PREDICTIVE EQUIPMENT MAINTENANCE FOR HYDRAULIC SYSTEMS USING MACHINE LEARNING

AYAT ABDULAZIZ GABER AL-KHULAQI
1191202335

SESSION 2023/2024

FACULTY OF COMPUTING AND INFORMATICS
MULTIMEDIA UNIVERSITY
JULY 2024

ACKNOWLEDGEMENT

I express my sincere gratitude to Allah for giving me the opportunity and ability to complete this Final Year Project Phase 1. I am also deeply grateful to Dr Palanichamy Naveen, a lecturer at the Multimedia University's Faculty of Computing and Informatics, who supervised my project and guided me throughout the process with her valuable suggestions and feedback. Furthermore, I thank my family and friends for their continuous support and encouragement. They always motivated me and helped me overcome my challenges and difficulties.

Abstract

Hydraulic systems are widely used in various industries. Still, they are prone to faults and failures that can adversely impact their performance and safety. Predictive maintenance is an intelligent and proactive method that employs data and analytics to monitor hydraulic components and systems, allowing for the anticipation and prevention of potential failures before they occur. Machine learning has become increasingly important in predicting machine failures and supporting predictive maintenance in recent years. Prior studies have applied diverse techniques to monitor and diagnose the condition of hydraulic systems, but they have yet to examine the association between condition variables. This research paper presents a predictive model for hydraulic system maintenance utilising machine learning and deep learning approaches. Furthermore, the paper aims to identify the correlation between the conditions in the hydraulic system. The paper applies different data preparation techniques, such as feature scaling, extraction, and selection, to enhance the model's performance and reduce dimensionality. The study uses three classifier models: random forest, catboost, and long short-term memory, and compares their outcomes using various metrics such as Precision, Recall, F1-Score, and Accuracy. This research contributes to the advancement of predictive maintenance modelling and the maintenance of hydraulic systems by utilising machine learning techniques and finding the correlation of the condition variables. The results demonstrate that the Catboost model performed the best overall, achieving perfect scores across multiple components.

Table of Contents

List of Figures

List of Tables

List of Equations

Chapter 1: Introduction

Chapter 1, the ***'Introduction,'*** dives into the research's background, problem statement, aim, research questions, objectives, scope, and significance. This chapter aims to clarify the research's key aspects, which will be explored further in subsequent chapters.

1.1 Background

Hydraulic systems are widely prevalent in various industries, including petroleum and aerospace. Maintenance of hydraulic equipment is a critical aspect of these industries, and condition monitoring plays an essential role in predictive maintenance. The use of sensors for identifying and addressing machinery faults before they lead to failure is crucial. Effective condition monitoring is vital for predictive maintenance, enabling timely and preventive repairs and predicting potential equipment failures.

Based on the book "Basics of Hydraulic Systems" by Q. (2019), hydraulic systems work with liquid, allowing them to handle very high pressure without much change in size. This enables hydraulic power systems to deliver much power using a small amount of liquid.

This highlights the efficiency and power of hydraulic systems and their sensitivity and complexity, necessitating careful monitoring and maintenance. As suggested by (Quatrini et al., 2020), accurate condition-based or predictive maintenance strategies can significantly aid plant safety management. Condition monitoring, which can reduce machine downtime and maintenance costs, is the first step towards proper condition-based or predictive maintenance. As mentioned in the book Complete Guide to Preventive and Predictive Maintenance (Levitt, 2003), predictive maintenance cannot directly extend the lifespan of equipment. However, it can prevent unexpected failures, reduce downtime, and improve reliability, thus practically utilising almost the entire lifespan of equipment. Hence, condition monitoring and predictive maintenance are crucial for ensuring the optimal performance and safety of hydraulic systems.

Hydraulic systems are subject to various types of faults and degradation, affecting their performance and reliability. Some faults in hydraulic systems are cooler degradation, valve malfunction, internal pump leakage, and hydraulic accumulator failure. These faults can have serious consequences, such as reduced efficiency, increased downtime, and safety hazards. Therefore, it is crucial to monitor the condition of hydraulic equipment using sensors and data analysis and perform predictive maintenance to prevent or mitigate faults before they lead to failure.

Predictive maintenance is a proactive maintenance strategy that uses scientific methods to monitor and diagnose the condition of equipment and estimate its remaining useful life. Predictive maintenance can help optimise the performance and efficiency of hydraulic equipment, reduce the risk of failure and downtime, and save costs and resources. Several research studies have been conducted on predictive maintenance modelling for hydraulic systems, using various data-driven approaches based on machine learning and deep learning techniques. However, these studies have yet to explore the correlation between the different conditions in the hydraulic system, which can provide valuable insights for fault diagnosis and prognosis. In addition, most existing studies have utilised machine learning algorithms, such as logistic regression, support vector machine, and random forest, with deep learning techniques largely disregarded. Deep learning techniques, such as convolutional and recurrent neural networks, can offer advantages over machine learning techniques, such as higher accuracy, robustness, and scalability.

The current data-driven methods for hydraulic system fault prediction need to be revised to address these new challenges. They need to improve on several limitations, such as ineffective feature extraction, improper data correlation, suboptimal classifier training, and unreliable or inaccurate predictions. A predictive model that uses sensor data and cycle features is needed to forecast the maintenance requirements of hydraulic equipment. This model should be able to extract and select representative features from the raw sensor data, correlate the features with the condition characteristics, and train and test a suitable classifier model. This model can help data scientists and maintenance engineers enhance the understanding and prediction of hydraulic system faults and improve the condition-based maintenance of hydraulic equipment.

1.1.1 Problem Statement

There is an increasing demand for hydraulic systems in various industrial applications, such as petroleum and aerospace. However, this equipment is susceptible to different types of faults that can cause severe damage and accidents if not detected and diagnosed in time. Therefore, condition monitoring and predictive maintenance of hydraulic equipment using sensors and data analysis are essential. The current data-driven methods for hydraulic system fault prediction need to be revised to address these new challenges. Features are not extracted effectively, data are not appropriately correlated, classifiers need to be trained optimally, and predictions need to be more reliable and accurate. A predictive model that can forecast the maintenance requirements of hydraulic equipment and find out how sensors and condition variables change over time and how they are correlated to each other. This model can help reduce the risk of hydraulic system failure, improve the performance and efficiency of hydraulic equipment, and save costs and resources.

1.2 Aim

This research aims to construct a predictive model that forecasts the maintenance requirements of a hydraulic system, which involves identifying the patterns of decline in sensors and condition variables and the correlation between the condition variables. It is aimed at data scientists and maintenance engineers focused on predictive maintenance modelling and maintenance of the hydraulic system.

1.2.1 Research Questions

- What machine learning techniques can be used to predict the maintenance of hydraulic systems?
- How can decline patterns on sensors and condition variables be identified?
- Is there any correlation between the condition variables?

1.2.2 Research Objectives

- To perform a literature survey on techniques for correlating condition variables in Predictive Equipment Maintenance in the Hydraulics System.
- To design and implement machine learning models using condition variables in Predictive Equipment Maintenance.
- To evaluate the performance of the models using different metrics and identify the optimal model.

1.3 Project Scope

- This research focuses on data scientists and maintenance engineers interested in predictive maintenance modelling and the maintenance of hydraulic systems.
- The research methodology involves identifying decline patterns in sensors and condition variables and finding correlations between the dependent variables.
- The research aims to construct a predictive model that forecasts the maintenance requirements of a hydraulic system using machine learning techniques.

1.4 Implementation Plan

The chart indicates each task's progress and completion status for Final Year Project 2, as shown in *Figure 1.1*.

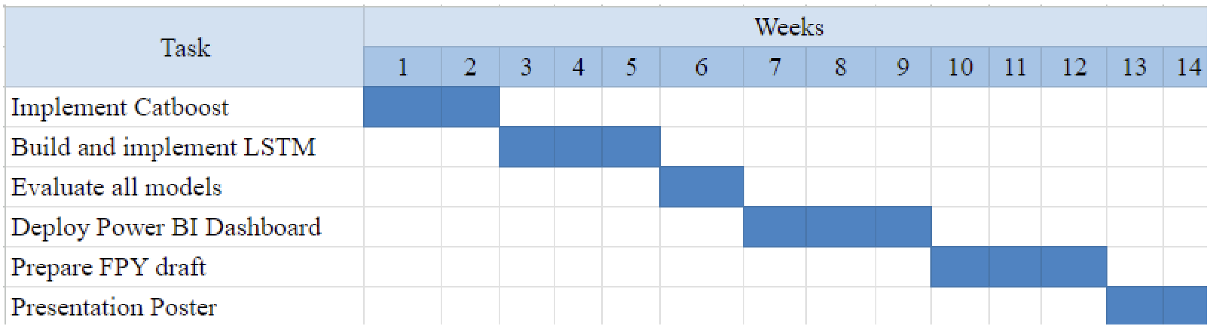


Figure 1.1 Final Year Project 2 Gantt Chart

1.5 Research Chapters

This research consists of five chapters. Chapter **1, *'Introduction,'*** introduces the research topic: building a predictive model for hydraulic system maintenance. It covers the Background, Problem Statement, Aim, Research Questions, Research Objectives, Scope and Area of Research, and Significance of Research. Chapter 1 gives an overview of the essential parts of the research, which will be discussed in more detail in the following chapters.

Chapter 2, ***'Background Research,'*** explains hydraulic systems, condition monitoring, and predictive maintenance. It also discusses the connection between condition monitoring and predictive maintenance. Finally, it includes an analysis of past studies, focusing on the feature engineering applied to the data and the classifier model used.

Chapter 3, ***'Theoretical Framework,'*** explores the research framework known as the Cross Industry Standard Process for Data Mining (CRISP-DM). The framework is divided into six parts, which are discussed in depth within the chapter.

Chapter 4, ***'Research Methodology,'*** describes the research project's data collection, understanding, preparation, modelling, evaluation, and deployment steps. It also explains the data sources, data features, data processing techniques, machine learning and deep learning algorithms, performance metrics, and software tools used in the project.

Chapter 5, ***'Implementation Plan,'*** presents the results and analysis of the project's data understanding and data preparation stages. It also discusses the challenges and limitations encountered in these stages and the solutions to overcome them.

Chapter 6, '***Testing***,' provides a comprehensive evaluation of the machine and deep learning models used in the project. This chapter covers the testing of these models, including hyperparameter tuning, to achieve optimal accuracy. The tuning focuses on two key hyperparameters: the number of estimators for machine learning models and batch size for deep learning models.

Chapter 7, '***Dashboard***,' provides a comprehensive overview of the visualization and analysis of the hydraulic system's condition using the dataset provided by ZeMA gGmbH. This chapter details the creation and interpretation of the dashboard using Power BI, which includes several key performance indicators and charts that monitor various components of the hydraulic system.

Chapter 8, ***'Conclusion,'*** summarises the main findings and contributions of the research project. It also discusses the implications and applications of the research outcomes for predictive maintenance and hydraulic systems. It also provides suggestions and recommendations for future work and improvement.

Chapter 2: Background Research

Chapter 2***, 'Background Research,'*** delves into more details about the hydraulic system, condition monitoring, predictive maintenance, how condition monitoring is related to predictive maintenance and previous research. Lastly, it discusses the study gap in the predictive maintenance of hydraulic systems.

2.1 Hydraulic Systems

According to the book "Basics of Hydraulic Systems" (Q., 2019), hydraulic systems use a liquid medium to transmit power. Using liquid in hydraulic systems enables them to withstand high-pressure levels while experiencing minimal alterations in volume. This characteristic renders hydraulic power systems proficient in transmitting substantial power quantities through a comparatively small volume of liquid. The prevalent application of hydraulic systems in facilitating power transmission for mobile equipment stems from their inherent capacity to carry and transmit significant amounts of energy. A standard hydraulic system's components include a hydraulic pump, a control valve, and a cylinder.

