the model was able to predict the values of *02EX* quite accurately in terms of magnitude as well as in fluctuations in the data.

On the other side is the prediction graph of the anomalous *04EX* compared to its actual values. This plot clearly stands out compared to any of the other plots presented in this chapter. The two graphs are far apart, seeming to have an average distance of approximately 30°C, which corresponds to the difference of means displayed in Table 4.2.2. Furthermore, the overall shape of the two graphs does not seem to match each other. Between $t = 0$ and $t \approx 38000$, the actual graph seems to have a slight positive linear trend, while the prediction graph seems to be mostly flat when ignoring the noise. Also, the model seems to fail to predict spikes and fluctuations correctly, seeing that the spike in the actual data at $t \approx 39000$ is not present in the prediction graph, although this could be a sensor error and therefore irrelevant, as well as seeing that the three upwards fluctuations in the actual graph between $t \approx 44000$ and $t \approx 85000$ are greatly exaggerated in the prediction graph.

CHAPTER

FIVE

DISCUSSION

This chapter addresses the choices, experiences, and results made throughout the project as well as suggesting future work. It reflects upon how the group has utilized theories and methodology principles as well as discussing whether the results are as expected and its significance.

5.1 Project Management

5.1.1 Project Plan

5.1.1.1 Research strategy

Conducting a research strategy initially based on former knowledge before exchanging ideas with the client and supervisor proved highly effective. The strategy facilitated independent research while also benefiting from professional insight, thereby enhancing the development and refinement of the research and planning process.

5.1.2 Scrum as project management methodology

5.1.2.1 Reason for choosing Scrum

The team has some prior experience with Scrum and prefer it due to its self-assessment approach, openness, and the sprint workflow. Scrum is a widely adopted framework for software development, as evidenced by the following statistics [68]:

- It is utilized by 66% of Agile businesses.
- Among Agile users, 86% are involved in software development.

Considering these factors Scrum was selected as the project management methodology for this project.

5.1.2.2 Our experience

The implementation of Scrum for project management proved to be successful, as the team demonstrated proficiency in operating within the Scrum framework. The initial plan was to conduct weekly sprints throughout the project's duration. However, a two-week pause occurred midway through the project due to examinations in another subject. Additionally, some sprints were extended to two weeks, particularly during periods such as the Easter holiday, where reduced productivity was anticipated.

5.1.2.3 Positive outcomes

- Sprint review meetings were consistently held every Friday, with a few exceptions. These meetings generated valuable feedback and were crucial for the project's development.
- The daily stand-up meetings facilitated a comprehensive understanding among the development team of each member's daily agenda and objectives.

5.1.2.4 Improvements

- Greater awareness of improvement points from the previous sprint could have been achieved. While these points were kept in mind during the sprint, there should have been a more consistent effort to revisit the retrospective and actively implement improvements.
- The supervisor also served as the Scrum Master. More active engagement with the Scrum Master could have provided better guidance on the optimal use of Scrum and ensured that the team followed the Scrum principles.

5.1.3 Scrum as development methodology

5.1.3.1 Our experience

Adopting Scrum as the development methodology was also a decision coming from prior experience and popularity. The implementation of Scrum strategies in the development process was beneficial, though the team were unaware of certain advanced Scrum strategies that could potentially enhance Scrum proficiency.

5.1.3.2 Utilizing Scrum for ML Projects

The group did experience difficulties while adopting agile methodology principles to this project. All previous experience with agile work has been in the context of application development, and the group thinks this could be the reason why they experienced some difficulties when adopting it in this project. It is the group's experience that these forms for agile methods are tailored for software development, but does not seamlessly combine with ML projects.

One of the challenges experienced was when the group attempted to assign user stories to the different issues. While user stories usually assigns a context to an issue from a users perspective, the group found it challenging to do this for issues in this project, as many of the issues were of the type "Find the best way to scale the dataset" or "Merge day-datasets into a month-dataset". From the group's perspective, the assignment specification from KM acted as a user story which encompassed the project as a whole, while sub-problems often was harder to formulate into realistic user stories.

5.1.3.3 Positive outcomes

- The utilization of issue tracking facilitated the team's effective task management. Assigning team members to specific issues ensured clarity in task ownership and prevented overlapping responsibilities.

- The implementation of Scrum enhanced team motivation. Establishing clear development goals for each sprint and striving to achieve these goals improved the developers' efficiency and persistence.

5.1.3.4 Improvements

- The project included only one user story, which primarily described how KM would use the result of the project. There was difficulties identifying other relevant areas for creating user stories. The supervisor concurred with this approach, concluding that it was acceptable to primarily use tasks. Nonetheless, it is believed that a more organized arrangement of issues could enhance the methodology.

- The planning poker strategy was discovered midway through the project. This lead to an attempt at assigning story points using the Fibonacci scale. However, the decision was made to forgo any pointing system, as it was deemed unimportant by the client and scrum master. Although implementing a point system could potentially improve the project, the development team, having operated effectively without it, agreed that it was unnecessary to alter a strategy that was already functioning well.

- The team's adherence to correct sprint principles could have been enhanced. There was a lack of consistency in issue tracking, with delays in moving issues to their respective positions on the scrum board. This meant that issues were transitioned directly from “to do” to “done”. Although this did not significantly impact the workflow, due to effective communication within the group, and mostly working together at campus. However, the team considers that a stricter management of issues would improve the credibility of the sprint management.

5.1.4 Team Work & Communication

The development team maintained good communication throughout the project, and the understanding between the developer team, project client and supervisor was good. However, the developer team would have benefited from more communication with the client and supervisor, in the earlier stages of the project. In the research phase and beginning of development phase, the tendency was to wait until the sprint review meetings to seek answers to questions. Which lead to less efficiency and periods of myopia on specific problems. Midway through the project, the team demonstrated significant improvement on reaching out for help outside of sprint meetings, leading to enhanced efficiency and accelerated development.

5.2 Evaluation of Methods and Achieved Results

5.2.1 Choice of Model Architecture

When addressing the project's objective to detect short-term and long-term anomalies in machinery behavior, the group investigated several research papers, previous projects with implementations, and results based on unsupervised anomaly detection, as described in Section 1.3. Putting everything together, the group decided to adopt LSTM as a first test of the problem objective, according to the findings and advice from the supervisors.

In Section 2.5.5, the LSTM model is described as particularly adept at identifying and learning both short- and long-term trends in time series data. This capability matches exceptionally well the characteristics of the data set used in this project, which included time-series data. In any case, it was not the only claim that LSTM was the reason for the choice of method, but also several research articles as mentioned earlier. For example, the findings reported in the article "A deep learning method for the prediction of ship fuel consumption in real operational conditions" from [3], as detailed in Section 1.3, provided a comparative analysis of various machine learning models, emphasizing MSE as a metric for prediction accuracy, as described in Section 2.5.11.2. Notably the Bidirectional LSTM model achieved a commendable average MSE of 0.0261 across 5-fold validation, ranking third best in terms of performance. This was surpassed only by the LSTM with attention model and the Bidirectional LSTM with attention model.

These results reinforce the robustness of LSTM-based architectures, particularly in handling time series data. Incorporating the findings from "A deep learning method for the prediction of ship fuel consumption in real operational conditions" not only highlights the capabilities of LSTM-based models, but also substantiates the group's results, which is demonstrated and described in Chapter 4, which demonstrate the effectiveness of LSTM architectures for time series data. The consistent performance of the Bidirectional LSTM and its variations across different studies, including this project, provides strong evidence supporting the use of LSTM models.

However, given the insights from the article "A deep learning method for the prediction of ship fuel consumption in real operational conditions", exploring the two better-performing LSTM architectures with attention mechanisms presents an opportunity for future research. By integrating these two deep learning models, the group could gain a deeper understanding of the patterns within the data, potentially leading to even more accurate predictions. These considerations will guide the group's future research directions and effort to refine the current model further.

The LSTM model, chosen for its ability to effectively analyze and predict patterns in time-series data, aligns precisely with Kongsberg Maritime's objectives as described in Section 1.2.3. By learning both short- and long-term trends, LSTM provides a solution to KM's needs for detecting and classifying patterns.

5.2.2 Alternative Model Architectures

Anomaly detection can be approached using a variety of model architectures. IEEE's research article "Machine Learning for Anomaly Detection: A Systematic Review" includes a table listing the most frequent models used for anomaly detection across several projects [5, tab. 4].

The most frequent technique according to IEEE's report [5, tab. 4] was SVM (Support Vector Machine). It appears frequently among projects as either a standalone model or in combination with other models i.e. hybrid models. Other popular models include IF (Isolation Forest) which is an ensemble model which were not combined with other models, and regular NN's (Neural Networks). LSTM was also appeared at a quite high frequency both as a standalone as well as part of a hybrid model.

The group did consider both SVM and IF for this project, but it was decided to use LSTM mainly due to the reasons given in Section 5.2.1 as well as due to the approach proposed in Section 3.3.

5.2.3 Model Assessments

5.2.3.1 Cylinder Exhaust Gas Temperature

During training, the EX model showed positive signs indicating that it managed to learn and generalize the data it was fed, learning the underlying patterns and dependencies in the data (Figure 4.1.1). The stable validation loss indicated that the model was properly fitted with the data.

When presented with unseen data, the model seemed capable of predicting all the EX attributes with good accuracy. In addition to predict the fluctuations of the attributes over time, the model was also being able to do accurately predict the actual temperatures for each cylinder even when these were different for each cylinder, as seen in Table 4.1.1, where the actual mean of different attributes ranged between 332.110 versus predicted mean of 333.444 for *04EX*, and an actual mean of 351.335 versus predicted mean of 350.522 for *06EX*.

The accuracy metrics of the model for each attribute were also quite good, having almost all predictions achieve an r^2 score over 0.92, and MSE scores between 25 and 63 (Table 4.1.2).

When plotting the prediction graphs with their corresponding actual graphs (Figure 4.1.2 & 4.1.3), it became clear that the model was more than capable of predicting the attributes over time at a reasonable precision level.

The group is satisfied by the performance of the EX model and confident that it is accurate enough to meet the requirements of the client. However, it would be wise to test the model on data at dates longer away from the data it was trained on. While its reasonable to believe that the model will continue to be accurate in terms of predicting the overall pattern of the attributes, it is possible that the predicted temperature level will differ more from the actual temperature in terms of magnitude if the data it predicts upon is not continuous with the data it was trained on. If this is the case, it could likely be improved by training the model with more data, and maybe by adding an input attribute capable of indicating the temperature of the respective cylinders. The group does, however, consider it to be more important that the model is able to accurately predict the fluctuations in the data than predicting the actual temperature in each cylinder precisely.

5.2.3.2 High-Temperature Cooling System

The HT model seemed to learn gradually throughout the training phase, but the validation loss was a bit unstable as seen in Figure 4.1.4. The training were also stopped early due to the early stopping mechanism made the overall training quite limited.