Hydraulics, according to the book "Hydraulic Fluid Power - A Historical Timeline" (Skinner, 2014), is a branch of science and engineering those studies and utilises the properties and applications of fluids under pressure. Throughout history, many inventors and mathematicians have contributed to developing and understanding hydraulics. Some of the most notable ones are Ctesibius, Leonardo Da Vinci, Simon Stevin, Blaise Pascal, and Joseph Bramah. Ctesibius invented the first water pump, water organ, and water clock using air pressure and displacement. Leonardo Da Vinci designed and sketched various hydraulic machines and devices, such as a water wheel, a canal lock, a submarine, and a hydraulic saw. The hydrostatic paradox, which explains the pressure exerted by a fluid at rest, was formulated by Simon Stevin. Blaise Pascal discovered and proved Pascal’s law, which describes pressure transmission in a confined fluid. He also invented the syringe and the hydraulic press based on this law. Joseph Bramah patented the first hydraulic press in 1795 and improved the design and efficiency of water pumps, water closets, and beer engines. Hydraulics has been applied to various fields and purposes throughout history, including improving water pumps, water closets, and beer engines by Joseph Bramah's patented hydraulic press in 1795.

Based on the relevant papers provided by ZeMA gGmbH, (Helwig et al., 2015) present the circuit diagram of a hydraulic test rig encompassing a primary working circuit (a), and a secondary cooling-filtration circuit (b) interconnected through an oil tank, as illustrated in Figure 2.1. This hydraulic system conditions test rig replicates diverse load levels and operational cycles for a hydraulic circuit featuring a primary pump MP1. A proportional pressure relief valve V11 regulates the load level, while a Programmable Logic Controller (PLC) is responsible for gathering and transmitting sensor data. Various process values, including pressure (PS1-6), flow (FS1-2), temperature (TS1-4), motor power (EPS1), vibration (VS1), particle contamination (CS, MCS & COPS), and oil parameters (COPS), are measured by the test rig using a variety of sensors equipped with both analogue and digital interfaces. The test rig can perform fixed and variable working cycles with predetermined or random load fluctuations to replicate scenarios found in industrial or mobile applications. The acquired sensor data is stored on a personal computer (PC) for subsequent analysis.

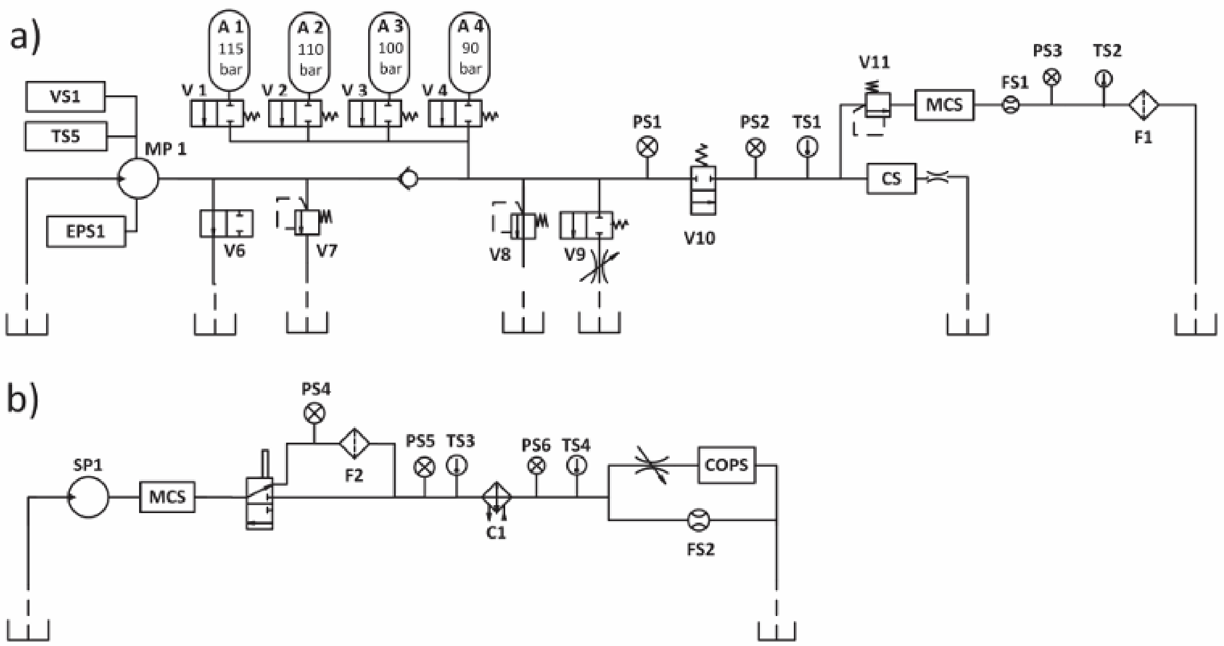


Figure 2.1 Circuit Diagram of a Hydraulic Test Rig (Nikolai Helwig, 2015)

2.2 Condition Monitoring

"Predictive Maintenance of Pumps Using Condition Monitoring" book by (Beebe, 2004) states that condition monitoring, whether online or offline, is a maintenance check. This process involves observing an operating asset and analysing the collected data. The aim is to identify signs of degradation, determine the reasons for any faults, and predict how long the asset can operate safely and cost-effectively.

The open-source dataset provided by ZeMa gGmbH (Schneider et al., 2018) forms the foundation for the condition monitoring of hydraulic systems. This procedure involves the assessment of a hydraulic test rig by utilising data collected from various sensors. The dataset, obtained through experimental means from the hydraulic test rig, encompasses information derived from a primary working circuit and a secondary cooling-filtration circuit interconnected through an oil tank. The system undergoes continuous load cycles, each enduring for 60 seconds. The sensors, including pressures, volume flows, and temperatures, are meticulously recorded during these cycles. The four hydraulic conditions—cooler, valve, pump, and accumulator—systematically vary quantitatively.

2.3 Predictive Maintenance

According to the book "Complete Guide to Preventive and Predictive Maintenance" (Levitt, 2003), predictive maintenance applies scientific methods to monitor and diagnose the condition of equipment and estimate its remaining useful life. Predictive maintenance does not directly extend the equipment's life but provides information that enables timely and effective corrective actions, such as repairs, replacements, or adjustments. These actions can prevent unexpected failures, reduce downtime, and improve reliability. Predictive maintenance can employ various techniques, depending on the equipment's type and failure mode. Some examples of predictive techniques are infrared scans, vibration analysis, oil analysis, ultrasonic testing, and thermography.

2.4 Previous Research

Table 2.1 reveals that the researchers utilised diverse fault analysis methodologies, ranging from predicting the failure of the entire hydraulic system to forecasting all or some types of condition failures or even different stages of each condition failure type. The following subsections will provide further details about the research papers.

Table 2.1: Methods Used in Prior Studies

No.	Authors	Hydraulic System Fault Analysis
1	M et al. (2022)	The whole hydraulic system.
2	Quatrini et al. 2020	Each condition of the hydraulic system.
3	Chekkala 2020	Each condition of the hydraulic system.
4	Guo et al. 2019	Four conditions of hydraulic system faults.
5	Keleko et al. 2023	Each condition has different hydraulic system failure stages.
6	Helwig et al. 2015	Four conditions of hydraulic system faults.
7	Kim et al. July 2022	The whole hydraulic system.
8	Chen et al. 2023	The whole hydraulic system.
9	Peng et al. 2020	The whole hydraulic system.
10	Zhang et al. 2022	Four conditions with different stages of hydraulic system failure.
11	Gaurkar et al. 2021	Four conditions of hydraulic system faults.
12	Askari et al. 2023	The whole hydraulic system.
13	Berghout et al. 2021	Four conditions of hydraulic system faults.
14	Soh et al. 2021	Each condition of the hydraulic system.
15	Wu et al. 2020	Four conditions of hydraulic system faults.
16	Xu et al. 2020	Four conditions of hydraulic system faults
17	Liu et al. 2023	Four conditions of hydraulic system faults.
18	Khan et al. 2023	Cool Component of Hydraulic System.
19	Buabeng et al. 2023	The whole hydraulic system.
20	Kim et al. 2022	Four conditions of hydraulic system faults.
21	Ma et al. 2021	Each condition of the hydraulic system.
22	Mallak et al. 2021	The whole hydraulic system.

2.4.1 Whole Hydraulic System Analysis

Within this section, it has been observed that some researchers have adopted an approach aimed at predicting the failure of the hydraulic system. Specifically, three out of four studies have highlighted using data preparation methods before deploying classifier models, as evidenced in *Table 2.2*.

Table 2.2: Whole Hydraulic System Analysis Methodology

Journal	Data Preparation	Classifier Models
M et al. 2022	Not mentioned	LR, KNN, DT, RF, and NB.
Kim et al. 2022 July	Signal shape features, Distribution density, and Pearson Correlation Coefficient.	DT, ADA, Ridge, LDA, SVM, RF, NB, KNN, XGBoost, LR.
Chen et al. 2023	Time domain features and Borderline-SMOTE.	XGBoost.
Peng et al. 2020	Nearest Centroid (NC) with Dtw barycenter averaging (DBA), Data Normalisation (Min-Max).	Softmax, DT, Linear SVM, Gaussian SVM and RF.
Askari et al. 2023	Gain Information (GI), Main Failure Indicators (MFIs), and Data Normalisation.	SVM, SSL-SVM, LR, SSL-LR, DT, SSL-DT, NB, SSL-NB, RF, SSL-RF.
Mallak et al. 2021	Principal Component Analysis (PCA), feature importance (FI), Cluster-Based Feature Selection (RkSE), and Time domain features.	LR, LDA , KNN, CART, NB, SVM, RF, CNN, and LSTM.

The investigation conducted by (M et al., 2022) aimed to proffer a hydraulic system integrated with machine learning algorithms to detect internal faults within the system. The proposed system encompasses dual hydraulic circuits, one dedicated to primary operations and the other for cooling and filtration. Real-time data from the hydraulic system are systematically gathered through sensors and subsequently stored in a distinct file. Implementing the machine learning algorithms and computation of parameters like accuracy, precision, and recall are facilitated through a Raspberry Pi controller and the Spyder IDE software toolchain. The study incorporates five classifier models: logistic regression, K-nearest neighbour, decision tree, random forest, and naive Bayes. Logistic regression exhibited the highest accuracy at 93%, while K-nearest neighbour achieved the highest precision at 87%, and naive Bayes recorded the highest recall at 98%.

In the investigation conducted by (Kim et al., July 2022), diverse feature engineering techniques were applied, encompassing signal shape features such as the slope of the linear fit, position of the maximum value, distribution density, and Pearson correlation coefficient. The research employed a range of machine-learning algorithms, including SVM, RF, NB, KNN, GB, and LR. Five models with notable accuracy, F1-score, and AUC were chosen as the foundational models for the ensemble method. The paper proposed a neural network-like structure comprising multiple layers of base models, where each layer utilised four out of five predictions from the preceding layer to generate a new dataset and train a subsequent model. The ultimate prediction was executed by a meta-model leveraging the predictions from the final layer. The effectiveness of the proposed method was assessed using four metrics: accuracy, precision, recall, and F1-score. Comparative analysis was conducted against existing stacking methods, machine learning models, and boosting algorithms. The study revealed that the proposed Multi-Layer Stacking method surpassed existing approaches, achieving an accuracy of 97.3%, precision of 98.3%, recall of 94%, and an F1-score of 96%. Additionally, the optimal number of layers for the proposed method was determined to be three, with the introduction of additional layers not resulting in performance improvement or degradation.

The study conducted by (Chen et al., 2023) proposes a Time-based Imbalanced Data Synthesis Technique (TIDS) designed to predict faults in hydraulic systems. The TIDS methodology encompasses two primary stages: initially, the extraction of time-domain features from the time series data, including parameters such as mean, variance, peak value, skewness, maximum value, and minimum value; subsequently, the generation of minority class data employing these time features with the Borderline-SMOTE algorithm, which specifically targets borderline samples for synthesis, thereby enhancing class separation. The study used the Condition Monitoring of Hydraulic Systems Data Set sourced from the UCI Machine Learning Repository, incorporating feature extraction and data synthesis. Subsequently, the XGBoost classifier was employed for fault prediction. The assessment of the proposed method's efficacy involved precision, recall, and F1 score, and a comparative analysis was performed against analogous methodologies. The results for TIDS & XGBoost Classifier were as follows: precision 94.86%, recall 94.21%, and F1 score 94.42%.

In the study conducted by (Peng et al., 2020), the proposed method consisted of three main steps: data processing, feature extraction, and classification. Firstly, the data were normalised to minimise the effect of varying magnitudes. Secondly, the data were segregated into different categories based on fault types, and the average sequence of each category was computed using Dynamic Time Warping Barycenter Averaging (DBA) to reduce dimensionality and maintain original features. Next, the Nearest Centroid (NC) was employed to identify the most representative features for classification. Finally, the Random Forest (RF) classifier was utilised to classify the features. The proposed method was compared with four other classifiers: softmax, decision tree, linear SVM, and Gaussian SVM, and evaluated on a hydraulic system dataset from UCI consisting of 2205 samples with 17 features and seven fault types. The findings revealed that the proposed method achieved an accuracy of 95.11%, significantly superior to the other classifiers.

In their paper, (Askari et al., 2023) propose a novel semi-supervised learning (SSL) method for fault detection and diagnosis in pneumatic and hydraulic systems. Their method combines graph-based label propagation with various classification algorithms, such as LR, NB, DT, and SVM, to leverage labelled and unlabelled data. They apply their method to two case studies and demonstrate that it can achieve higher accuracy than baseline methods, such as supervised and unsupervised learning, with much less labelled data. They report that the highest accuracy was achieved by SSL-LR, NB, and SSL-NB (92.928%) compared to other methods. They also analyse the factors that affect the performance of their method, such as the quality and quantity of labelled data, the similarity metric, and the leading failure indicators.

Their research (Mallak et al., 2021) proposed a comprehensive fault detection and diagnosis (FDD) approach for hydraulic systems using a hydraulic test rig, which can handle sensor and condition faults. Their approach consisted of two phases: detection and diagnosis. The detection phase used a Long Short-Term Memory (LSTM) autoencoder to reconstruct the healthy signal from the sensor readings and measure the deviation from the input signal. The diagnosis phase used machine learning and deep learning classifiers to identify the type and nature of the faults detected by the autoencoder. They applied their approach to two experiments: one for sensor faults and one for condition faults, using different data pre-processing and feature engineering methods. They showed that their FDD approach achieved high accuracy and robustness in detecting and diagnosing other types of faults in hydraulic systems. They also compared their approach with the existing methods based on autoencoders in the literature and demonstrated its superiority. The best results were obtained using time-domain features with CART (99.51%) or LSTM (95.68%) classifiers for sensor faults and feature importance with an RF classifier for condition faults.

2.4.2 Analysis of Hydraulic Failures

In this section, it has been brought to light that certain researchers have adopted an approach aimed at predicting some or all of the condition failures of the hydraulic system. Through an analysis of *Table 2.3*, it has been discovered that three papers utilised a similar or identical method for data preparation techniques.

Table 2.3: Analysis of Hydraulic Failures Methodology

Journal	Data Preparation	Classifier Models
Guo et al. 2019	Distribution density and Pearson Correlation Coefficient.	LDA, ANN, Ensemble SVM.
Quatrini et al. 2020	Signal shape features, Distribution density, Pearson Correlation Coefficient, and Linear Discriminant Analysis.	LR, NN, DF, and SVM.

Journal	Data Preparation	Classifier Models
Chekkala 2020	Quantile Transform Scalar, One-Hot Encoding, Classification *Imbalance using Synthetic Minority Oversampling Technique* (SMOTE), Principal Component Analysis (PCA), t-distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP).	ANN, Catboost, LightGBM, LR, RF, and XGboost.
Helwig et al. 2015	Signal shape, Distribution density characteristics, and Pearson Correlation Coefficient.	LDA, ANN, SVM (linear) and SVM (RBF).
Gaurkar et al. 2021	Data Normalisation, Attribute Selection, Attribute Generation, and Discretization.	LSTM, and RF.
Berghout et al. 2021	Time Domain, Data Normalisation, Spearman's correlation analysis, and Compressed Sensing (CS)	Auto-NAHL, LSTM, 1D-CNN, and DBN.
Soh et al. 2021	Average Filter and Data Normalisation.	Gaussian mixture model (GMM).
Wu et al. 2020	Time Domain and Pearson Correlation Coefficient.	LDA, RF, GMSVM, E-SVMs, LSTM, 1D-CNN, Heterogeneous Stacking and EGMSVMs.
Xu et al. 2020	Time Domain, Spearman's Correlation, and Principal Component Analysis (PCA).	Multi-output SVM and Multi-class SVM.
Liu et al. 2023	Time and Frequency Domains, Confidence Attenuation-Based KNN and LDA.	MO-HKELM, MO-RPELM, MO-DSAE, MO-RF, MO-SVM, and MO-KNN.
Khan et al. 2023	Fast Fourier Transform and Time Domain.	Residual Network (ResNet).

Journal	Data Preparation	Classifier Models
Buabeng et al. 2023	Improved Complete Ensemble EMD with Adaptive Noise (ICEEMDAN) and Principal Component Analysis (PCA).	LSSVM, LDA, SVM, and ANN.
Kim et al. 2022	Spearman's Rank Correlation Coefficient, Pearson Correlation Coefficient, Boruta Algorithm, Time and Frequency Domains, and Linear Discriminant Analysis.	LDA, LR, SVC, DT, RF, XGBoost, LightGBM, and multi-layer perceptron.
Ma et al. 2021	Time-frequency analysis. Normalise and preprocess the raw signals and transform the multilabel classification problem into a regression problem by setting thresholds for the label values.	MRSIFF-2, MSFTFF, PCNN and FAC-CNN, DNN, 1D-CNN and 2D-CNN.

Their research (Guo et al., 2019) introduces a methodology for determining the health status of four distinct hydraulic system conditions: the cooler, valve, pump, and accumulator. The study employed statistical features derived from sensor signals, utilised the Pearson correlation coefficient for feature selection, and implemented an ensemble Support Vector Machine (SVM) based on stacking for health condition identification. An empirical study was conducted using a hydraulic test rig developed by ZeMA to replicate diverse fault scenarios within the hydraulic system. Comparative analyses were performed with classical methods, including Linear Discriminant Analysis (LDA) and Artificial Neural Networks (ANN), demonstrating that the proposed methodology exhibited superior identification accuracy across all four conditions. Specifically, the cooler and pump achieved 100% accuracy in LDA, ANN, and Ensemble SVM. The highest average accuracy among the methods was observed in the Ensemble SVM, reaching 88.675%.

In their research, (Quatrini et al., 2019) proposed a machine-learning approach to classify the degradation state of four conditions of a complex hydraulic system based on sensor data and cycle features. The paper applied various methods to extract and select representative features from the raw sensor data, such as signal shape functions, distribution density characteristics, Pearson's correlation, and linear discriminant analysis. The authors tested four machine learning algorithms for the classification task: neural network, support vector machine, logistic regression, and decision forest. They evaluated the accuracy and F-score of each algorithm for each label and compared the results with previous studies on the same dataset. The paper found that the best performance was achieved by Pearson with 60 features for the cooler (99.89%) and valve conditions (100%), LDA with 60 or 80 features for the internal pump leakage (99.89%) and hydraulic accumulator (99.16%), and LDA with 80 features and decision forest algorithm for the stable flag(95.63%). The paper also found that the decision forest algorithm was robust to noisy features and that the support vector machine and logistic regression algorithms performed poorly on most labels.

The research paper (Chekkala, 2020) delves into predictive maintenance for fault diagnosis and failure prognosis within hydraulic systems by applying data analytics and machine learning methodologies. The study utilised a multi-sensor dataset from the UCI Machine Learning Repository, encompassing sensor and vibrational data from a hydraulic test rig featuring five conditions: cooler, valve, pump, accumulator, and stable flag. The paper employed three distinct approaches to model the fault conditions and system stability, including parametric and non-parametric techniques LR and ANN, gradient-boosting decision tree algorithms XGBoost, LightGBM, and CatBoost, and a bagging technique (RF). Performance assessment of each model was conducted utilising classification metrics such as accuracy, confusion matrix, precision, recall, and F1-score. These three methodologies were applied individually to each condition within the hydraulic system, with the cooler exhibiting optimal results across all three approaches, ANN, CatBoost, and RF, and achieving a perfect accuracy rate of 100%.

In their research, (Helwig et al., 2015) presented a systematic approach for automated training of condition monitoring systems for complex hydraulic systems based on linear discriminant analysis (LDA) and feature extraction and selection from sensor data. To evaluate the performance of their proposed system, the authors developed a hydraulic test rig that allows simulation of different fault scenarios and grades of severity for conditions such as cooler, valve, pump, and accumulator. The authors used cycle-based and window-based features extracted from the time and frequency domains of the sensor signals and evaluated their correlation with the fault characteristics. They achieved high classification rates for cooler and valve faults using cycle-based features and fixed or random working cycles. The cooler and valve had the best classification rate results, 100%, in all methods. However, the valve result for SVM RBF was 95.7%. The highest average accuracy of the methods is ANN, 82.6%. The classification rates for pump and accumulator faults were improved using window-based features and longer-time windows. The authors compared their LDA-based method with alternative classifiers such as artificial neural networks and support vector machines and found similar or slightly better performance.

The research paper by (Gaurkar et al., 2021) proposed predictive maintenance of industrial machines using machine learning. They used the UCI Machine Learning Repository as the data source and implemented neural networks, long short-term memory, and random forest algorithms to process and analyse the sensor data. They evaluated the performance of the algorithms in classifying four types of faults (cooling, valve, pump, and accumulator) and predicting the remaining useful life of the hydraulic system. They reported that the LSTM algorithm achieved the highest accuracy for cooling (100%), pump leaks (99%), Accumulator (97%) and Valve (95%) and that the system could successfully predict the breakdown period of the hydraulic conditions.

The research paper by (Berghout et al., 2021) presented a machine-learning paradigm for implementing condition-based maintenance in intricate industrial systems. Their methodology was applied in a hydraulic plant with four distinct conditions, each susceptible to various faults. The proposed approach involved a multi-step process encompassing feature extraction and selection, compressed sensing, and training of an artificial neural network with an augmented hidden layer, Auto-NAHL. Incorporating the augmented hidden layer facilitated the network in assimilating multiple representations derived from various random linear mappings, ultimately producing a singular comprehensive representation. Network training was executed through least squares and particle swarm optimisation, and the evaluation encompassed diverse classification benchmarks and the hydraulic system dataset. The results demonstrated their method's superior classification accuracy, attaining 100% for the cooler, valve, and pump conditions and 96.37% for the accumulator. Additionally, the study highlighted the method's notable attributes of low complexity and reduced computation time.

The research paper by (Soh et al., 2021) presented a probability-based algorithm for condition monitoring of a complex hydraulic system. Their algorithm used the Gaussian Mixture Model (GMM) to train and classify the data obtained from 17 sensors in the hydraulic system. They also compared the performance of using different subsets of data based on their importance or interval. Both methods had the same results for the cooler (99.9%), the pump (99.7%), and the stable (94.1%), while method 2 had slightly better results for the valve (96.3%) and the accumulator (97%).

The research paper by (Wu et al., 2020) introduces a novel health evaluation method for complex degradation systems based on ensemble generalised multiclass support vector machines (EGMSVMs). The method preprocesses and selects features from multisensory data, constructs and trains EGMSVMs using stacking-based ensemble learning and GMSVM algorithm and applies EGMSVMs to online health evaluation of multiple conditions. The method is tested on a simulated hydraulic platform with four conditions: cooler, valve, pump, and accumulator. The method achieves higher accuracy than other machine learning algorithms such as LDA, RF, SVM, 1-D CNN, and LSTM on all conditions, with accuracy of cooler (100%), valve (100%), pump (100%), and accumulator (76.5%).