Even with the short amount of training, the model was very accurate at its predictions of *06HT*, attaining a low MSE, a low difference in mean, and a high r^2 score (Table 4.1.3 & 4.1.4).

Judging by the plot of the prediction graph with the actual graph (Figure 4.1.5), the trained model was very accurate in terms of both temperature accuracy as well as how the attribute developed over time. The two graphs were very similar to each other, seeming like only noise could tell them apart.

Based on the achieved results, the group is confident that the model is accurate enough for anomaly detection. The target attribute has a quite linear nature with occasional fluctuations, which makes it reasonable to believe that the model will continue to predict the data accurately. The most usual source of anomalies in the HT will likely be in the form of a decreased or heightened water temperature in *06HT*, which should be easy to identify considering that the model most likely will continue to predict a stable temperature with occasional fluctuations which can be identified by corresponding changes in the input attributes.

The model did not receive as much training as intended. This could probably affect the accuracy of the predictions in the long run, especially when predicting upon data which is not chronologically continuous with the data it was trained on. This could likely be remedied by training the model on more data, but in order to be able to do this, the root cause for the unstable validation loss must be resolved.

5.2.3.3 Low-Temperature Cooling System

Judging by the loss functions from the training phase of the LT model (Figure 4.1.6), it seemed like the model was about to be overfitted. The early stopping mechanism stopped this and restored the model's state after completing epoch two, resulting in a rather short training time.

Once the model was trained, it scored remarkably well in the comparison of means test (Table 4.1.5) as well as the accuracy scores (Table 4.1.6), having the mean of predictions deviate by the actual mean by 0.021, a MSE of as little as 0.008, and an r^2 of 0.997.

The plotting of the prediction graph with the actual graph (Figure 4.1.5) proved that the model was very accurate in its predictions at that time. The attribute it predicted, *10LT*, was almost perfectly stable, only being interrupted by a single downward spike and a temperature fall in the last period of the prediction time frame. Both these changes were accurately predicted by the model.

The attribute subject to prediction for this model is quite uniform, which makes it reasonable to believe that the model will preserve its accuracy when predicting on future data as well. The model did not have a lot of training time, but the amount of training it received may be sufficient after all, considering how stable the target attribute is.

5.2.3.4 Lube Oil System

Throughout the training phase, the LO model seemed to continuously learn from the data, judging by the loss function shown in Figure 4.1.8. The validation loss seemed to be quite stable as well, but the early stopping mechanism restored to the model's weights in the 2^{nd} epoch. This resulted in a short amount of training time.

The model was quite similar to the LT model in terms of prediction accuracy, having a difference of 0.017 between the mean of the predicted values and the actual mean (Table 4.1.7), a very low MSE of 0.003, and an r^2 of 0.972 (Table 4.1.8).

After plotting the prediction graph and the actual graph (Figure 4.1.9), the first observation that was made was that the attribute, *10LO*, had a quite uniform nature which resembled the attribute *10LT* in the low-temperature cooling system. Apart from having a downwards spike in the middle of the time interval, the attribute was almost completely stable, having a negligible degree of noise. The prediction graph seemed to be almost perfectly identical to the actual graph, separated only by the somewhat more significant degree of noise in the prediction graph.

Since the target attribute, *10LO*, has such a stable nature, the model will most likely be able to continue predicting it very accurately. Anomalies in the lube oil system will almost certainly cause the attribute to be more unstable, which could easily be detected by comparing the actual attribute values with the stable predictions made by the model. The model did not receive much training, and it would probably be wise to decrease the sensitivity level of the early stopping mechanism, since it activated even though the validation loss was quite stable.

5.2.3.5 Fuel Oil System

There were two DO model's trained for the fuel oil system, where one was trained on data which was median filtered, while the other trained on unfiltered data. The filtered model seemed to get the most out of the training phase (Figure 4.1.11). The non-filtered model seemed to simply stop learning quite early (Figure 4.1.10), which may be a symptom of the vanishing gradient problem (see Section 2.5.4).

Both models performed quite well according to the quantitative measurements of prediction accuracy (Table 4.1.11 & 4.1.12). The filtered model achieved better results than the non-filtered model, where the latter scored quite low in terms of r^2, achieving a score of only 0.115.

Plotting the prediction graphs next to the actual graphs for both models showed how the median filter affected the performance of the model's (see Figure 4.1.12 & 4.1.13). The non-filtered model appeared to accurately predict the fluctuations in the target attribute *10DO*, and was also able to do so with much less noise than what the actual data had. This model will likely be quite useful for future predictions due to its ability to be precise in its predictions and exclude the noise. The filtered model produced a prediction graph which had a close resemblance to the actual, median filtered graph, both in terms of magnitude and shape. All in all, both models are likely to be acceptable for anomaly detection, each emphasising different qualities, but in terms of accuracy, the best model was probably the filtered model.

5.2.4 Capability of Anomaly Detection

The EX model was also tested on anomalous data in order to see how it would affect the accuracy of its predictions (see Section 4.2). As expected, the model preserved its accuracy for the non-anomalous cylinders, but proved to be severely inaccurate in its predictions of the anomalous cylinder in terms of both fluctuations and magnitude. As seen in Table 4.2.2, the model achieved an MSE score of 939.281 and an r^2 score of -34.212 for its predictions of the anomalous cylinder *04EX*, while the predictions on the healthy cylinder *02EX* gave an MSE of 24.432 and an r^2 score of 0.641. As mentioned in Section 2.5.11.3, a negative r^2 score means that the model fits worse with the data than a horizontal line. This proved that the idea of detecting anomalies by comparing predictions and actual data was feasible, meaning that the hypothesis was not falsified in this experiment in accordance to the hypothetical-deductive method (see Section 2.1).

5.2.5 Reflecting upon Model Evaluation Methods

The models were evaluated by comparing the mean value of predictions with actual mean, by calculating the MSE and r^2 as well as qualitative evaluation through plotting graphs. Although these measures seemed to be quite efficient while validating the models, it is likely necessary to look into other ways to validate the models.

While the evaluation methods used in this project does make it possible to quantify the accuracy of the models, there are other factors than accuracy which could be taken into consideration. Such factors could be e.g. sensitivity level. Furthermore, there are other accuracy measures to use, such as Mean Absolute Error and F1-score.

MSE and r^2 was chosen due to their ability to describe accuracy in terms of magnitude and fluctuations respectively, which were considered to be the two most important factors in this scenario. It was also decided that it would be wise to actually plot the predictions next to the actual values as well in order to get a visual representation of the accuracy as well. When the system is to decide whether there are anomalies or not in an attribute, it is dependent on the MSE and r^2 score in order to address this. All in all, the group is satisfied with the measures used as of now, but they do not want to rule out that it could be wise to validate the models using other measures as well, which would likely give the models more credibility and the anomaly detection system more precise.

5.2.6 Sources of Error

5.2.6.1 Overfitted Models

Even though measures were made in order to prevent overfitting, where the most important ones was in the form of early stopping mechanisms, model dropout layers, and thorough data preprocessing, it does not guarantee that all models are completely immune to overfitting. If overfitting still occurred while these measures were taken, it would likely be harder to recognize. Since the model was trained on continuous data, it would maybe still be possible for the model to accurately predict attributes from data close to the data it was trained on, but the model will probably achieve progressively less accurate results if the distance between the training data and the input data increases.

5.2.6.2 Insufficient Validation

The accuracy of the model's were assessed by having them predict the remaining 20% of a month-sized dataset which they were trained on. The LSTM architecture excels at sequential data and is capable of learning long-term dependencies in the data. Even though testing the model on data close to the training data in terms of time seemed to be the most appropriate choice by the group, it is possible that this presented better results than what they are in general. This was tested to some extent by having the EX model predict data which were a month apart from the training data, but the model still managed to score with sufficient accuracy.

5.2.6.3 Anomalies in Training Data

All models were trained on data originating from the same month. This dataset was thoroughly inspected to ensure that the models was not trained on data with anomalies in them. Still, it may be possible that some anomalies were undiscovered during this inspection. If this is the case, this could affect the model's usability in terms of anomaly detection, since the models being unfamiliar with anomalous patterns forms the basis for this approach.

5.3 Classification

Anomaly classification was part of the thematic for this project, but the group was not able to use ML for this purpose. This was mainly due to the lack of data which could be used for classification.

Kongsberg Maritime have specified requirements for anomaly classification compatible for their needs. The main requirement was that every type of anomaly, which could be separated from each other by its failure pattern, must be initially classified by an employee in KM which had expertise in the product in which the anomaly occurred. KM did not have enough data of this to use for training a classification model. This made it impossible to use ML in order to meet these needs in traditional ways.

It is possible that the group could have been able to find a different approach which may have resulted in an anomaly classification system, but this was not possible given the time set off to this project. KM has expressed understanding for this, and acknowledged that also including anomaly classification in the project would be too ambitiously, stating that there are not sufficient amounts of data in the industry for this sort of task as of now anyway. However, the group is satisfied with the achievement of producing a sophisticated anomaly detection system.

It could likely be possible to train models for anomaly classification in the future, given that more data with labeled failure patterns is attained. If this is the case, it would be possible to upgrade the current system of anomaly detecting ML models with the functionality of classification as well.

5.4 Alignment with Project Motivations and Objectives

5.4.1 Development team motivation

As mentioned in Section 1.1, the group wanted to be challenged with an industrial-level machine learning problem. This motivation aligned well with the project's objective, and the team was satisfied with the level of difficulty presented. The project provided the opportunity to conduct independent research and select the preferred approach, while maintaining involvement with the industry through collaboration with the client at KM

5.4.2 Objectives

KM's objectives, as listen in Section 1.2.1, aim to enhance the product for their customers. These objectives are contingent upon the successful execution and proper functioning of the model. Achieving a functional model that is beneficial for KM aligned with the interest of both the development team and KM, as it supported KM's objectives and the development team's motivation. Consequently, both parties strived to achieve a productive collaboration, working towards this common goal.

5.4.3 Deviations

5.4.3.1 Detecting Anomalies Indirectly

Initially, the strategy involved directly predicting and detecting anomalies through the model. This approach as altered to one where models predict the values for specific sensor subgroups, and when deviations between actual and predicted values occur, it will alert a anomaly. This deviation from the original plan was due to the adoption of supervised machine learning, which offers a broader selection of models and facilitates easier validation. Although the initial solution with unsupervised machine learning could have been more exact in predicting anomalies, the chosen approach was endorsed by KM, as it aligned with the organization's values and objectives.

5.4.3.2 Multiple Models

The original intention was to develop a singular model encompassing all sensor attributes within the dataset. However, the team encountered some difficulties with this approach. Therefore, the team decided to consult with the client from KM. From this consultation, a decision was made to segment the dataset into subgroups and to develop separate models for these segments. This strategy was positively received by KM, as it would still benefit their application, and with this approach the models could be relevant for more motors.