The research paper by (Xu et al., 2020) proposes a compound fault diagnosis method in hydraulic systems using a multi-output support vector machine (SVM). The method selects the most relevant sensor signals based on Spearman ranking correlation coefficients, reduces the data dimensionality using principal condition analysis (PCA), and designs a multi-output classifier based on SVM and One-vs-all. The method is applied to a dataset collected from a hydraulic test rig with four simulated fault types. The results show that the multi-output SVM method provided better results for the pump (96.69%), valve (97.93%), and accumulator (94.90%), while the multi-class SVM method achieved an accuracy of 100% for the cooler.

The research paper by (Liu et al., 2023) proposed an advanced composite fault diagnosis method tailored for hydraulic systems, employing a combination of linear discriminant analysis (LDA) and multi-output hybrid kernel extreme learning machine (MO-HKELM)—the proposed methodology encompassed three integral stages: data pre-processing, data selection, and fault classification. LDA was employed to select sensitive channels and features associated with each condition derived from multi-channel signals. Furthermore, the MO-HKELM technique was utilised to output the fault status of multiple conditions concurrently by integrating a multi-output strategy into the HKELM framework. The reported outcomes indicated that their method achieved the highest accuracy for pump diagnosis (99.91%) compared to results obtained by other researchers. Even so, notable accuracies were also attained for the cooler (99.94%), valve (99.98%), and accumulator (99.54%), albeit marginally lower than the findings of other research endeavours.

Their research (Khan et al., 2023) proposed a novel approach for the fault classification of the cooling system of a hydraulic test rig using a deep learning model based on ResNet-18 architecture. They generated spectrograms from the time series data and used them as inputs for the ResNet-18 model. They compared the results with other machine learning and deep learning models. They found that the ResNet-18 model achieved a high accuracy of 95% for the classification of the fault conditions of the cooler system.

In their paper, (Buabeng et al., 2023) introduce a hybrid artificial intelligent predictive maintenance model designed for multiclass fault classification within a hydraulic system. The model integrates Improved Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (ICEEMDAN), Principal Component Analysis (PCA), and Least Squares Support Vector Machine (LSSVM), optimised through the amalgamation of Coupled Simulated Annealing and Nelder-Mead Simplex optimisation algorithms, denoted as ICEEMDAN-PCA-LSSVM. The authors apply this model to a benchmark dataset derived from multi-sensors in a hydraulic test rig, conducting a comparative analysis with three established methods: LSSVM, LDA, SVM, and ANN. The accuracy results demonstrate superior performance with ICEEMDAN-PCA pre-processing, showcasing LSSVM achieving an accuracy of 99.44% for the accumulator, LSSVM and ANN achieving 99.83% for the cooler, LSSVM and SVM achieving 100% accuracy for the pump, and LSSVM achieving 99.84% accuracy for the valve. Notably, for the cooler without ICEEMDAN pre-processing, LSSVM attains the same accuracy of 99.83%.

(Kim et al., 2022) present a study on anomaly detection and fault diagnosis of hydraulic system conditions using machine learning and feature extraction techniques. They extract and select features based on the shape and density characteristics of the sensor data and apply methods such as linear regression, Spearman's rank correlation coefficient, Pearson correlation coefficient, and Boruta algorithm. They report that the best accuracy results for each condition are 88.4% for the cooler using Boruta Algorithm and RF, 96.4% for the valve using Pearson algorithm and XGBoots Classifier, 95.5% for the pump using Pearson algorithm and LightGBM Classifier, and 98% for the accumulator using Spearman algorithm and LightGBM Classifier.

The research paper by (Ma et al., 2021) presents an innovative approach for the fault diagnosis of intricate mechanical systems utilising multirate sensor data. The method strategically utilises multidimensional convolution blocks to extract fault features from raw signals with varying sampling rates, amalgamating them into a unified feature vector conducive to fault classification. Additionally, the methodology incorporates time-frequency analysis to unveil fault-related information in the time-frequency domain. The MRSIFF-2 method is implemented on a hydraulic system condition monitoring dataset, showcasing superior accuracy compared to existing methods across five distinct tasks: cooler (99.98%), valve (99.97%), pump (100%), accumulator (97.60%), and stable flag (97.35%). The research paper provides compelling evidence regarding the efficacy and robustness of the proposed method for the fault diagnosis of intricate mechanical systems.

2.4.3 Staged Failure Analysis

This section has revealed that certain researchers have employed an approach to predict some or all condition failures in various hydraulic system failure stages. An analysis of *Table 2.4* has indicated that the two papers utilised data normalisation and sampling methods as data preparation techniques.

Table 2.4: Staged Failure Analysis Methodology

Journal	Data Preparation	Classifier Models
Keleko et al. 2023	Data normalisation (Min-Max), One-Hot-Encoding, Data Sampling, and Cross-Validation.	DNN (DeepSHAP explainable XAI).
Zhang et al. 2022	Down-sampling, Up-sampling, and Data Normalisation (Min-Max).	Fully Convolutional Networks (FCN).

Their research (Keleko et al., 2023) proposed a comprehensive framework for the condition monitoring of a hydraulic system, utilising a combination of deep neural network (DNN) classification and DeepSHAP explanation methods. The research employs a data-driven approach to predict the degradation states of five critical conditions within the hydraulic system: the cooler, valve, internal pump leakage, hydraulic accumulator, and stable stage. Additionally, the DeepSHAP method is incorporated to elucidate the significance and contribution of each sensor in the decision-making process of the DNN model. The DNN model's performance is evaluated through accuracy, F1-score, recall, and precision. Moreover, the DeepSHAP method is benchmarked against alternative explanation methods, including LIME, SHAP, and DeepLIFT. The outcomes of the DNN model for multi-class classification of degradation levels within each state of the hydraulic system demonstrate exemplary results, with the cooler achieving a notable F1-score of 100% across various stages and the pump achieving a perfect 100% F1-score in the no-leakage scenario in comparison to other stages.

In their research, (Zhang et al., 2022) proposed a multi-fault diagnosis method for hydraulic systems using time-series representation learning based on the Fully Convolutional Networks (FCN). The paper presented a four-step methodology: data preprocessing, feature extraction, fault classification, and performance evaluation. The method can handle multi-rate data and multiple faults with different degrees. The paper evaluated the proposed method on a hydraulic system test bench and demonstrated good performance. The study presented two distinct cases, one utilising sampled data and the other fast sample data slowly. The results showed that the fast sample data outperformed the slowly sampled data, achieving an F1 score of 100% for various stages of failures in the cooler, pump, and valve conditions. Moreover, the accumulator condition achieved 100% accuracy in the normal failure stage compared to the other three failure stages, with slight failure achieving a score of 99.35%, medium failure achieving a score of 98.86%, and severe failure achieving a score of 99.69%.

2.4.4 Summary

Research studies used machine learning and deep learning algorithms to predict hydraulic system failures. In the previous sections, we discussed three approaches researchers have adopted to predict faults in hydraulic systems. The first approach involves predicting the failure of the entire hydraulic system, while the second approach focuses on predicting the failure of specific conditions. The third approach employs a staged failure analysis to predict condition failures in various stages of failure.

Table 2.5: Data Preparation Methods

Name	Freq
Time domain	8
Data Normalisation (Min-Max)	7
Pearson Correlation Coefficient	6
Distribution density and Principal Component Analysis (PCA).	4

Name	Freq
Signal shape features, Linear Discriminant Analysis, and Spearman's correlation analysis.	3
One-Hot Encoding and Frequency Domain.	2
Borderline-SMOTE, Nearest Centroid (NC) with Dtw barycenter averaging (DBA), Gain Information (GI), Main Failure Indicators (MFIs), feature importance (FI), Cluster-Based Feature Selection (RkSE), Quantile Transform Scalar, Classification Imbalance using (SMOTE), t-distributed Stochastic Neighbor Embedding (t-SNE), Uniform Manifold Approximation and Projection (UMAP), Attribute Selection, Attribute, Generation, and Discretization, Compressed Sensing (CS), Average Filter, Confidence Attenuation-Based KNN, Fast Fourier Transform, Improved Complete Ensemble EMD with Adaptive Noise (ICEEMDAN), Boruta Algorithm, Time-frequency analysis, Down-sampling, Up-sampling, Data Sampling, and Cross-Validation.	1

Table 2.6: Machine Learning Algorithms

Name	Freq
RF	9
LR and LDA	7
DT and SVM	5
NB and XGBoost	4
KNN	3
Ensemble SVM, LightGBM, and Linear SVM	2
Gaussian SVM, ADA, Ridge, Softmax, SSL-SVM, SSL-LR, SSL-DT, SSL-NB, SSL-RF, MO-RPELM, MO-RF, MO-SVM, MO-KNN, CART, DF, Catboost, GMM, GMSVM, Heterogeneous Stacking, Multi-output SVM, Multi-class SVM, LS-SVM, SVC, Auto-NAHL, EGMSVMs, MO-HKELM, and SVM Radial Basis Function	1

Table 2.7: Deep Learning Algorithms

Name	Freq
LSTM and ANN	4
1D-CNN	3
DNN	2

Name	Freq
CNN, ResNet, DBN, FCN, MO-DSAE, NN, Multi-Layer Perceptron, MRSIFF-2, and 2D-DNN	1

This research project consists of two stages: data preparation and modelling. In the data preparation process, several methods are utilised. The Signal Shape method, which uses functions like slope, average, and maximum, captures deterministic and reproducible fault effects over short time scales and reduces data complexity by transforming high-dimensional data to low-dimensional space (Helwig et al. 2015, Kim et al. July 2022). The Distribution Density method, using statistical functions like median, deviation, and kurtosis, captures stochastic and non-reproducible fault effects over long time scales and performs well with ample data(Helwig et al. 2015, Kim et al. July 2022). Data Normalization scales data values to a specified range, ensuring equal contribution from all features and simplifying model training (Berghout et al., 2021; Zhang et al., 2022). For feature selection, the Pearson Correlation method ranks features based on their correlation with the label, reducing model training time and complexity (Quatrini et al., 2020; Wu et al., 2020), while the Linear Discriminant Analysis (LDA) method generates a new feature dataset that best separates classes (Liu et al., 2023). These methods are compared to determine their performance with the classifier model and are among the most commonly used methods in data preparation, as indicated in *Table 2.5*.

The most and least frequent machine learning algorithms displayed in *Table 2.6*, RF and Catboost, will be implemented for the modelling stage. RF is a machine learning method that uses multiple decision trees to classify data and reduce overfitting (Kim et al., July 2022). It has shown the best performance in many indicators such as accuracy, F1-score, and AUC among ten machine learning algorithms tested on the hydraulic system dataset(Kim et al. July 2022). Catboost is a robust and efficient model for predictive modelling, especially for handling categorical features (Chekkala, 2020). In addition, the most frequent deep learning algorithm, as illustrated in *Table 2.7*, which is LSTM, will be implemented since much research has achieved remarkable results. LSTM is a recurrent neural network that can handle sequential data and long-term dependencies (Berghout et al., 2021). It has been used for various tasks such as fault detection, fault diagnosis, and remaining valid life prediction of the hydraulic system (Mallak et al., 2021).

The performance of the machine learning models is evaluated using various metrics such as accuracy, F1-score, recall, and precision. It has been observed that several research studies have previously delved into the various aspects of predictive maintenance modelling for hydraulic systems. However, these studies have yet to explore the correlation between the conditions present in the hydraulic system. In addition, most existing studies have utilised machine learning algorithms, with deep learning techniques being primarily disregarded.

Chapter 3: Theoretical Framework

Chapter 3, "***Theoretical Framework***," introduces the theoretical framework of the research project. The Cross Industry Standard Process for Data Mining (CRISP-DM), which consists of six sequential stages, has been adopted as the research framework.

3.1 Research Framework

The research framework employed in this study adheres to the Cross Industry Standard Process for Data Mining (CRISP-DM), serving as the fundamental structure for the data science workflow. Comprising six sequential stages, namely Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation, and Deployment, this process model delineates a systematic approach to the various facets of the data science process. Each stage is designed to cater to distinct elements of the overall data science procedure, encompassing comprehension of business objectives and implementing outcomes. Adopting this structured process model ensures that the project team can execute a well-organized, efficient, and effective data science process.

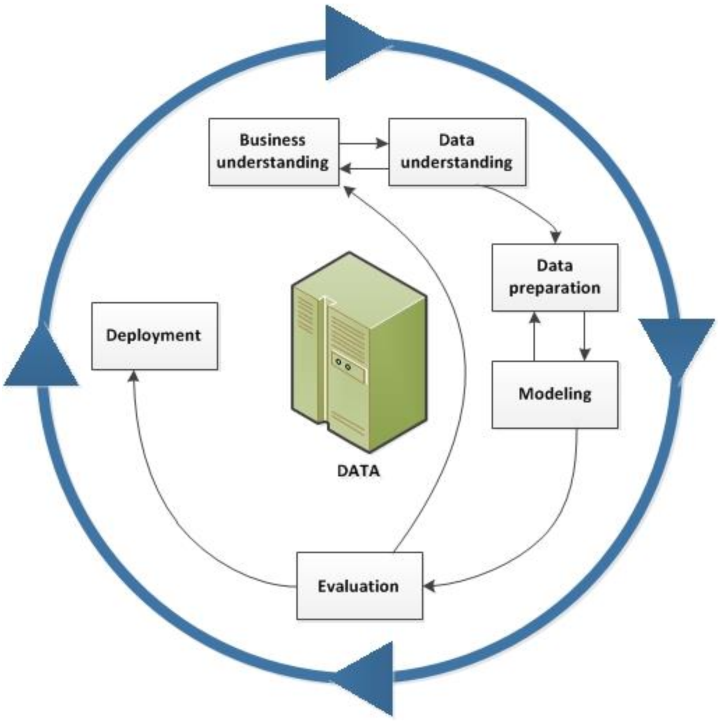


Figure 3.1 Cross-Industry Standard Process for Data Mining (IBM, 2021)

3.1.1 Business Understanding

The Business Understanding phase is vital to the data science process and consists of four critical tasks. These tasks involve determining business objectives, assessing project situations, defining data science goals, and producing a project plan. Completing this phase sets the foundation for an efficient data science process. The business understanding of the Hydraulic System Predictive Maintenance Project emphasises the significance of early success measurement, the pivotal role of hydraulic systems in industrial applications, and the indispensable nature of predictive maintenance in ensuring safety and cost-effectiveness. Focusing on condition stability, risk mitigation, and cost-benefit analysis aligns with the broader business objectives of revenue maximisation and cost minimisation. The selection of a multi-class classification dataset and the analytical approach utilising machine learning classification algorithms underscore the commitment to achieving accurate predictions. Using classification metrics to evaluate algorithm performance ensures the reliability of predictions for effective maintenance scheduling and contributes to the overall success of predictive maintenance in hydraulic systems. Furthermore, the project plan aligns with the broader business goals of revenue maximisation and cost minimisation, and the Key Performance Indicators (KPI) and Objectives and Key Results (OKR) are closely aligned with the understanding of the business, as seen in *Table 3.1*.

Table 3.1: Business Understanding

Key Performance Indicators (KPI)	Objectives and Key Results (OKR)
- Creating or transforming features using signal shape and distribution density measures. - Standardising or normalising numerical features using data normalisation techniques. - Relevant and informative features were selected using the Pearson correlation coefficient and linear discriminant analysis methods. - Choosing and comparing different classification methods such as RF, Catboost, and LSTM for fault prediction. - Measure and improve the model's performance using precision, recall, F1-score, and accuracy.	**Objective:** To construct a predictive model that forecasts the maintenance requirements of a hydraulic system using machine learning techniques. Key Results: - Identify decline patterns on sensors and condition variables. - Find the correlation between the condition variables.

3.1.2 Data Understanding

Data Understanding is a critical step in data analysis. It involves comprehending the collected data and assessing its quality, identifying potential issues, and exploring relationships between variables. Data collection often occurs in parallel with problem definition, and once the problem is defined, data collection becomes more formal. Adequate data understanding is essential for generating reliable insights and making informed decisions.

3.1.3 Data Preparation

The data preparation process in machine learning comprises three essential steps: feature extraction, scaling, and selection, as seen in *Figure 3.2*. Feature extraction involves the creation of new features or transforming existing ones to make them more informative to machine learning models. This process is crucial for capturing the underlying patterns present in the data. Scaling is standardising or normalising numerical features to a similar scale. This is important since several machine learning algorithms are sensitive to the input features' scale. Lastly, feature selection is an approach for choosing a subset of relevant features from the original set to improve the model's performance and reduce dimensionality. Doing so can lead to more straightforward and interpretable models and mitigate the risk of overfitting. Collectively, these steps contribute to enhancing the model's predictive performance and generalisation ability, which are critical for effective machine learning.

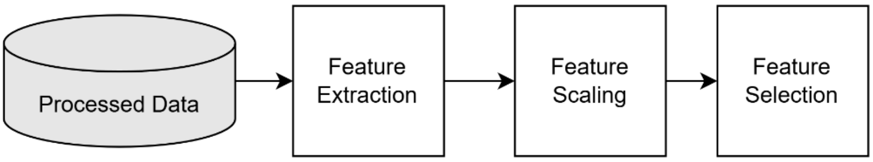


Figure 3.2 A standard pipeline for feature extraction, scaling, and selection

Feature Extraction:

A condition in developing machine learning models involves transforming or creating new features that may enhance the model's performance. Signal shape features include the slope of a linear fit and position of maximum value and distribution density measures like variance, skewness, and kurtosis.

Functions representing the signal shape include the following parameters:

- Slope Of Linear Fit (SOLF): a parameter denotes the slope obtained from a singular measurement of the specific sensor during each operational cycle.
- Maximum Value Position (MVP): a parameter that denotes the specific position of the maximum value recorded for each sensor in every operational cycle. It serves the purpose of ranking the maximum value of the respective sensor in comparison to all other maximum values acquired from various lines.

Functions of Distribution Density Measures:

- Median (MED): This parameter signifies the value considered by the statistical units in the midpoint of the distribution for each sensor and cycle.
- Variance (VAR): For every sensor and cycle, this metric represents the extent of variability within the data.
- Skewness (SKEW): It indicates the symmetry index inherent in the data for each sensor and cycle.
- Kurtosis (KURT): This parameter reflects, for every sensor and cycle, a deviation from the normal distribution, indicating either more significant flattening or more significant stretching of the distribution than a normal distribution.

Feature Scaling:

A preprocessing technique standardises numerical features to ensure they are on similar scales. Data normalisation is a popular form of scaling that prevents certain features from dominating the learning process due to differences in their scales.

The function of Data Normalisation:

A normalisation operator becomes necessary when the features exhibit considerable variations in their ranges. This procedure aims to mitigate the impact of the magnitude of individual features during the learning phase. Specifically, the min-max normalisation is employed, resulting in the normalised signal denoted as X_i, with X_{max} and X_{min} representing the maximum and minimum values within the signal X_i. The collective pre-processed data, comprising all samples and dimensions, is symbolised as X_{Norm}.

$$X_{Norm} = \frac{X_i - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

Feature Selection:

Techniques like the Pearson Correlation Coefficient and Linear Discriminant Analysis can be used to identify the most informative and discriminative features.

Functions of Pearson Correlation Analysis:

The Pearson correlation coefficient serves as a valuable statistical tool for discerning the magnitude and direction of linear associations between variables. Represented by the symbol "r," this statistical measure quantifies the strength and direction of a linear relationship existing between two variables.

The coefficient's numerical value ranges from -1 to +1, where +1 signifies a flawless positive linear relationship, -1 denotes a flawless negative linear relationship, and 0 implies the absence of a linear correlation. The calculation of the Pearson correlation coefficient follows a defined formula, expressed as:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}} \tag{3.2}$$

“x_i” and “y_i” represent individual features of the x-variable and y-variable, respectively, and “x bar” and “y bar” denote their respective means. This coefficient evaluates the features' strength and direction of a linear relationship. Understanding the hydraulic system parameters, their interactions, and the correlation between these features is essential for predictive maintenance and condition monitoring.

Functions of Linear Discriminant Analysis:

Meanwhile, LDA serves both dimensionality reduction and feature selection, pinpointing the most discriminative features. This enhances the efficiency of grouping data into classes by projecting it into a reduced feature space while preserving essential discriminant information. Implementing LDA enables the creation of a new feature dataset that captures key feature combinations, contributing to improved classification in our hydraulic system analysis.

3.1.4 Modelling

Three distinct classification methods will be applied: two machine learning approaches (RF and Catboost) and one deep learning technique (LSTM).

Random Forest (RF):

According to (Zhang et al., 2012), random Forests is a powerful and versatile machine-learning technique that can handle regression and classification tasks efficiently and accurately. They are based on combining multiple decision trees, each trained on a random subset of the data and features and aggregating their predictions. Random Forests offer several computational and statistical advantages over other methods, such as fast training and prediction, low sensitivity to tuning parameters, built-in error estimation, high-dimensional applicability, parallel implementation, and additional features such as variable importance, class weighting, missing value imputation, visualisation, outlier detection, and unsupervised learning.

Categorical Boosting (Catboost):

According to the book “Artificial Intelligence in Highway Safety” by (Das 2022), CatBoost is a gradient-boosting decision tree algorithm that can handle categorical features effectively during training. It uses a random permutation method and an unbiased boosting technique to reduce the model's information loss and variance. CatBoost also considers all the possible combinations of categorical features when generating splits for the tree, which enhances its performance and generalisation ability.

Long Short-Term Memory (LSTM):

According to the book “Supervised Sequence Labelling with Recurrent Neural Networks” (Graves, 2012), Long Short-Term Memory (LSTM) stands as a type of recurrent neural network designed to address the vanishing gradient problem inherent in conventional RNNs, thereby extending the contextual scope. The distinctive feature of LSTM lies in its utilisation of memory blocks comprising self-connected memory cells and three gates that regulate the flow of information into and out of these cells. Notably, LSTM exhibits proficiency in capturing long-term dependencies within sequential data, and its efficacy has been demonstrated across diverse applications, including but not limited to time series prediction, natural language processing, and speech recognition. This discourse introduces the fundamental architecture of LSTM, elucidates the computation of error gradients, discusses certain extensions of LSTM, and explores the impact of preprocessing on the management of long-range dependencies.

3.1.5 Evaluation

The project will use six metrics to evaluate the model: Precision, Recall, F1-Score, and Accuracy. These metrics will help to measure, compare, and improve the model’s performance.

Accuracy

Accuracy is a widely used metric for classifier performance. It measures the ratio of correct predictions to all predictions made by the model. The formula for accuracy from the confusion matrix is:

$$\text{Accuracy} = \frac{TP+TN}{TP + FP + TN + FN} \tag{3.3}$$

Precision

Precision, or positive predictive value, is a metric derived from the confusion matrix. It measures the ratio of true positives to all positive predictions. The formula for precision is:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.4}$$

Recall

Recall, or sensitivity, is a metric that measures the proportion of relevant instances that the model correctly identifies. It is calculated as the ratio of true positives to all actual positives. It is also known as hit rate, coverage, or sensitivity. The formula for the recall is:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.5}$$

F1-Score

The F1 score is a metric that measures the trade-off between precision and recall. It is the harmonic mean of precision and recall, which is higher when both are high. It helps us optimise a classifier for balanced performance on both metrics. The formula for the F1 score is:

$$\text{F1-Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.6}$$

ROC Curve

The ROC Curve is a graphical tool to evaluate the performance of binary or multi-class classifiers. It stands for Receiver Operating Characteristic, a term from radar signal detection. It shows the trade-off between a classifier's True Positive Rate (TPR) and the False Positive Rate (FPR) at different threshold levels. The ROC Curve is plotted by varying the threshold for the probability or score output of the classifier and calculating the corresponding TPR and FPR. The ROC Curve can help to compare different classifiers, select the optimal threshold, and measure the overall quality of the classifier.

AUC

The Area Under the Curve (AUC) is a metric that measures the performance of a classifier based on its ROC Curve. The AUC is the two-dimensional area under the ROC Curve, ranging from 0 to 1. A higher AUC indicates a better classifier that can distinguish between the positive and negative classes.