5.4.3.3 Framework for a Classification Model

The initial idea for the project involved the creation of a machine learning model capable of both detecting and classifying anomalies. However, due to a lack of data on classified anomalies, the development of a classification model was deemed unfeasible. Instead, a framework for how such a classification model might be developed was proposed. KM was satisfied with this solution as it provided them with valuable information regardless of the data limitations.

5.5 Implementation in Real Life

5.5.1 Flexibility & Scalability

Implementing the trained models as they are now would probably be fine as long as it is implemented on the vessel in which the data the models were trained originates. KM would maybe prefer to train the models on more data before implementing it in full scale.

However, it is not guaranteed that the trained models will be able to accurately predict attributes for other vessels, since there are many factors to consider such as how the vessel usually operates, how the ship is loaded, the structure of the engine, placement of sensors, etc. The EX model, for example, would not be compatible for anomaly detection for an engine which is not an eight-cylinder build. It will therefore be necessary to train EX models which are tailored to the specific engine types. On the other side, one could argue that some of the other models such as the DO and LO model could turn out to be more flexible, since these models are simpler and is based on isolating a filter inside the respective systems, meaning that as long as there is data for this on a vessel, these models are compatible with them as they are now.

The group did have this in mind when writing the code for training the models, attempting to standardize it as much as possible (see Section 3.8.1). By having a standardized script for model training, the process of producing new models can be streamlined, since it would only require to change the input and target attributes of the model. It will still be necessary to assess the models, as well as test out different configurations in order to get the best results.

All models were designed in such a way that it would make it possible to perform anomaly detection on different time intervals. In other words, the group found it important that it would be possible to use the model for anomaly detection in a short interval such as a single day, or longer intervals such as an entire month. This can be done by adjusting how much observations are fed to the model for prediction. The thought was that the models could be used to first inspect an entire month of data, and then inspect each day respectively in order to pinpoint the exact time of anomalous behaviour.

5.6 Challenges

5.6.1 Storage of Data

Upon receiving the dataset from Kongsberg Maritime, our group encountered challenges due to the large size of the data. However, the group quickly resolved this by using Google Drive as a solution for sharing and storing. This strategy proved to be beneficial as it seamlessly integrated with Google Colab, the platform selected for executing the project. By utilizing Google Disk, the group was able to directly import the dataset into Google Colab, which significantly simplified our data management and facilitated a more efficient workflow.

5.6.2 Version Control

Moving from data storage solutions to software development practices, the group encountered another substantial challenge: *Implementing an effective version control system.* Initially, the group considered using a traditional platform like GitHub, integrated with Google Colab, to manage the codebase. However, the group found it insufficient for tracking individual contributions as when committing from Colab, the whole file is replaced rather than changing the specific lines in the file which was altered, which undermines the usage of version control. Consequently, the group implemented a documentation process into the routines. Each team member was required to report any changes and findings during the daily meetings, ensuring that the group maintained a high standard of code quality and consistency throughout the project.

5.6.3 Computation

The datasets used in this project were rather massive. This did introduce some challenges in terms of computational power. The most significant challenge occurred when the group tried to merge several datasets into a single dataset containing data for a whole month of vessel operation. This turned out to be rather expensive in terms of both time and computational power. This was firstly attempted using one of the group members personal computer, but the computations necessary used more RAM than what the computer had. This was solved by running the script in Google Colab, which offered larger RAM.

Furthermore, one of the disadvantages of LSTM model's is that they are rather consuming in terms of computational power, as mentioned in Section 2.5.5. Google Colab provided a limited amount of so called "compute units", and these were quickly spent when training the models. This problem was solved partially by being more conscious when choosing what hardware components to use when training the models. As mentioned in Section 2.2.1.1, Colab provides a variety of hardware components such as the A100 GPU and TPU. It was discovered that using a TPU resulted in only slightly slower processing time, but it was much more efficient in terms of compute units compared to the A100 GPU.

5.7 Assessment of References

The resource material which is referenced to in this thesis forms the basis of the foundation of the research conducted throughout this project. Theoretical background, as well as methodology, and discussion of results are strengthened through a variety of literature.

When selecting which literature to use, the group has to their best ability adept to proper source criticism. Most of the literature used in this project originates from well-known and acknowledged sources, such as syllabuses, scientific reports, IBM, and respected academic institutions such as Massachusetts Institute of Technology, Technical University of Denmark and Stanford University. Other sources includes websites such as Medium, Techtarget, and Science Direct.

All sources were evaluated before being used in this project. When the credibility of a source was doubtful, the group tried to validate its content by cross-validating it with other sources citing the same topics. Some of the material used in this project originates from internal or unpublished documents. Such documents were given to the group by Kongsberg Maritime.

The group would also like to point out that there are some figures in this report which is from references, and that these the origins of these figures are then referenced to in their captions. However, some figures are produced by the group. These figures includes plots made such as correlation heatmaps and graphs, which were produced by the scripts made in this project. Other figures includes flowcharts and illustrations such as Figure 2.1.1 and Figure 3.3.2, which were made by the group using Figma drawing software [69].

5.8 Future Work

There are several enhancements possible for the health management system. Given that Kongsberg Maritime would like to proceed with the solution proposed in this thesis, there are several alternatives likely to be considered.

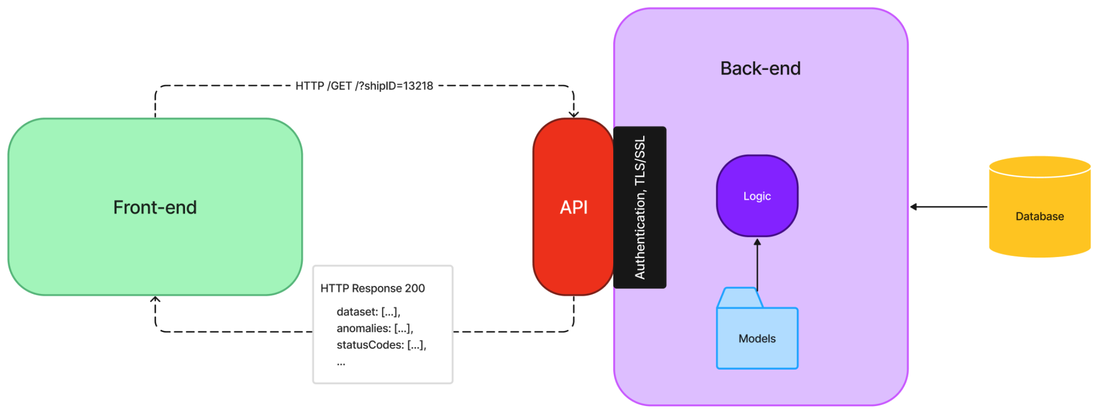
5.8.1 Other Machines

Kongsberg Maritime's current condition monitoring system for naval vessels is not only applied for ship engines exclusively. Their system is also used to monitor low-speed rotating machines such as winches, and steering gears, and high-speed rotating machines such as generators, thrusters, and pumps. If this project is to be developed further in the future, it would be natural to expand it to support anomaly detection for these sorts of machinery as well.

In order to implement anomaly detection for other machinery, it will be necessary to process datasets with sensor measurements from the machines. Given that there are high quality data available, it could be possible to train ML models for the different machines following the same strategies as the ones used for this project.

5.8.2 Embedded System

If the anomaly detection system is to be implemented into KM's systems, it could be so in the form of an embedded system. One way this can be done is to develop a client-server application. Such a system would likely consist of the following components described in Figure 5.8.1:



```
HTTP Response 200
    dataset: [...],
    anomalies: [...],
    statusCodes: [...],
    ...
```

Figure 5.8.1: Simplified application architecture illustration

5.8.2.1 Back-End

Since all models are exported to files, they can be stored on a server running a back-end application. The back-end will be responsible for the anomaly detection, using the models to predict attributes and compare it to the actual values. The back-end would also need appropriate security features such as user authentication and encryption.

5.8.2.2 Front-End

The front-end could be in the form of a desktop application or a website. Here, KM could keep track of all the vessels they are monitoring. The application could display the status of the different systems, such as the cylinders, and see the attribute values over time as well as marking time frames where anomalous behaviour occurs and what the expected values should be at that time.

5.8.2.3 Application Programming Interface

It will be necessary to implement an API, allowing the client and the server to communicate with each other. One possible solution could be to implement REST API endpoints for authentication and data transmission. With this solution, front-end could send requests to the back-end for updates regarding a vessel, and the back-end could respond by sending the relevant information, all through the HTTP protocol with TLS/SSL encryption (HTTPS).

5.8.2.4 Database

The back-end would likely be connected to a database containing sensor data, user data, vessel information, etc. The database could also possibly receive sensor data continuously from another server which is responsible for collecting and processing sensor data from all the vessels.

5.8.3 Classification

Due to the lack of data necessary to implement ML algorithms for anomaly classification, it would not be possible due implement an anomaly classification system using ML at the present time. It could possibly be done in the future if there were to be more data. However, a non-ML classification system could be implemented given that the embedded system discussed above was made.

The idea is that the front-end application could allow product experts to select time frames with anomalous behaviour in various machines, and label it with a diagnosis of the unit. This classification could then be sent to the back-end, making sure that the product expert's assessment of the given anomaly is registered. Such a solution could also be the start of a process of gathering samples of anomalies and their corresponding classification label, which later could be used in order to train a ML model for anomaly classification.

5.8.4 Forecasting

Another enhancement of the anomaly detection system could be to utilize ML to predict the future development of attributes. This would be highly beneficial, since this would make it possible to schedule repairs and replacement of machine components before they are wearied out.

CHAPTER
SIX

CONCLUSIONS

6.1 Client's Assessment

The contact person for KM was David Vågnes, who is considered to be the main client for this project. Upon completing the models, he was asked to make a statement to include in this report. The statement was to act as a review of the results, and Vågnes responded as follows:

> *In an era of increasing digitalization in the maritime industry, coupled with heightened awareness of sustainability and financial constraints faced by ship owners, the demand for predictive maintenance through condition monitoring has surged. This demand is further fueled by the global trend in AI and ML methods, notably exemplified by ChatGPT's impact in early 2023, prompting a rush towards their application in various domains.*
>
> *This bachelor thesis, undertaken in collaboration with Kongsberg Maritime, addresses these converging trends by leveraging Machine Learning to analyze data from real sailing vessels, facilitating predictive maintenance through anomaly detection. The demonstrated efficacy of this approach in detecting long-term degradation highlights its potential for commercialization, contingent upon integration within established frameworks for a comprehensive monitoring service.*
>
> *Moving forward, the focus shifts towards commercializing the developed methods. This involves refining the framework for seamless integration into the commercial solutions of Kongsberg Maritime, more testing at known failures to ensure quality and executing targeted marketing strategies for market penetration. By addressing these steps, the potential of the developed methods can take part in shaping predictive maintenance of the maritime industry, delivering enhanced efficiency and cost savings.*
>
> *- David Vågnes, May 15th 2024*

Vågnes was also asked to describe his experience of the collaboration and communication between himself and the group. To this, he made the following statement:

> *The collaboration with the student group has been marked by professionalism and diligence. Regular bi-weekly meetings have facilitated discussions on technical intricacies and project progress, supplemented by proactive engagement outside formal sessions. This proactive approach has been instrumental in driving project success.*

6.2 Answering the Problem Statement

This thesis presented a ML approach for anomaly detection in engines of vessels for KM. The proposed framework for anomaly detection consists of five LSTM models addressing the most important subsystems of a ship engine, and has proven proficiency in accurately predicting attributes in the system, making it possible to indirectly detect anomalies by comparing the predicted and actual state of the engine.