3.1.6 Deployment

Power BI is a tool for visualising data and creating interactive dashboards. The dashboards can be customised and shared for collaborative decision-making. A Power BI dashboard is made of tiles that can show reports, images, text, web content, or data from other sources. The project’s final stage involves deploying a dashboard.

Chapter 4: Research Methodology

Chapter 4, "***Research Methodology***," describes the research methods and techniques used to implement the proposed predictive model for hydraulic system maintenance1. The chapter follows the CRISP-DM framework and covers five stages: data understanding, data preparation, modelling, evaluation, and deployment. The chapter also explains the rationale and justification for choosing each stage's specific methods and tools.

The CRISP-DM methodology is explained in detail, and the steps followed in this project are illustrated in *Figure 4.1*. The first stage is data understanding, which involves data transformation, exploration, and analysis. The second stage is data preparation, which consists of three steps: feature extraction using signal shape and distribution density, feature scaling using data normalisation, and feature selection using two methods, namely Pearson correlation and discriminant analysis. These two methods are then compared based on the results of the classifier modelling, which uses three algorithms: RF, Catboost, and LSTM. The evaluation stage compares the performance of the different selection methods and classifier models. The final stage is deploying the Power BI dashboard, which displays the data and the models using Python and R scripts.

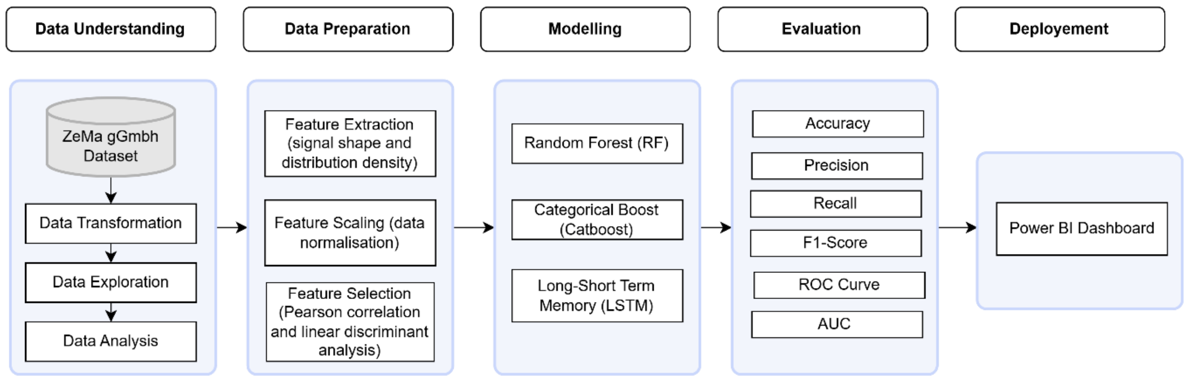


Figure 4.1 Methodology Flowchart

4.1 Data Collection

This research project utilises an open-source dataset from ZeMA gGmbH (Schneider et al., 2018), a German research institute specialising in intelligent engineering systems. The dataset contains sensor data from a hydraulic test rig that simulates four types of faults with different severity levels. The test rig comprises a primary working circuit and a secondary cooling-filtration circuit connected through an oil tank. The system cyclically measures process values for all sensors every 60 seconds, while the four hydraulic conditions are quantitatively varied.

4.2 Data Understanding

The document "description.txt," provided by (Schneider et al., 2018), a renowned German research institute, provides a detailed overview of the data. *Table 4.1* outlines the independent variables, represented by various sensors and their respective physical quantity, unit, and sampling rate. *Table 4.2*, on the other hand, highlights the dependent variables, which include cooler, valve, pump, accumulator, and stable flag attributes, along with their descriptions and equipment stages.

Table 4.1: Hydraulic System Sensors (independent variables)

Sensor	Physical quantity	Unit	Sampling rate
PS (1-6)	Pressure	bar	100 Hz
EPS1	Motor power	W	100 Hz
FS (1-2)	Volume flow	l/min	10 Hz
TS (1-4)	Temperature	°C	1 Hz
VS1	Vibration	mm/s	1 Hz
CE	Cooling efficiency (virtual)	%	1 Hz
CP	Cooling power (virtual)	kW	1 Hz
SE	Efficiency factor	%	1 Hz

Table 4.2: Hydraulic System Components (dependent variables)

Attribute	Description	Values
Cooler condition/%	The efficiency of the cooler	3: close to total failure 20: Reduced efficiency 100: full efficiency
Valve condition /%	The switching behaviour of the valve	100: Optimal switching behaviour 90: Small lag 80: Severe lag 73: Close to total failure
Internal pump leakage	The leakage level of the pump	0: no leakage 1: weak leakage 2: Severe leakage

Attribute	Description	Values
Hydraulic accumulator/ bar	The pressure level of the accumulator	130: Optimal pressure 115: slightly reduced pressure 100: severely reduced pressure 90: Close to total failure
Stable flag	The stability of the system	0: Conditions were stable 1: Static conditions might not have been reached yet

The data understanding stage, as shown in *Figure 4.2*, involves transforming the text files into data frames. Then, the data is explored by examining and documenting its essential characteristics, such as data format, number of records, and field names. The data is also queried, visualised, and analysed to identify the relationships among the variables. A correlation matrix was applied to determine the association between all sensors in the hydraulic test rig.

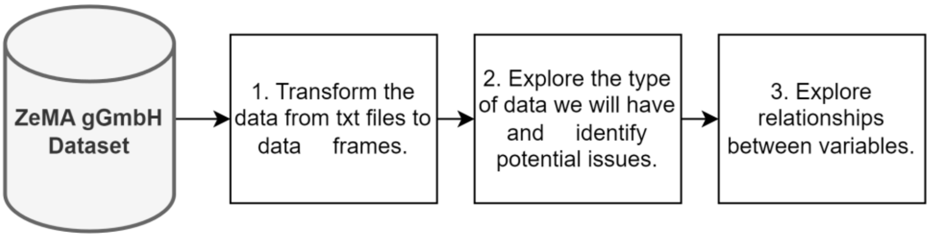


Figure 4.2 Data Understanding

4.3 Data Preparation

Signal shape features (slope of linear fit and position of maxim value), Distribution density (variance, skewness, and kurtosis), Data normalisation, Pearson Correlation Coefficient, and Linear Discriminant Analysis.

As shown in *Figure 4.3*, the data preparation process in machine learning consists of three key steps: feature scaling, extraction, and selection. Feature scaling is applied using data normalisation to standardise or normalise numerical features to a common scale. This is essential because many machine learning algorithms are affected by the scale of the input features. Feature extraction is performed using signal shape and distribution density to extract the latent patterns in the data. Feature selection is a technique for selecting a subset of relevant features from the original set to enhance the model's performance and reduce dimensionality. Two methods are employed for feature selection: Pearson correlation and linear discriminant analysis. These methods can result in more straightforward and interpretable models and prevent overfitting.

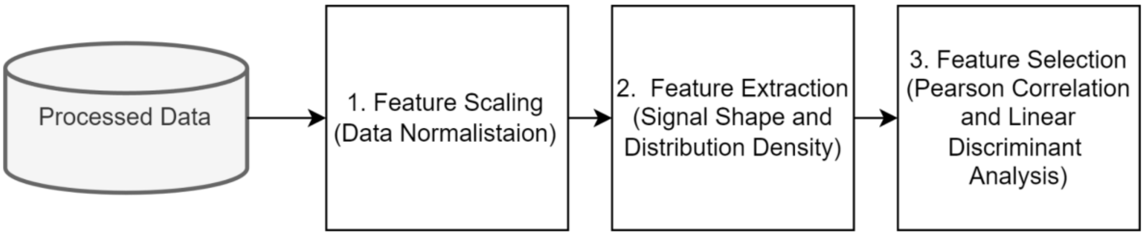


Figure 4.3 Data Preparation Methodology

4.3.1 Feature Extraction and Engineering using Signal Shape and Distribution Density:

Two methods are employed. The first, signal shape features, detects deterministic and repeatable fault effects on sensor signals over short durations, including the slope of a linear fit and the maximum value's position for each sensor and cycle. The second, distribution density measures, identifies random and non-repeatable fault effects on sensor signals over longer durations, including the median, variance, skewness, and kurtosis for each sensor and cycle. These methods are applied to the hydraulic system's 17 sensors, yielding 68 features per cycle. These features will reveal the hydraulic system faults' hidden patterns and characteristics, enhancing the classification models' performance. This process not only streamlines the data but also aids in other preprocessing steps like eliminating irrelevant words and reducing words to their root form, significantly enhancing the machine learning model's performance by reducing the influence of irrelevant or redundant words on the data.

4.3.2 Feature Scaling using data normalisation:

Data normalisation changes the original values to a range from 0 to 1. Data normalisation can lower outliers' impact and speed up the convergence of some algorithms. In this project, data normalisation will be used on the sensor data from the hydraulic test rig, which has different units and ranges. Normalising the data will give the features equal weight to the machine learning models and simplify the training process.

4.3.3 Feature Selection using Pearson correlation and linear discriminant analysis:

Feature selection is choosing a subset of features from the original set that are most relevant for the model's performance and dimensionality reduction. These features are ranked based on their correlation with the label, using Pearson correlation, which can help remove redundant or irrelevant features. Another method called linear discriminant analysis (LDA) is used to create a new feature dataset that best separates the classes, which can improve the features' discriminative power. These two methods are applied to the hydraulic system dataset, which has sensor data from a hydraulic test rig with four conditions: cooler, valve, pump, and accumulator. Both feature selection methods will be tested with machine and deep learning models to compare which feature selection is better. Using feature selection, the most informative and representative features for each condition can be selected, which can help build more accurate and reliable predictive models for hydraulic system maintenance.

4.4 Modelling

After the data is prepared, the dataset is partitioned into suitable segments to facilitate the training and testing phases of the predictive equipment maintenance model for hydraulic systems. In this context, an 80% training set and a 20% testing set will be generated from the dataset. Upon completing data pre-processing, training was assigned 80% of the data, while 20% was reserved for testing. This allocation is consistent with research suggesting that the best results are obtained using 20-30% of all available test data sets and 70-80% on training according to (Gholamy, et al, 2018). This specific division has been selected to provide the model with ample data for learning, ensuring a robust evaluation of its performance. It is crucial to emphasise the significance of an adequately sized training set to prevent potential issues such as the model's inability to generalise effectively or bias towards a specific subset of the dataset. Therefore, adopting an 80/20 split, where 80% comprises the training data and 20% is designated for testing, is recommended to balance effective learning and rigorous assessment.

Then, the partitioned data will be fed into three classification methods: two machine learning methods (RF and Catboost) and one deep learning method (LSTM). The research will select these methods based on the literature review and their suitability for the problem domain. These methods will forecast the health status and degradation levels of the hydraulic system conditions. The performance and accuracy of these methods will be compared and evaluated using various metrics. The research will identify the best method and perform further analysis to optimise the results.

4.5 Evaluation

After implementation, the classifier models are assessed by various methods to measure their effectiveness and identify their strengths and weaknesses. The methods include calculating common evaluation metrics and producing a classification report. The metrics are accuracy, precision, f1-score, recall, ROC Curve and AUC.

4.6 Deployment

The project's final step is to deploy a dashboard, that includes charts that illustrate the overall degradation states of the hydraulic system components conditions. From the charts, it makes it easier to identify the state of equipment and to monitor it. The Power BI dashboard was developed using Python scripts. The Python scripts enable the dashboard to access and process the data from the models and present the outcomes in various forms. The dashboard also shows the exploratory data analysis conducted using R visuals.