The primary objective of this project was articulated as a problem statement in Section 1.2.3. In order to determine whether the proposed solution is satisfactory to the problem statement, both the group's and the client's assessment will be taken into consideration.

In Section 5.2.3, the five models showcase their adeptness in accurately predicting attributes. Additionally, Section 5.2.4 delves into the approach of detecting anomalies, illustrating the models' ability in this regard. While the models may not classify anomalies, Section 5.8.3 discusses a prospective solution involving these models.

In Section 6.1, the client expressed KM's satisfaction with the achieved results, emphasizing the transition towards commercializing the developed methods. The client highlighted the potential for these methods to significantly influence predictive maintenance in the maritime industry, leading to improved efficiency and cost savings. Regarding anomaly classification, the client verbally endorsed the prospective solution mentioned in Section 5.8.3 as a promising option for future investigation.

The problem statement poses the question: "How can machine learning be implemented in order to detect and classify anomalous behavior in machinery of various types?" A methodology for anomaly detection has been developed and validated, alongside a proposed approach for anomaly classification. Considering both the group's evaluation and client feedback, it is determined that this thesis has produced a viable solution to the problem statement.

6.3 Conclusions for the Development Process

The group is satisfied with the work they have put in for this project, and thinks the group has worked in harmony both internal as well as in cooperation with the client. The client expressed a similar view, stating that the collaboration with the student group has been marked by professionalism and diligence.

Moreover, the group is satisfied with their consistent application of agile methodology throughout the project's duration. Overcoming the challenges associated with integrating agile practices into ML projects has provided the group with invaluable insights, enhancing their readiness for future endeavors.

CHAPTER
SEVEN

SOCIAL IMPACT

This chapter addresses the social impact of the project and sets the consequences of the project into three perspectives being ethics, sustainability, and financial impact.

7.1 Ethics

Artificial intelligence is one of the fastest developing technologies in present time. The rise of "thinking machines" introduces a vast number of ethical questions which should be taken into consideration.

Engineers plays a critical role in introducing new technologies in order to contribute to the welfare of the planet and the life it contains. Due to the massive responsibility entrusted to the world's engineers, it is of paramount importance that they possess good understanding of ethics and critical thinking. The Institute of Industrial & System Engineers has defined four fundamental principles that are meant to ensure that engineers uphold and advance the integrity, honor, and dignity of the engineering profession [70, paragraph. 1]. The four principles are defined as follows:

1. "Using their knowledge and skill for the enhancement of human welfare"
2. "Being honest and impartial, and serving with fidelity the public, their employers and clients"
3. "Striving to increase the competence and prestige of the engineering profession"
4. "Supporting the professional and technical societies of their disciplines"

AI may help the human race solve complex problems which can contribute to enhancements in human welfare, but one must approach new technologies with reason and should not be ignorant to consequences which may follow. AI introduces new philosophical and ethical questions regarding what intelligence and consciousness is, distribution of responsibility, and privacy.

The group did carefully evaluate this project's theme, ensuring that it satisfied the terms of engineering ethics by conducting such a project. It was concluded that this implementation of machine learning did not set the privacy of human beings at risk, and that a possible solution would not interfere with the security of employees of Kongsberg Maritime or the crew of the vessels in which the solution might be implemented, but rather enhance security measures for the crew as well as reducing environmental influences as a product of inefficient use of equipment.

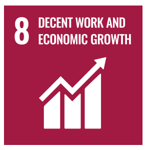
7.2 Sustainability

7.2.1 UN's Sustainability Goals

The United Nations has defined 17 goals known as the sustainability goals. Among these goals, the group would argue that the ML implementations of this project could contribute to the following goals [71]:

7.2.1.1 Goal 8 - Decent Work and Economic Growth

This goal aims towards promoting sustainable economic growth with productive employment and decent work for all. Implementing a autonomous anomaly detection system will likely lead to reduced costs in vessel repairments, which would increase the profits.



[72]

7.2.1.2 Goal 9 - Industry, Innovation and Infrastructure

The 9th goal aims towards promoting sustainable industrialization as well as foster innovation. The HMADC project builds on innovative technologies within ML, and could likely be used to make marine industrialization more sustainable.

[72]

7.2.1.3 Goal 12 - Responsible Consumption and Production

Goal 12 emphasises the transition towards sustainable consumption and production. One could argue that this project can satisfy this goal, since the anomaly detection system will enable KM to monitor engine components better, which could lead to increased efficiency in terms of component attrition and fuel usage.

[72]

7.2.1.4 Goal 13 - Climate Action

This goal towards taking measures to prevent climate changes. By implementing the anomaly detection system made in this project, it could reduce the CO_2 footprint of the vessels, since improper usage of the vessel or worn out components tends to increase the CO_2 emissions.

[72]

7.2.1.5 Goal 14 - Life Below Water

Goal 14 aims towards conservation of the oceans as well as sustainable usage of marine resources. Ensuring more optimal usage of vessels at sea, this is likely to contribute positively towards ocean conservation. Given that this project contributes towards goal 13 - climate action, it is reasonably to believe that it would contribute to this goal as well.

[72]

7.3 Financial Impact

The global predictive maintenance (PdM) market has witnessed remarkable growth, reaching a valuation of \$5.77 billion in 2022, and is poised to grow from \$7.57 billion in 2023 to \$66.46 billion by 2031 (see Figure 7.3.1), at a compound annual growth rate (CAGR) (See section 2.8.1) of 31.2% during the forecast period (2024-2031), stated from Skyquest [73].

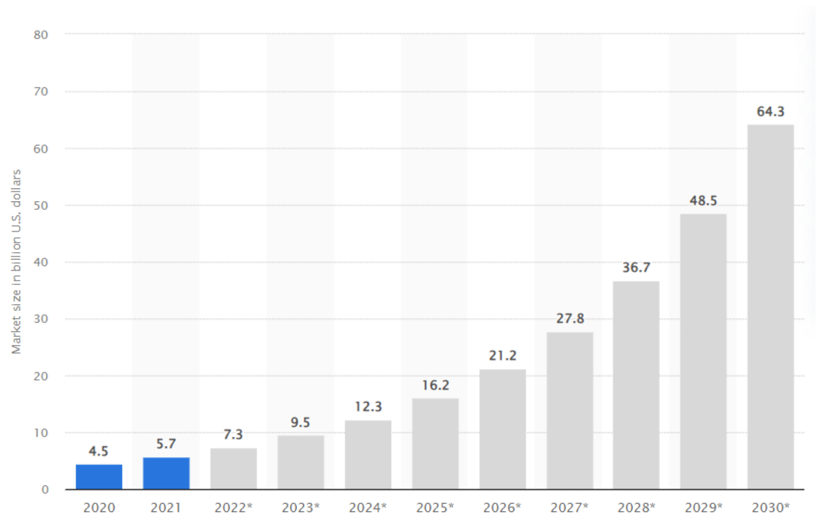


Figure 7.3.1: PdM Market Value forecast for 2022 to 2030 [74]

Consider the Figure 7.3.2 below, the graph illustrates the predictive maintenance market share distribution across various regions from 2019 to 2013, the figure is obtained from [75]. North America consistently maintains the largest share, indicating robust growth and adoption in the region, followed by Europe and Asia Pacific. This positive trend underscores the expanding global research and relevance of predictive maintenance technologies. The forecasted growth into 2031 suggests a continuing global expansion and adoption of these technologies.

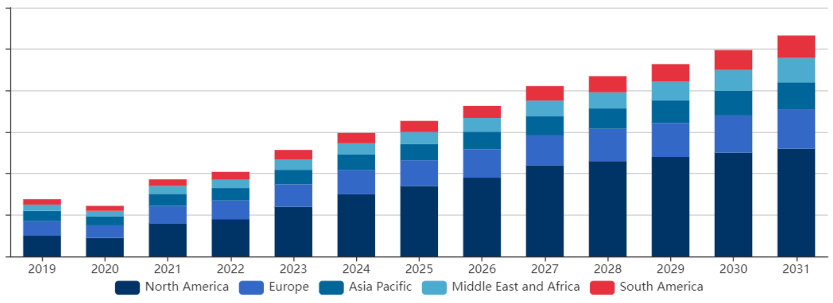


Figure 7.3.2: PdM Market Share (%) by Region (2019-2031) [75]

7.3.1 Case 1 - Injector fouling detected

Kongsberg Maritime has implemented a predictive maintenance system as part of its Engine Health Management for PSVs [2]. This system is designed to enhance operational efficiency by predicting and addressing engine issues before they lead to failures.

The predictive maintenance system identified injector fouling in one of the PSV engines, which typically leads to increased fuel consumption and reduced engine efficiency. Without intervention, this issue could escalate, resulting in higher operational costs and potential downtime.

Following the system's recommendation, a service intervention was planned to clean the injectors and retune the engine. This preemptive approach aimed to restore engine performance without disrupting normal operations.

7.3.1.1 Results

The results of implementing predictive maintenance for the PSV highlight significant improvements across multiple operational and financial aspects. Initially, by addressing injector fouling proactively, the PSV quickly returned to optimal operation with engines performing at standard levels. This early intervention not only restored engine efficiency but also prevented unexpected downtime, which typically results in considerable costs.

In terms of cost savings, the early detection and correction of the issue led to a notable reduction in engine wear and fuel consumption, culminating in a savings of approximately 150,000 NOK over three months. Complementary to these financial savings, power consumption analyses conducted before and after maintenance showed a significant reduction in energy usage.

In conclusion, the predictive maintenance (PdM) market is on a robust growth, as evidenced by its expansion from \$5.77 billion in 2022 to an estimated \$66.46 billion by 20131 as illustrated in Figure 7.3.1, with a CAGR of 31.2%. This grows is a proof to the increasing adoption and integration of PdM technologies across key global markets, particularly in North America, Europe, and Asia Pacific. The investments in these regions reflect a broader industry trend towards optimizing operational efficiencies and enhancing maintenance strategies.

REFERENCES

[1] Vågnes, David. *Health Management - Anomaly Detection & Classification.* [Internal document]. Oct. 2023.

[2] Vågnes, David & Kongsberg. *Health Management.* [Internal document].

[3] Mingyang Zhang et al. *A deep learning method for the prediction of ship fuel consumption in real operational conditions.* Apr. 2024. DOI: `https://doi.org/10.1016/j.engappai.2023.107425`. (Visited on 05/12/2024).

[4] Peihua Han et al. *Fault Detection with LSTM-based Variational Autoencoder for Maritime Components.* Aug. 2021. DOI: `https://doi.org/10.1109/JSEN.2021.3105226`. (Visited on 05/12/2024).

[5] Ali Bou Nassif et al. *Machine Learning for Anomaly Detection: A Systematic Review.* May 2021. DOI: `https://doi.org/10.1109/ACCESS.2021.3083060`. (Visited on 05/13/2024).

[6] Bo Wang et al. *Research on anomaly detection and real-time reliability evaluation with the log of cloud platform.* Sept. 2022. DOI: `https://doi.org/10.1016/j.aej.2021.12.061`. (Visited on 05/13/2024).

[7] Hanne Andersen and Brian Hepburn. *Scientific Method.* June 2021. URL: `https://plato.stanford.edu/entries/scientific-method/` (visited on 04/15/2024).

[8] Kjelsberg, Ronny. *Teknologi og vitenskap - Histore, metode, etikk og miljø.* Vol. 2. Universitetsforlaget, 2019. ISBN: 978-82-15-02480-6.

[9] Google. *Google Colaboratory.* May 2023. URL: `https://colab.google/` (visited on 04/15/2024).

[10] Google. *Colab Pro.* Apr. 2024. URL: `https://colab.research.google.com/signup` (visited on 04/18/2024).

[11] Google. *Where are my notebooks stored, and can I share them?* Oct. 2017. URL: `https://research.google.com/colaboratory/faq.html` (visited on 04/15/2024).

[12] Atlassian. *Atlassian.* 2024. URL: `https://www.atlassian.com` (visited on 05/15/2024).

[13] *Jira Scrum Board.* Apr. 2024. URL: `https://jira.iir.ntnu.no/secure/RapidBoard.jspa?projectKey=HMADC` (visited on 04/04/2024).

[14] *Confluence main page.* May 2024. URL: https://confluence.iir.ntnu.no/display/HMADC/Health+Management+-+Anomaly+Detection+and+Classification (visited on 05/19/2024).

[15] Atlassian. "About Confluence". In: *Confluence basics* (2024). URL: https://www.atlassian.com/software/confluence/resources/guides/get-started/overview#about-confluence (visited on 05/03/2024).

[16] Oleksandr Mandryk. *The Agile: Scrum framework at a glance.* Feb. 2016. URL: https://icoderman.wordpress.com/2016/02/23/the-agile-scrum-framework-at-a-glance/ (visited on 04/15/2024).

[17] Claire Drumond. *What is scrum?* Apr. 2024. URL: https://www.atlassian.com/agile/scrum (visited on 04/17/2024).

[18] Max Rehkopf. *What are sprints in project management?* Apr. 2024. URL: https://www.atlassian.com/agile/scrum/sprints (visited on 04/17/2024).

[19] Dave West. *What is sprint planning?* Jan. 2019. URL: https://www.atlassian.com/agile/scrum/sprint-planning (visited on 05/19/2024).

[20] Ken Schwaber and Jeff Sutherland. *The Scrum Guide.* Nov. 2020. URL: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf (visited on 05/19/2024).

[21] Dave West. *What are the three scrum roles?* URL: https://www.atlassian.com/agile/scrum/roles (visited on 05/19/2024).

[22] Herlau, Tue and Schmidt, Mikkel N. and Mørup, Morten. *Introduction to Machine Learning and Data Mining.* Vol. 1. [Lecture notes, Spring 2022, Version for print]. Technical University of Denmark (DTU), Apr. 2022. ISBN: unknown.

[23] Sadhvi Anunaya. *Data Preprocessing in Data Mining: A Hands On Guide.* Feb. 2024. URL: https://www.analyticsvidhya.com/blog/2021/08/data-preprocessing-in-data-mining-a-hands-on-guide/ (visited on 05/03/2024).

[24] Binay Gupta. *Feature Scaling in Machine Learning.* May 2023. URL: https://www.scaler.com/topics/machine-learning/feature-scaling-in-machine-learning/ (visited on 04/18/2024).

[25] Apache. *RobustScaler.* URL: https://nightlies.apache.org/flink/flink-ml-docs-master/docs/operators/feature/robustscaler/ (visited on 04/18/2024).

[26] Medium. *From Raw to Rescaled: A Guide to Z-Score, Normalization, and Standardization in Data Preprocessing.* Aug. 2023. URL: https://medium.com/@anushruthikae/from-raw-to-rescaled-a-guide-to-z-score-normalization-and-standardization-in-data-preprocessing-173874df077d (visited on 04/18/2024).

[27] Medium. *Transformations, Scaling and Normalization.* July 2019. URL: https://medium.com/@isalindgren313/transformations-scaling-and-normalization-420b2be12300 (visited on 04/18/2024).

[28] George Lawton. *Definition Data Preprocessing.* Jan. 2022. URL: https://www.techtarget.com/searchdatamanagement/definition/data-preprocessing (visited on 05/02/2024).

[29] Geeks for Geeks. *What is Feature Engineering?* Dec. 2023. URL: https://www.geeksforgeeks.org/what-is-feature-engineering/ (visited on 04/19/2024).

[30] IBM. *What is machine learning (ML)?* URL: https://www.ibm.com/topics/machine-learning (visited on 04/18/2024).

[31] Foster, Kelsey. *Machine Learning Concepts for Beginners.* Feb. 2022. URL: https://www.assemblyai.com/blog/machine-learning-concepts/ (visited on 04/18/2024).

[32] Google Cloud. *Supervised vs. unsupervised learning: What's the difference?* URL: https://cloud.google.com/discover/supervised-vs-unsupervised-learning (visited on 04/17/2024).

[33] IBM. *What is a neural network?* 2024. URL: https://www.ibm.com/topics/neural-networks (visited on 05/12/2024).

[34] IBM. *What is a neural network?* URL: https://www.ibm.com/topics/neural-networks (visited on 05/03/2024).

[35] Hardesty. Larry and Massachutes Institute of Technology. *Explained: Neural networks.* Apr. 2017. URL: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414 (visited on 05/10/2024).

[36] Geeks For Geeks. *Introduction to Recurrent Neural Network.* Dec. 2023. URL: https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/ (visited on 05/12/2024).

[37] Medium. *Vanishing Gradient Problem in Deep Learning: Understanding, Intuition, and Solutions.* June 2023. URL: https://medium.com/@amanatulla1606/vanishing-gradient-problem-in-deep-learning-understanding-intuition-and-solutions-da90ef4ecb54 (visited on 05/12/2024).

[38] GeeksforGeeks. *Deep Learning | Introduction to Long Short Term Memory.* Dec. 2023. URL: https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/ (visited on 04/22/2024).

[39] Hesaraki, Saba. *Long Short-Term Memory (LSTM).* Oct. 2023. URL: https://medium.com/@saba99/long-short-term-memory-lstm-fffc5eaebfdc (visited on 04/20/2024).

[40] Geeks For Geeks. *What is the Difference Between val_loss and loss during training in Keras?* Feb. 2024. URL: https://www.geeksforgeeks.org/what-is-the-difference-between-val_loss-and-loss-during-training-in-keras/ (visited on 05/12/2024).

[41] Alake, Richmond. *Loss Functions in Machine Learning Explained.* Nov. 2023. URL: https://www.datacamp.com/tutorial/loss-function-in-machine-learning (visited on 04/18/2024).

[42] Vijay M. *What is the difference between Training Loss Validation Loss and Evaluation Loss.* Sept. 2023. URL: `https://medium.com/@penpencil.blr/what-is-the-difference-between-training-loss-validation-loss-and-evaluation-loss-c169ddeccd59` (visited on 05/02/2024).

[43] Wikipedia. *Anomaly detection.* Apr. 2024. URL: `https://en.wikipedia.org/wiki/Anomaly_detection` (visited on 05/04/2024).

[44] DataCamp. *Classification in Machine Learning: An Introduction.* Sept. 2022. URL: `https://www.datacamp.com/blog/classification-machine-learning` (visited on 04/23/2024).

[45] Khang Pham. *Overfitting, Generalization, the Bias-Variance Tradeoff.* May 2023. URL: `https://medium.com/@khang.pham.exxact/overfitting-generalization-the-bias-variance-tradeoff-5800f8c2200` (visited on 05/12/2024).

[46] Geeks For Geeks. *Bias and Variance in Machine Learning.* June 2023. URL: `https://www.geeksforgeeks.org/bias-vs-variance-in-machine-learning/` (visited on 05/16/2024).

[47] IBM. *What is data visualization?* URL: `https://www.ibm.com/topics/data-visualization` (visited on 05/16/2024).

[48] Encord. *Mean Square Error (MSE).* URL: `https://encord.com/glossary/mean-square-error-mse/` (visited on 05/16/2024).

[49] Ejiro Onose. *R Squared: Understanding the Coefficient of Determination.* Aug. 2023. URL: `https://arize.com/blog-course/r-squared-understanding-the-coefficient-of-determination/` (visited on 05/16/2024).

[50] Kongsberg. *C2533 Propulsion Engine.* [Internal document].

[51] Glemmestad, Even and Glemmestad Håkon. *Traktor og Motor.* Landbruksforlaget, Jan. 1977. ISBN: 82-529-0280-4.

[52] Martin Leduc. *The marine Diesel engine - Part two: The four stroke engine.* 2001. URL: `https://www.dieselduck.info/machine/01%20prime%20movers/diesel_engine/diesel_engine.02.htm` (visited on 04/23/2024).

[53] Wikipedia. *Engine Power.* Jan. 2024. URL: `https://en.wikipedia.org/wiki/Engine_power` (visited on 04/26/2024).

[54] Science Direct. *Lube Oil System.* July 2006. URL: `https://www.sciencedirect.com/topics/engineering/lube-oil-system` (visited on 05/02/2024).

[55] Anish. *General Overview of Central Cooling System on Ships.* 2019. URL: `https://www.marineinsight.com/guidelines/general-overview-of-central-cooling-system-on-ships/` (visited on 05/02/2024).

[56] Wikipedia. *Platform supply vessel.* Apr. 2024. URL: `https://en.wikipedia.org/wiki/Platform_supply_vessel` (visited on 04/23/2024).

[57] Kongsberg-Maritime. *UT 7400 - Platform supply vessel.* URL: `https://www.kongsberg.com/contentassets/4a443bb2cfbf444f9c82d652d1d2981b/8102-ut-design.jpg` (visited on 04/19/2024).