Chapter 5: Implementation

5.1 Data Understanding

The first step of data understanding is extracting the data from all Txt files, which consists of two parts: extracting the hydraulic system sensors and condition data, as shown in *Figure 5.1* and *Figure 5.2*.

```
# Extracting the pressure sensors data from txt files.
ps1=np.genfromtxt(r"Dataset/PS1.txt")
ps2=np.genfromtxt(r"Dataset/PS2.txt")
ps3=np.genfromtxt(r"Dataset/PS3.txt")
ps4=np.genfromtxt(r"Dataset/PS4.txt")
ps5=np.genfromtxt(r"Dataset/PS5.txt")
ps6=np.genfromtxt(r"Dataset/PS6.txt")

# Extracting the Motor power sensor data from txt file.
eps1=np.genfromtxt(r"Dataset/EPS1.txt")

# Extracting the Volume flow sensors data from txt files.
fs1=np.genfromtxt(r"Dataset/FS1.txt")
fs2=np.genfromtxt(r"Dataset/FS2.txt")

# Extracting the Temperature sensors data from txt files.
ts1=np.genfromtxt(r"Dataset/TS1.txt")
ts2=np.genfromtxt(r"Dataset/TS2.txt")
ts3=np.genfromtxt(r"Dataset/TS3.txt")
ts4=np.genfromtxt(r"Dataset/TS4.txt")

# Extracting the Vibration sensor data from txt file.
vs1=np.genfromtxt(r"Dataset/VS1.txt")

# Extracting the Cooling efficiency (virtual) sensor data from txt file.
ce=np.genfromtxt(r"Dataset/CE.txt")

# Extracting the Cooling power (virtual) sensor data from txt file.
cp=np.genfromtxt(r"Dataset/CP.txt")

# Extracting the Efficiency Factor sensor data from txt file.
se=np.genfromtxt(r"Dataset/SE.txt")
```

Figure 5.1 Extracting Hydraulic System Sensors (Independent Variables)

```
# Extracting the Cooler condition, Valve condition, Internal pump leakage,
#Hydraulic accumulator and Stable flag data from profile.txt file.
profile_arr=np.genfromtxt(r"Dataset/profile.txt")
```

Figure 5.2 Extracting Hydraulic System Components (Dependent Variables)

For the seventeen sensors, the mean is calculated along axis 1, which is used to reduce the noise and variation in the sensor readings, improving the accuracy and reliability of the predictive models, as shown in *Figure 5.3*.

```
# Calculating the mean along axis 1 for all pressure sensors
ps1_arr=ps1.mean(axis=1)
ps2_arr=ps2.mean(axis=1)
ps3_arr=ps3.mean(axis=1)
ps4_arr=ps4.mean(axis=1)
ps5_arr=ps5.mean(axis=1)
ps6_arr=ps6.mean(axis=1)

# Calculating the mean along axis 1 for Motor power sensor
eps1_arr=eps1.mean(axis=1)

# Calculating the mean along axis 1 for all Volume flow sensors
fs1_arr=fs1.mean(axis=1)
fs2_arr=fs2.mean(axis=1)

# Calculating the mean along axis 1 for all Temperature sensors
ts1_arr=ts1.mean(axis=1)
ts2_arr=ts2.mean(axis=1)
ts3_arr=ts3.mean(axis=1)
ts4_arr=ts4.mean(axis=1)

# Calculating the mean along axis 1 for Vibration sensor
vs1_arr=vs1.mean(axis=1)

# Calculating the mean along axis 1 for Cooling efficiency (virtual) sensor
ce_arr=ce.mean(axis=1)

# Calculating the mean along axis 1 for Cooling power (virtual) sensor
cp_arr=cp.mean(axis=1)

# Calculating the mean along axis 1 for Efficiency Factor sensor
se_arr=se.mean(axis=1)
```

Figure 5.3 Calculating the Mean along Axis 1 (Independent Variables)

The data extracted from the Txt files were stored in data frames. Separate data frames were created for the independent and dependent variables and then merged into a single data frame named ***'df_final'***. This was because the dependent variable data was contained in a single txt file called **"profile.txt"**. This process is illustrated in *Figure 5.4 to Figure 5.6.*

```
df=pd.DataFrame({"PS1":ps1_arr,"PS2":ps2_arr,"PS3":ps3_arr,"PS4":ps4_arr,"PS5":ps5_arr,"PS6":ps6_arr,
                 "EPS1":eps1_arr, "FS1":fs1_arr, "FS2":fs2_arr,"TS1":ts1_arr,"TS2":ts2_arr,"TS3":ts3_arr,
                 "TS4":ts4_arr, "VS1":vs1_arr, "CE":ce_arr, "CP":cp_arr, "SE":ce_arr})
```

Figure 5.4 Data Framing the Independent Variables

```
df_profile=pd.DataFrame(profile_arr,columns=["Cooler_Condition","Valve_Condition",
                                             "Internal_Pump_Leakage","Hydraulic_Accumulator","Stable_Flag"])
```

Figure 5.5 Data Framing the Dependent Variables

```
df_final=pd.concat([df,df_profile],axis=1)
```

Figure 5.6 Combine Both Data-frames

As seen in *Figure 5.7*, the data frame consists of 2205 rows and 22 columns with a datatype of float64 for data, with no missing values.

```
df_final.info()
✓ 0.0s
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2205 entries, 0 to 2204
Data columns (total 22 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   PS1                     2205 non-null   float64
 1   PS2                     2205 non-null   float64
 2   PS3                     2205 non-null   float64
 3   PS4                     2205 non-null   float64
 4   PS5                     2205 non-null   float64
 5   PS6                     2205 non-null   float64
 6   EPS1                    2205 non-null   float64
 7   FS1                     2205 non-null   float64
 8   FS2                     2205 non-null   float64
 9   TS1                     2205 non-null   float64
 10  TS2                     2205 non-null   float64
 11  TS3                     2205 non-null   float64
 12  TS4                     2205 non-null   float64
 13  VS1                     2205 non-null   float64
 14  CE                      2205 non-null   float64
 15  CP                      2205 non-null   float64
 16  SE                      2205 non-null   float64
 17  Cooler_Condition        2205 non-null   float64
 18  Valve_Condition         2205 non-null   float64
 19  Internal_Pump_Leakage   2205 non-null   float64
 20  Hydraulic_Accumulator   2205 non-null   float64
 21  Stable_Flag             2205 non-null   float64
dtypes: float64(22)
memory usage: 379.1 KB
```

Figure 5.7 Data-frame Information

The correlation matrix in *Figure 5.8* offers insights into the relationships between different sensors in a system. For example, the Pressure Sensors (PS3-6) have a strong positive correlation with each other, indicating that they tend to increase or decrease together. These correlations provide valuable insights into the interdependencies between different parameters in the system, which can be used to optimise performance and detect anomalies.

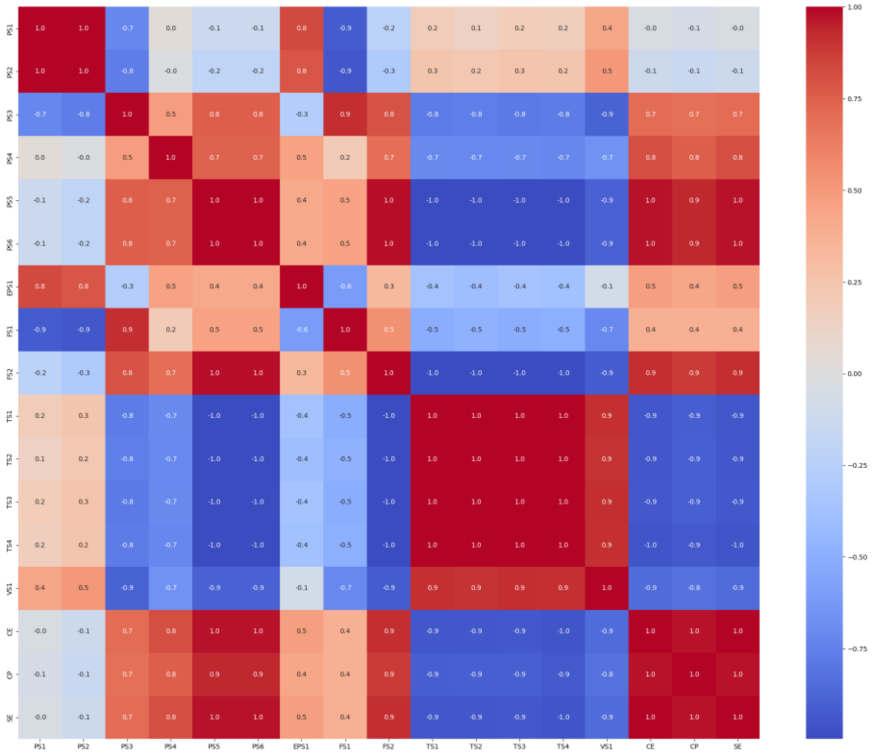


Figure 5.8 Correlation Matrix between Independent Variables

The correlation matrix illustrated in *Figure 5.9* shows the interdependencies among five degradation processes: Cooler Condition, Valve Condition, Internal Pump Leakage, Hydraulic Accumulator, and Stable Flag. For example, the cooler condition has a slight negative correlation with the Hydraulic Accumulator, indicating that as the cooler condition degrades (i.e., efficiency decreases), the pressure in the hydraulic accumulator tends to decrease. These correlations provide valuable insights into the interdependencies among the system's conditions and their impact on system stability.

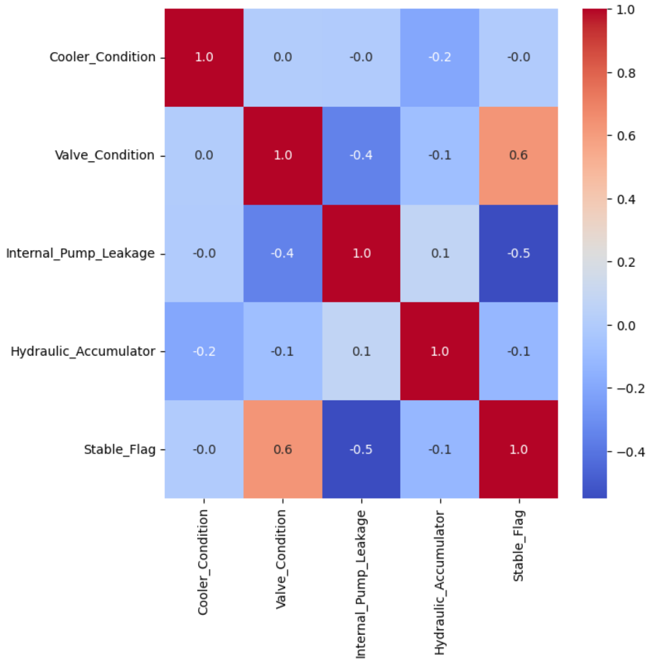


Figure 5.9 Correlation Matrix between Dependent Variables

The correlation matrix in *Figure 5.10* represents the correlation between the sensor data and condition data from a hydraulic test rig's working cycle. The matrix reveals key relationships among these attributes. For instance, the Cooler Condition, CE and SE show a strong positive correlation, indicating a direct relationship. These correlations provide valuable insights into the interdependencies among the system's conditions and their impact on system stability.

	Cooler_Condition	Valve_Condition	Internal_Pump_Leakage	Hydraulic_Accumulator	Stable_Flag
CE	0.991943	-0.001463	-0.000381	-0.204599	-0.015316
SE	0.991943	-0.001463	-0.000381	-0.204599	-0.015316
CP	0.956220	0.008605	-0.015806	-0.241432	-0.003231
PS6	0.950316	0.020284	-0.016262	-0.210282	0.021353
PS5	0.949962	0.020317	-0.016579	-0.209262	0.021458
FS2	0.881340	0.038651	-0.034926	-0.139471	0.055942
PS4	0.832446	-0.146385	0.154950	-0.001021	-0.244314
PS3	0.662852	0.163278	-0.355070	-0.006242	0.237027
EPS1	0.497514	-0.174618	0.426320	-0.239692	-0.267649
FS1	0.330446	0.190141	-0.421702	0.092695	0.288568
PS1	-0.002000	-0.152883	0.311859	-0.198940	-0.232049
PS2	-0.075386	-0.098491	0.309810	-0.180580	-0.206720
VS1	-0.818256	-0.105555	0.156278	0.146263	-0.114613
TS3	-0.906371	-0.024683	0.022200	0.184104	-0.030507
TS1	-0.910865	-0.023876	0.019793	0.193597	-0.029021
TS2	-0.911803	-0.016850	0.003141	0.197030	-0.017206
TS4	-0.922572	-0.022594	0.021279	0.195244	-0.025854

Figure 5.10 Correlation Matrix between Dependent and Independent Variables

5.2 Data Preparation

Data preparation is divided into three sub-processes: feature extraction, scaling, and selection.