[58] ScienceDirect. *Interpreted Language.* 2024. URL: `https://www.sciencedirect.com/topics/computer-science/interpreted-language` (visited on 05/08/2024).

[59] Jupyter. *Jupyter*. 2024. URL: https://jupyter.org (visited on 05/08/2024).

[60] Wikipedia. *pandas (software)*. Apr. 2024. URL: https://en.wikipedia.org/wiki/Pandas_(software) (visited on 05/09/2024).

[61] NumPy. *About Us*. 2024. URL: https://numpy.org/about/ (visited on 05/08/2024).

[62] NumPy. *NumPy*. 2024. URL: https://numpy.org (visited on 05/08/2024).

[63] TensorFlow. *An end-to-end platform for machine learning*. 2024. URL: https://www.tensorflow.org (visited on 04/19/2024).

[64] Keras. *Keras: Simple. Flexible. Powerful.* 2024. URL: https://keras.io (visited on 04/19/2024).

[65] Scikit-learn. *Scikit-learn: Machine Learning in Python*. 2024. URL: https://scikit-learn.org/stable/ (visited on 04/19/2024).

[66] Swasti Dharmadhikari. *Compound annual growth rate*. Apr. 2024. URL: https://en.wikipedia.org/wiki/%20Compound_annual_growth_rate (visited on 05/13/2024).

[67] NTNU. *Idun*. Apr. 2024. URL: https://www.hpc.ntnu.no/idun/ (visited on 05/15/2024).

[68] Jeff Beckman. *Top Scrum Usage Statistics*. Aug. 2023. URL: https://techreport.com/statistics/scrum-usage-statistics/ (visited on 05/13/2024).

[69] Figma. *Figma - Homepage*. URL: https://www.figma.com (visited on 05/16/2024).

[70] Institute of Industrial & Systems Engineers. *Engineering Code of Ethics*. URL: https://www.iise.org/details.aspx?id=299 (visited on 05/13/2024).

[71] United Nations. *The 17 Goals*. URL: https://sdgs.un.org/goals (visited on 05/13/2024).

[72] United Nations. *Sustainable Development Goals*. URL: https://www.un.org/sustainabledevelopment/news/communications-material/ (visited on 05/13/2024).

[73] Sky Quest. *Global Predictive Maintenance Market Size, Share, Growth Analysis, By Components(Solution and Services), By Deployment Mode(On-Premises and Cloud) - Industry Forecast 2024-2031*. Mar. 2024. URL: https://www.skyquestt.com/report/predictive-maintenance-market (visited on 05/13/2024).

[74] Statista. *Size of the predictive maintenance market worldwide in 2020 and 2021, with a forecast for 2022 to 2030*. Jan. 2022. URL: https://www.statista.com/statistics/748080/global-predictive-maintenance-market-size/ (visited on 05/13/2024).

[75] Swasti Dharmadhikari. *Predictive Maintenance Market Report 2024 (Global Edition)*. Mar. 2024. URL: https://www.cognitivemarketresearch.com/predictive-maintenance-market-report (visited on 05/13/2024).

APPENDICES

A - PRELIMINARY PROJECT PLAN

Health Management – Anomaly Detection and Classification
Preliminary Project Plan

Version 1.0

Revision history

Date	Version	Description	Author
22/01/24	0.1	The initial draft of the preliminary project report.	Harald W Fredriksen, Even J. P. Haslerud, Jørgen Finsveen
24/01/24	0.2	Elaboration of problem statement, outcome measures, and execution	Harald W Fredriksen, Even J. P. Haslerud, Jørgen Finsveen
25/01/24	0.3	Inclusion of tables and figures, attaching appendix, improvement of the risk assessment.	Harald W Fredriksen, Even J. P. Haslerud, Jørgen Finsveen
26/01/24	1.0	Final revision and text enhancement.	Harald W Fredriksen, Even J. P. Haslerud, Jørgen Finsveen

Table of contents

1. Goals and frameworks

1.1 Orientation

Health Management is a bachelor project at the Norwegian University of Science and Technology (NTNU) presented and given to by Kongsberg Maritime. This project addresses a genuine issue faced by maritime vessels, a challenge that Kongsberg Maritime previously attempted to resolve in 2018. The project is focused on automation of vessel monitoring using machine learning algorithms.

Given the advancements in information and resources currently available, Kongsberg Maritime is making another attempt at this problem, seeking fresh perspectives to approach it. Each member of the bachelor group has an interest in machine learning. Consequently, when the group initiated contact with Kongsberg Maritime regarding a potential project for the bachelor's degree, the suggested case was found particularly compelling.

1.2 Problem statement / project description and objectives

The Health Management venture is intended as a series of preventative measures focusing on providing customers with operational stability. The objective is to monitor critical vessel machinery through collecting data logged from various sensors measuring different aspects of a given unit, and to analyse said data for detection and classification of anomalies as well as maintenance prediction. By utilizing condition monitoring, it is expected to help:

- Avoiding unplanned maintenance and keep critical assets operating.
- Predict maintenance needs which can be planned accordingly.
- Reducing the total cost of asset ownership.
- Maximizing the operational profitability.
- Ensuring sustainable maintenance and optimized machinery performance.

Kongsberg Maritime are already collecting and monitoring data from vessels at sea, however, they seek to find ways to automate the monitoring and classification of the data. In order to achieve this, Kongsberg would like to approach the problem as a subject to machine learning implementation. A satisfactory solution would be to implement machine learning techniques which makes an autonomous system capable of investigating machinery data in order to:

1. Detect anomalous long-term trends.
2. Classify detected anomalies.

The present Health Management service utilizes Failure Mode analysis of machinery data, which includes recognizing familiar failure patterns in operational data. Kongsberg Maritime has previously given a significant effort attempting to utilize machine learning for anomaly detection, but their approach of trying to detect instantaneous anomalies rather than long-term trends proved to be insufficient to the requirements of equipment condition monitoring. As of date, detected anomalies are investigated and diagnosed by product experts, even upon recurring anomalies in the same unit.

As there has been a vast number of technological advancements since the previous attempt to automate the Health Management service, Kongsberg Maritime seeks to approach the problem again with new insight and perspectives. The formulation of the goals and objectives of this project forms the basis for defining the following problem statement:

How can machine learning be implemented in order to detect and classify anomalous behaviour in machinery of various nature?

The problem statement describes the fundamental question which Kongsberg Maritime would like to find answers to, and which introduces the process of designing a machine learning model which can be considered to be an acceptable solution to the problem. The problem demands a concrete solution which can be applied to the most critical units subject to condition monitoring. Therefore, the following assumptions and delimitations are made:

- A trained machine learning model should be able to detect and classify long-term anomalies in engines, low- and high-speed rotating machines.
- Different pieces of machinery may require monitoring of different parameters describing important aspects of the condition of the machine.
- Anomaly classification must be based on an initial classification by a product expert.
- The machine learning model must have a sensitivity-level for anomaly detection which is appropriate for the significance of the given anomaly.

In order for a solution to be recognized as satisfactory for the problem, it should be capable of achieving the following results:

1. Detect anomalous behaviour in various machinery with a reasonable accuracy and sensitivity.
2. Correctly classify various different sorts of detected anomalies.
3. Prove to be more efficient than manually detecting and classifying potential anomalies from data.

By the end of the project, the developers aim to provide the client with a solution which fulfils these criteria, and which may exert an impact on the products and services of Kongsberg Maritime. The solution should provide new insight and value to the Health Management venture and enable the client to adopt a sophisticated condition-based maintenance strategy for their maritime services.

1.3 Outcome measures

In the course of this bachelor's project, the group aims to deepen their understanding of machine learning and gain proficiency in agile methodologies within the context of collaboration with a major corporation. Partnering with Kongsberg represents a strategic opportunity to acquire practical skills and valuable insight that will significantly enhance the group's professional competencies. The project is centred around a challenge previously addressed by Kongsberg; therefore, a successful resolution on the student's part could result in the development of an advanced model capable of seamless integration into Kongsberg's existing systems. Such an achievement would enable Kongsberg to enhance their offerings and deliver superior value to their clientele.

Furthermore, Kongsberg Maritime has defined a model known as the "Data Value Chain" which describes the process of processing data to gather insight which can produce value to the company.

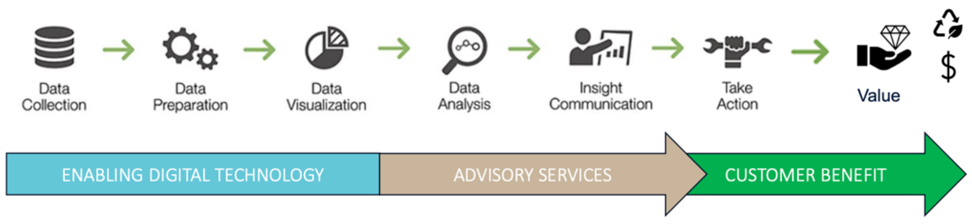


Figure A1 – Data Value Chain

(ref. David Vågnes from Kongsberg Maritime [Unpublished PowerPoint slides])

As seen in Figure A1, Kongsberg defines the three sub-chains marked with different colouring in the data value chain as subject to three different services. From left to right, these services are:

- **Enabling digital technology**: Health Management
- **Advisory services**: Marine & DP Operations
- **Customer benefit**: Eco Insight

Of these sub-chains, the first three links regarding data collection, preparation, and visualization are of most relevance for the health management service and therefore for this project. However, data analysis is a crucial part of the machine learning process and will therefore also be of relevance to the project.

It is believed that a satisfactory solution to the project’s problem would provide the necessary resources to enhance the insight communication link and the following links in the data value chain as well as streamlining the health management and condition monitoring services which Kongsberg provides for their customers.

1.4 Boundaries

In the development of the machine learning algorithm, the needs are primarily software, eliminating the necessity for physical materials. This aspect significantly enhances the cost-effectiveness of the project. Presently, the group do not anticipate the need of greater investment in additional software solutions to facilitate the completion of the task.

The primary constraint is time. Given the complexity of the problem at hand, the group must allocate substantial time to research and planning prior to development of the solution. Furthermore, the processing of extensive datasets in machine learning is time consuming. Therefore, strategic planning is imperative for the effective utilization of the time resources in model experimentation and refinement.

A potential need which may be encountered during the execution of the project is a means of storing data which will be used to train the machine learning models. The data may be extensive, which may require the group to store it externally in a cloud solution or a database. This could be relevant if the data is of sensitive nature, or if it is so extensive that it would be difficult to store it locally.

2. Structuring

The project is being conducted under the auspices of a collaborative effort between Kongsberg Maritime and the Norwegian University of Science and Technology (NTNU). Kongsberg Maritime, a renowned industry leader in maritime technology, functions as the primary client for the project, providing the projects problem and corresponding datasets.

NTNU, among the largest of the Norwegian universities, is the secondary client and the educational facilitator of this project. This project will serve as a demonstration of what the team has learned so far from their bachelor's programme in computer engineering.

Group 9

Name	Affiliation	Role
David Vågenes	Kongsberg Maritime	Client
Saleh Abdel-Afou Alaliyat	NTNU	Supervisor

Student team developer	Roles/Responsibilities
Even Johan Pereira Haslerud	Team leader
Harald Wangsvik Fredriksen	Project coordinator, Quality assurance
Jørgen Finsveen	Lead developer, Scrum master

3. Execution

3.1. Main activities

Setup

During the initial week, the group conducted introductory meetings with representatives from Kongsberg and the supervisory team. These sessions provided a valuable opportunity for all project participants to meet face-to-face, fostering personal connections and facilitating a discussion of the project's objectives and requirements.

In preparation of the forthcoming stages, the group dedicated some time for establishing and familiarizing themselves with various systems essential for effective collaboration and documentation. The group prepared environments such as Confluence for project management and documentation, Jira for issue and project tracking, Discord for real-time communication and GitHub for source code management, preparing the team for the development phase.

Documentation

Documentation is a crucial part of the project deliverable, describing the terms, theory, and the overall execution of the project. As there is continuously a series of events which is subject to documentation, the works of writing this will be an activity which runs in parallel throughout the project's lifetime. Several aspects of the project could also be prone to be updated dynamically, and such will therefore be documented in living documents.

The responsibility for clear and sufficient documentation lies on all members of the group. If necessary, a group member may be delegated the overall responsibility of project documentation, though neither of the other group members will be freed of responsibility in such an event.

Group 9

The process of writing high quality documentation requires a structured way of working. In order to ensure this, the group intends to invest time in creating an organized work environment both physically and digitally, applying agile development strategies such as Scrum and Lean scheduling principles such as Kanban. Combining the above principles with regular meetings with the client and the supervisor as well as logging produced information and protocols using the online software tools Jira and Confluence will facilitate for an organized project execution.

Research and planning

As the project group considers to be flat organization where each member equally wants to learn and contribute, the research needed for the project will be a mutual responsibility. In order for the group to be able to work in harmony while working on a solution to the project, it is important that all members have a general understanding of the subjects. The amount of quality research may prove to be a crucial factor to whether the project culminates in an acceptable solution to the project problem or not. Planning will naturally be of just as much importance.

Research and planning will be conducted when the need arises, and in the planned time slot between the middle of January and the middle of February. As the group intends to adopt the Scrum framework to this project, planning of which issues from the product backlog are to be included in the upcoming sprints will also take place regularly throughout the project execution.

Development

The development will start early in February and continue towards the start of May. Although every group member will actively contribute to the development, it may be necessary to delegate a group member with more responsibility for report writing, documentation, etc., as several of these activities coincide during this phase of the project. In such a scenario, it may be applicable to utilize role rotation.

In the event that a group member falls ill or is somehow unable to contribute to the development, will the group conduct an extraordinary meeting where the alternatives for adjustment will be discussed.

Finalization

As the project approaches completion, it may be wise to divide the group into smaller units, where different units have different jurisdictions such as testing, documentation, refactoring, report writing, and other activities. This will ensure that no activity is overlooked, and that the group is more flexible in this time-sensitive phase.

Project report

The task of compiling the project report is an ongoing task that spans the entire duration of the project. This approach is adopted to guarantee a comprehensive and detailed report while preventing overwhelming volume of work as the project nears completion. The report is a collaborative effort among all team members, to ensure that all aspects and insights are included. This collaborative process will be facilitated through either joint authorship or through a systematic and continuous peer review mechanism, ensuring that the report reflects the collective expertise and contributions of the entire team.

3.2. Milestones

Week	Project steps	Date
4	Preliminary Project Plan submission	26.01.2024
-	Oral Presentation	April (To be determined)
21	Bachelor Project submission/presentation	21.05.2024

4. Follow-up and quality assurance

4.1 Quality assurance

To ensure that the work conducted during the project meets the expected degree of quality, the group will implement quality assurance strategies, including:

- Regular peer reviews of code and execution.
- Consultation with Kongsberg Maritime’s experts to validate approaches and findings.
- Implementation of comprehensive testing protocols to ensure the reliability and accuracy of the machine learning models.

The practice of peer code review among team members significantly contributes to maintaining the highest standards of code quality and ensures that all developers remain abreast of the latest developments within the application.

Furthermore, the team are supported by an exceptional partner. The close collaboration with Kongsberg affords the team with the opportunity to consult with their experts. This collaboration facilitates the validation of the team's methodologies and enables them to receive insightful recommendations as necessary, thereby enhancing the overall quality and efficacy of the project.

To safeguard against bugs and uphold the reliability and accuracy of the machine learning models, the team will institute a thorough testing framework. This framework will encompass a variety of testing methodologies tailored specifically to the unique challenges presented by machine learning systems.

4.2 Reporting

The group will report regularly to both Kongsberg Maritime and the supervisor at NTNU. These reports will include progress updates, challenges encountered, and preliminary results. Reporting will be bi-weekly with additional updates as needed. These activities will also be a means of receiving feedback and advise from both the client and the supervisor.

In addition to reporting to Kongsberg Maritime and the supervisor, there will be reporting in the developer team internally. The group members will be committed to report to each other regarding current status in their activities. Such communication will be conducted on a daily basis during stand-up meetings in accordance with the Scrum framework.

5. Risk assessment

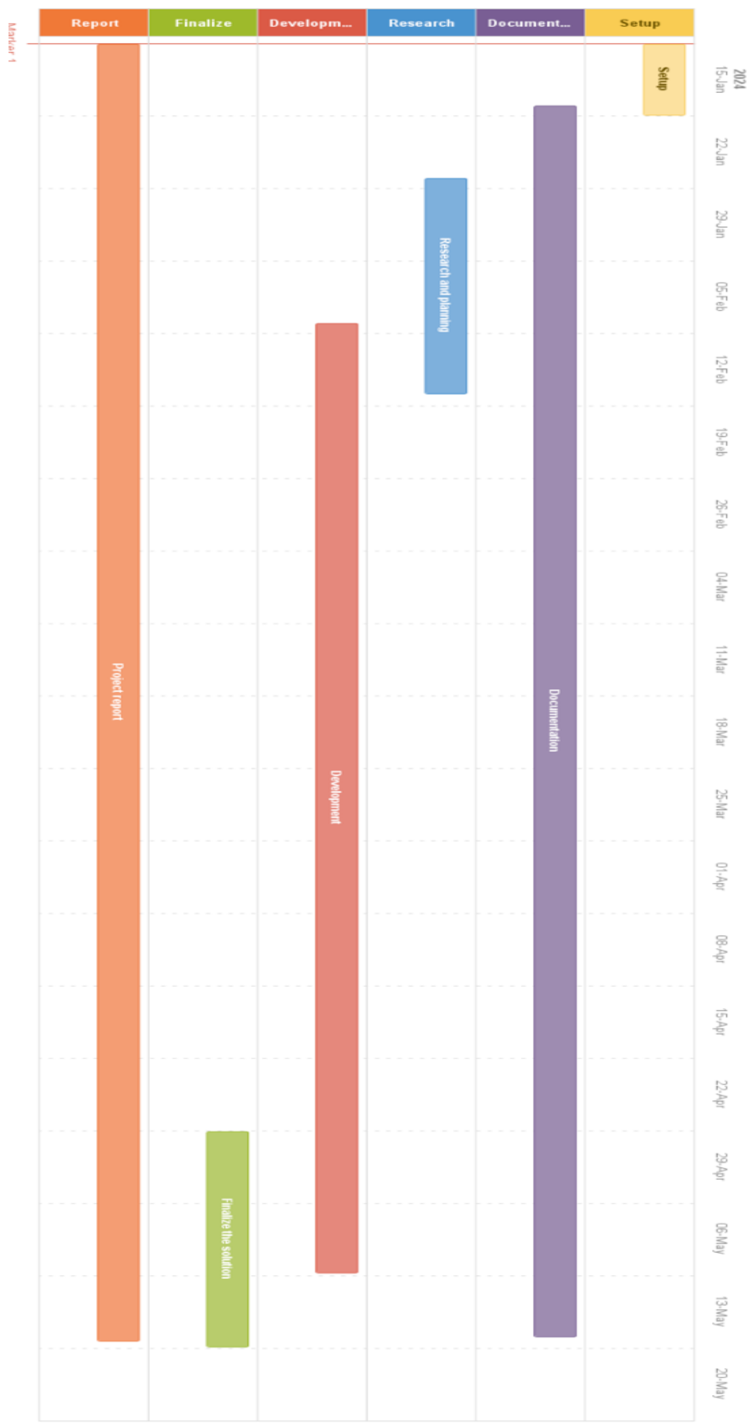
Risks will be continuously assessed throughout the project, considering factors such as data quality, model accuracy, group communication, and project timelines. Mitigation strategies will be developed for identified risks, and contingency plans will be in place.

Vulnerability	**Probability** *(1-5)*	**Consequence(s)**	**Preventive measure(s)**
Insufficient team coordination and communication.	2	Reduced efficiency, Reduced quality, Team conflicts, Bad use of resources	Conduct daily stand-up meetings following Scrum, Regular meetings with supervisor and client, Peer reviews, Long-term planning, Thorough planning, Sprint retrospectives
Resource bottle-necks and constraints.	3	Delayed progress, Reduced quality	Regular communication with supervisor and client, Thorough resource assessment
Unsuspected group member incapacity. (e.g. illness or injury)	1	Uneven distribution of work, Work overload, Reduced efficiency	Back-up solutions, Planning with margins, Adjustment of tasks
Tunnel vision of pending activities.	2	Reduced quality, Bad use of resources	Sprint retrospectives, Role rotation, Sprint planning, Status reports
Missing/forgetting deadlines.	3	Reduced quality, Penalty from client or NTNU	Thorough planning, Reminders, Time scheduling
Insufficient or clustered data.	4	Reduced ML model accuracy, Delayed progress	Extensive data preparation, Regular communication with client, Statistical adjustments
Inaccurate or biased ML model.	5	Reduced quality, Delayed progress, Project objective failure	Thorough research, Model experimentation, Communication with supervisor, Assessment of algorithms, Ensure high quality data, Extensive testing, Feature engineering
Work overload	3	Reduced efficiency, Reduced quality, Delayed progress	Extensive planning with margins, Clarification of expectations, Regular communication, Adjustment of tasks

6. Appendix

6.1 Schedule

Roadmap from the project's designated wiki-page on Confluence is attached. Note that this roadmap is an initial estimate and may be subject to change after future revisions.



6.2 Contact list

Name	Company	Phone number	Email	Address
Harald Wangsvik Fredriksen	NTNU	+47 941 33 252	haraldwf@stud.ntnu.no haraldwangsvik@gmail.com	Fjellgata 5, 6003 Ålesund
Even Johan Pereira Haslerud	NTNU	+47 944 82 572	ejhasler@stud.ntnu.no evenjohanhaslerud@gmail.com	Keiser Wilhelms Gate 44 6003 Ålesund
Jørgen Finsveen	NTNU	+47 917 04 251	jorgen.finsveen@ntnu.no joergen.finsveen@gmail.com	Vågavegen 29, 401 6008 Ålesund
Saleh Abdel-Afou Alaliyat	NTNU	+47 701 61 530	Alaliyat.a.saleh@ntnu.no	Ankeret B323, NTNU Ålesund
David Vågnes	Kongsberg Maritime	+47 412 35 171	david.vagnes@km.kongsberg.com	Borgundvegen 340, 6009 Ålesund

6.3 Contract documents

6.3.1 Cooperation contract for the bachelor group

Cooperation contract is attached.

6.3.2 Standard agreement

Standard agreement is attached.

B - PROJECT HANDBOOK

(See attached document in Inspera Assessment.)

C - AI DECLARATION FORM

NTNU | Fakultet for informasjonsteknologi og elektroteknikk

Deklarasjon om KI-hjelpemidler

Har det i utarbeidingen av denne rapporten blitt anvendt KI-baserte hjelpemidler?

☐ Nei

☒ Ja

Hvis *ja*: spesifiser type av verktøy og bruksområde under.

Tekst

☐ **Stavekontroll.** Er deler av teksten kontrollert av: *Grammarly, Ginger, Grammarbot, LanguageTool, ProWritingAid, Sapling, Trinka.ai* eller lignende verktøy?

☐ **Tekstgenerering.** Er deler av teksten generert av: *ChatGPT, GrammarlyGO, Copy.AI, WordAi, WriteSonic, Jasper, Simplified, Rytr* eller lignende verktøy?

☒ **Skriveassistanse.** Er en eller flere av ideene eller fremgangsmåtene i oppgaven foreslått av: *ChatGPT, Google Bard, Bing chat, YouChat* eller lignende verktøy?

Hvis *ja* til anvendelse av et tekstverktøy - spesifiser bruken her:

Vi har brukt ChatGPT til å finskrive noe tekst i rapporten. Vi har skrevet tekst og innhold selv, og deretter fått ChatGPT til å se over teksten og foreslå en mer akademisk og formell versjon. Så har vi brukt det til inspirasjon og skrevet teksten på nytt. Vi har ikke brukt ChatGPT til å generere helt ny tekst, kun til å skrive om tekst vi allerede hadde skrevet. ChatGPT har derimot blitt brukt til å komme med forslag til navn på delkapitler.

Kode og algoritmer

☒ **Programmeringsassistanse.** Er deler av koden/algoritmene som i) fremtrer direkte i rapporten eller ii) har blitt anvendt for produksjon av resultater slik som figurer, tabeller eller tallverdier blitt generert av: *GitHub Copilot, CodeGPT, Google Codey/Studio Bot, Replit Ghostwriter, Amazon CodeWhisperer, GPT Engineer, ChatGPT, Google Bard* eller lignende verktøy?

Hvis *ja* til anvendelse av et programmeringsverktøy - spesifiser bruken her:

ChatGPT har blitt til noen spesifikke oppgaver som f.eks. å transponere matriser, eksportere objekter, og fylle NaN-verdier i datasettet med gjenomsnittsverdien for gjeldende kolonne. Slike ting har blitt markert i kildekoden. Vi har ikke brukt ChatGPT til å implementere selve modellene. Utility-scriptet som vi har laget består også av noe kode som ChatGPT har laget for oss, men dette er da markert i filen. En KI-funksjon innebygd i Google Colab har blitt brukt til å plotte grafer underveis, men dette ble ikke tatt i bruk i den endelige versjonen av kildekoden.

Bilder og figurer

☐ **Bildegenerering.** Er ett eller flere av bildene/figurene i rapporten blitt generert av: *Midjourney, Jasper, WriteSonic, Stability AI, Dall-E* eller lignende verktøy?

Hvis *ja* til anvendelse av et bildeverktøy - spesifiser bruken her:

☐ **Andre KI verktøy.** har andre typer av verktøy blitt anvendt? Hvis ja spesifiser bruken her:

[X] Jeg er kjent med NTNUs regelverk: *Det er ikke tillatt å generere besvarelse ved hjelp av kunstig intelligens og levere den helt eller delvis som egen besvarelse.* Jeg har derfor redegjort for all anvendelse av kunstig intelligens enten i) direkte i rapporten eller ii) i dette skjemaet.

Jorgen Finsveen [illegible] Even J. P. Haslerud

19.05.2024 / Ålesund

Underskrift/Dato/Sted

D - SOURCE CODE

All code, tools, models, and results are included in the GitHub repository linked below. The repository also contains some documents including a brief description of the dataset attributes, cooperation contract, and standard agreement.

GitHub repository link

- `https://github.com/Bachelor-9/anomaly-detection`

(Compressed zip-file also attached in Inspera Assessment.)

E - JIRA & CONFLUENCE WIKI

Atlassian Jira

- `https://jira.iir.ntnu.no/projects/HMADC`

Atlassian Confluence

- `https://confluence.iir.ntnu.no/display/HMADC`

F - UTILITY SCRIPT

The utility script has been used in almost all Jupyter Notebooks written for this project. It contains functions which frequently were used for dataset operations. It was stored in Google Drive and imported into the notebooks as follows:

```python
from google.colab import drive
drive.mount('/content/drive')

import drive.MyDrive.scripts.utils as util

df = util.open_dataset_at_month(FILE)
df, x, y, controlled_parameters = util.produce_modified_dataset(df
    , x_attr, y_attr)
df = util.remove_rows_at(df, 'df["87XS"] >= 10')
```

Listing 1: Importing and using the utility script

Below is the utility script:

```python
import os
import warnings
import numpy as np
import pandas as pd

warnings.filterwarnings('ignore')


"""
UTILITY SCRIPT
-------------------------------------

    authors:
    - @joergen-finsveen
    - @harald-wangsvik-fredriksen
    - @even-johan-pereira-haslerud
    version:
    - 2024-03-18

    This script contains utility functions used for various
   dataset operations. When working with the dataset and making
   different models, we noticed that we had to recycle some
   functions for every script we made. In order to reduce
   duplication and in order to make our scripts simpler and easier
    to read, we extracted the dubplicated code into this script.

    Notice that this script is not intended to be used for other datasets. It is specifically designed to work with datasets from Kongsberg Maritime stored in a .parquet file format. In order for the open_dataset_at_date() and open_dataset_at_month() functions to work, the dataset must be ordered in a certain directory structure. The applicable structure is as follows:

    ~/
     |---[ship].parquet/
     |       |---year=[year]/
     |            |---month=[month]/
     |                 |---day=[day].parquet
     |
     |---[ship].parquet/

"""



def open_dataset_at_date(path: str, f: dict) -> pd.DataFrame:
    """Opens a dataset file from a given ship and date and returns a Pandas DataFrame with time as index.

    Parameters
    ----------
        path: Relative path to the dataset directory.
        >>> path = 'drive/MyDrive/dataset'

        f: A dictionary containing the vessel ID and the date of the dataset to open. The dict should be on the following format:
        >>> f = {
                'ship': '13218',
                'year': '2023',
                'month': '9',
                'day': '4'
            }

    Returns
    ----------
        >>> pandas.DataFrame

    """
    path = f'{path}/{f["ship"]}.parquet/year={f["year"]}/month={f["month"]}/day={f["day"]}.parquet'

    df = pd.read_parquet(path)
    df = df.sort_values(by = 'time')
    df['time'] = df['time'].dt.strftime('%Y-%m-%d %H:%M:%S')
    df = df.set_index(pd.DatetimeIndex(df['time']))
    df = df.drop('time', axis=1)
    df = df.drop('timestamp', axis=1)

    return df



def open_dataset_at_month(path: str, f: dict) -> pd.DataFrame:
```

```python
    """Opens a dataset file from a given ship at a given year and month and returns a Pandas DataFrame with time as index.

    Parameters
    ----------
        path: Relative path to the dataset directory.
        >>> path = 'drive/MyDrive/dataset'

        f: A dictionary containing the vessel ID and the year and month of the dataset to open. The dict should be on the following format:
        >>> f = {
                'ship': '13218',
                'year': '2023',
                'month': '9'
            }

    Returns
    ----------
        >>> pandas.DataFrame

    """
    path = f'{path}/{f["ship"]}.parquet/year={f["year"]}/month={f["month"]}/'

    all_files = [os.path.join(path, file) for file in os.listdir(path) if file.endswith('.parquet')]

    df = pd.DataFrame()

    for file in all_files:
        daily_data = pd.read_parquet(file)
        df = pd.concat([df, daily_data])

    df = df.sort_values(by = 'time')
    df['time'] = df['time'].dt.strftime('%Y-%m-%d %H:%M:%S')
    df = df.set_index(pd.DatetimeIndex(df['time']))
    df = df.drop('time', axis=1)
    df = df.drop('timestamp', axis=1)

    return df



def produce_modified_dataset(df: pd.DataFrame, x_attr: list, y_attr: list) -> tuple:
    """Produces a modified version of the dataset where:
    * Attributes with constant values are removed.
    * Attributes containing non-numeric values are removed.
    * Attributes attributes not present in neither ```x_attr``` and ```y_attr``` are removed.

    Parameters
    ----------
        df: The ```pandas.DataFrame```to modify.
        x_attr: List of x attributes to preserve if present in the dataset and not constant.
        y_attr: List of y attributes to preserve if present in the
```

~~~python
 dataset and not constant.
        controlled_parameters: List of all attributes which exists in the dataset and also is numerical.

    Returns
    ----------
        Tuple with the modified ```pandas.DataFrame``` and the x and cylinder lists with constant attributes removed.
        >>> (pd.DataFrame, x_attr, y_attr)

    """
    for column in df.columns:
        if len(set(df[column].values)) == 1:
            if any(element == column for element in x_attr):
                x_attr.remove(column)
            if any(element == column for element in y_attr):
                y_attr.remove(column)

    numeric_columns = df.select_dtypes(include=[np.number])
    controlled_parameters = list(set(x_attr) & set(numeric_columns.columns.tolist()))
    controlled_parameters.sort()
    columns = controlled_parameters + y_attr

    return df[columns], x_attr, y_attr, controlled_parameters



def remove_rows_at(df: pd.DataFrame, constraint: str) -> pd.DataFrame:
    """Remove rows in the dataset which meets a given condition.

    Parameters
    ----------
        df: The ```pandas.DataFrame``` object.

        constraint: String containing a logical expression as e.g.:
        >>> constraint = "df['21EL'] > 250"

    Returns
    ----------
        >>> pandas.DataFrame

    """
    return df[eval(constraint)]
~~~

Listing 2: Utility script used for dataset operations

NTNU
Norwegian University of
Science and Technology