Feature Extraction:

The given code in *Figure 5.11* implements multiple functions to extract various features from a signal shape. The `**slope_of_linear_fit**` function computes the slope of a linear fit to the given data by employing the method of least squares regression. The `**max_value_position**` function determines the index of the highest value in the given array. The `**median value**` function computes the median value of the given array, whereas the `**variance_value**` function determines the variance of the array. The `**distribution_density_measures**` function calculates the skewness and kurtosis of the supplied data. If either measure results in NaN, the function returns 0 for that measure.

```
# Signal Shape
# Function to extract feature slope of linear fit
def slope_of_linear_fit(x):
    n = len(x)
    X = np.arange(n)
    A = np.vstack([X, np.ones(n)]).T
    m, c = np.linalg.lstsq(A, x, rcond=None)[0]
    return m

# Function to extract maximum value position
def max_value_position(x):
    return np.argmax(x)
```

```
# Function to extract distribution density measures
# Function to extract the median value
def median_value(x):
    return np.median(x)

# Function to extract the variance
def variance_value(x):
    return np.var(x)

def distribution_density_measures(x):
    skewness = skew(x)
    kurtosis_val = kurtosis(x)
    # Handle cases where skewness or kurtosis are NaN
    if np.isnan(skewness):
        skewness = 0
    if np.isnan(kurtosis_val):
        kurtosis_val = 0
    return skewness, kurtosis_val
```

Figure 5.11 Feature Extraction Functions

The code presented in *Figure 5.12* establishes a function called `**extract_features**`, which analyses sensor data to get many statistical features for each sensor reading. The function computes the slope of the linear slope, the position of the maximum value, the skewness and kurtosis, the median, and the variance for each row of the sensor data. The retrieved features are subsequently stored in a list and transformed into a NumPy array. A dictionary called `**extracted_features**` is utilised to store the features for each sensor by iterating through the sensor data. The features obtained from each sensor are subsequently allocated to distinct variables, which correspond to individual sensors distinguished by their respective names.

```python
# Define a function to extract all features for each sensor reading
def extract_features(sensor_data):
    features = []
    for i in range(sensor_data.shape[0]):
        row = sensor_data[i, :]
        solf = slope_of_linear_fit(row)
        mvp = max_value_position(row)
        skewness, kurtosis_val = distribution_density_measures(row)
        median = median_value(row)
        variance = variance_value(row)
        features.append([solf, mvp, skewness, kurtosis_val, median, variance])
    return np.array(features)

# Dictionary to store the extracted features
extracted_features = {}

# Apply feature extraction to each sensor
for sensor_name, data in sensor_data.items():
    extracted_features[sensor_name] = extract_features(data)

# Assign the extracted features
ps1_features = extracted_features["ps1"]
ps2_features = extracted_features["ps2"]
ps3_features = extracted_features["ps3"]
ps4_features = extracted_features["ps4"]
ps5_features = extracted_features["ps5"]
ps6_features = extracted_features["ps6"]
eps1_features = extracted_features["eps1"]
fs1_features = extracted_features["fs1"]
fs2_features = extracted_features["fs2"]
ts1_features = extracted_features["ts1"]
ts2_features = extracted_features["ts2"]
ts3_features = extracted_features["ts3"]
ts4_features = extracted_features["ts4"]
vs1_features = extracted_features["vs1"]
ce_features = extracted_features["ce"]
cp_features = extracted_features["cp"]
se_features = extracted_features["se"]
```

Figure 5.12 Extracting Features from Sensor Data Function

The given code in *Figure 5.13* merges feature arrays from numerous sensors into a unified NumPy array and subsequently creates a thorough data frame to store all these features. The `**np. hstack**` function initially combines the different feature arrays (`ps1_features`, `ps2_features`, etc.) horizontally into a single array called `**features_combined**`. Afterwards, a dynamic list of column names for the features is created by iterating through a predetermined list of sensor names (`["PS1", "PS2", ..., "SE"]`) and adding suffixes (`_SOLF`, `_MVP`, etc.) for each type of feature associated with each sensor. Ultimately, a data-frame named `**features_df**` is generated by employing the `**pd.DataFrame**` function. The data for this data frame is derived from `**features_combined**`, while the column names are specified by `**feature_columns**`.

```python
# Combine all features into a single DataFrame
features_combined = np.hstack([
    ps1_features, ps2_features, ps3_features, ps4_features, ps5_features, ps6_features,
    eps1_features, fs1_features, fs2_features, ts1_features, ts2_features, ts3_features, ts4_features,
    vs1_features, ce_features, cp_features, se_features
])
```

```python
feature_columns = []
for sensor in ["PS1", "PS2", "PS3", "PS4", "PS5", "PS6", "EPS1", "FS1", "FS2", "TS1", "TS2", "TS3", "TS4", "VS1", "CE", "CP", "SE"]:
    feature_columns.extend([
        f"{sensor}_SOLF", f"{sensor}_MVP", f"{sensor}_Skewness", f"{sensor}_Kurtosis",
        f"{sensor}_Median", f"{sensor}_Variance"
    ])

features_df = pd.DataFrame(features_combined, columns=feature_columns)
```

Figure 5.13 Combine Sensor Features into a Structured Data-frame

The code presented in *Figure 5.14* carries out two main tasks to prepare the dataset for analysis. The initial step involves extracting the column names from the data frame `**features_df**` and subsequently storing them in a list named `**data_names**`. This list includes the names of all the features found in the `features_df`, which are used in further coding. Next, it combines the data-frame `**features_df**` with another data-frame `**df_profile**` horizontally along the columns (axis=1) to generate a new data-frame called `**df_final**`. The process of merging two data sets by combining their columns into a single data frame is then used for further analysis in feature scaling.

```python
# features_df is your DataFrame
data_names = features_df.columns.tolist()
```
✓ 0.0s

```python
df_final=pd.concat([features_df,df_profile],axis=1)
```
✓ 0.0s

Figure 5.14 Prepare and Combine Feature Data-frames

Feature Scaling:

The code presented in *Figure 5.15* normalises sensor data from a data frame and arranges it into a new data frame. A dictionary called `**normalized_data_dict**` is constructed initially to store the normalised data. It is initially empty. The `**MinMaxScaler**` object from the `**sklearn. The preprocessing**` module is created to rescale the data so that it falls between the range of 0 and 1. The code subsequently iterates through each sensor name in the `**data_names**` list, reforming the sensor data into a two-dimensional array as mandated by the scaler. The sensor data is standardised using the `**fit_transform**` function of the `**MinMaxScaler**` class, and the resultant standardised values are saved in the dictionary with their corresponding sensor names. Once all the sensors have been processed, the dictionary is transformed into a data frame called `**scaling_df**`. Finally, a duplicate of `**scaling_df**` is created to ensure that it is defragmented, resulting in a clean and continuous data frame for future use. This process helps to reorganise the underlying data storage, making it contiguous in memory.

```
# Dictionary to store normalized data
normalized_data_dict = {}

# Loop through each sensor and normalize the data to [0, 1]
for sensor_name in data_names:
    sensor_data = df_final[sensor_name].values.reshape(-1, 1)

    # Normalize the data using MinMaxScaler
    normalized_data = scaler.fit_transform(sensor_data)
    normalized_data_dict[sensor_name] = normalized_data.flatten()

# Convert the dictionary to a DataFrame
scaling_df = pd.DataFrame(normalized_data_dict)

# To get a de-fragmented frame, make a copy of the DataFrame
scaling_df = scaling_df.copy()
```

Figure 5.15 Normalize Sensor Data and Store in a Data-frame

Feature Selection:

The code displayed in *Figure 5.16* defines a variable '**traget_variables**', which is used in the feature selection analysis to select the features (`_SOLF`, `_MVP`, etc) using Pearson Correlation Analysis and Linear Discriminant Analysis.

```
# List of target variables
target_variables = ["Cooler_Condition", "Valve_Condition", "Internal_Pump_Leakage",
                    "Hydraulic_Accumulator", "Stable_Flag"]
```

Figure 5.16 List of target variables

The code in Figure 5.17 snippet detects and stores the selected features for each target variable based on their Pearson correlation coefficients. A dictionary called `**selected_features_corr_dict_pearson**` is first generated to store the selected features for each target variable. The code then iterates through each target variable in the `**target_variables**` list. The code calculates the Pearson correlation between each target variable and all features in the `**scaling_df**` data frame using the `**corrwith**` function. Features with a correlation coefficient greater than 0.1 are chosen, and their names are saved in a list. The list of chosen characteristics is subsequently placed in the dictionary using the appropriate target variable key.

```python
# Dictionary to store selected features for each target variable
selected_features_corr_dict_pearson = {}

# Loop through each target variable
for target_variable in target_variables:
    # Calculate Pearson correlation using scaling_df for both target and features
    correlation_target = scaling_df.corrwith(df_final[target_variable])

    # Select features based on correlation threshold (
    selected_features_corr = correlation_target[abs(correlation_target) > 0.1].index.tolist()

    # Store selected features in the dictionary
    selected_features_corr_dict_pearson[target_variable] = selected_features_corr
```

Figure 5.17 Pearson Correlation Analysis

The code presented in Figure 5.18 selects and stores specific features for each target variable based on their relevance scores obtained from a Linear Discriminant Analysis (LDA) model. A dictionary called `**selected_features_lda_dict**` is first built to store the selected features for each target variable. The code proceeds to iterate through each target variable in the `**target_variables**` list. The data frame `**df_final**` is divided into the features (X) and the target variable (y) for each target variable. The LDA model is created and trained using the characteristics and target variable. The importance scores for each feature are derived from the absolute values of the coefficients of the model. When dealing with a target variable that has more than two classes, the average of the absolute coefficients across all classes is utilised. Features that have an importance score higher than 0.1 are chosen and saved in a list. The list of selected features is saved in the dictionary under the key corresponding to the target variable. This creates a collection of features that are important for differentiating across classes in the LDA model.

```python
# Dictionary to store selected features for each target variable
selected_features_lda_dict = {}

# Loop through each target variable
for target_variable in target_variables:
    X = df_final.drop(columns=[target_variable])
    y = df_final[target_variable]

    # Instantiate and fit the LDA model
    lda = LinearDiscriminantAnalysis()
    lda.fit(X, y)

    # Get importance scores for each feature
    if len(lda.classes_) > 2:
        # Handle multiple classes by taking the mean of absolute coefficients across classes
        feature_importances = abs(lda.coef_).mean(axis=0)
    else:
        feature_importances = abs(lda.coef_[0])

    # Select features based on importance threshold
    selected_features_lda = [feature for feature, importance in zip(data_names, feature_importances) if importance > 0.1]

    # Store selected features in the dictionary
    selected_features_lda_dict[target_variable] = selected_features_lda
```

Figure 5.18 Linear Discriminant Analysis

5.3 Data Modelling

In the process of data modelling, the Random Forest algorithm is utilised with three distinct methods of feature selection: features selected at random, features chosen based on Pearson correlation, and features identified through Linear Discriminant Analysis (LDA). As part of the second phase of the Final Year

Project (FYP), two additional models, Catboost and Long Short-Term Memory (LSTM), will be implemented.

Initialising hyperparameters

Figure 5.19 initialises key hyperparameters for both machine learning and deep learning models. For the machine learning model, it sets the random seed `**random**` to 200 for reproducibility, the number of estimators `**n_est**` to 100 for ensemble methods, and the test size `**ts_size**` to 0.2, indicating 20% of the data will be used for testing. For the deep learning model, it defines the number of epochs `**epoch**` as 20 and the batch size `**batchSize**` as 16, determining the number of samples processed before the model's internal parameters are updated.

```
random= 200
n_est= 100
ts_size = 0.2
# Deep Learning
epoch = 20
batchSize = 16
```

Figure 5.19 Initializing hyperparameters

Random Forest

Figure 5.20 is structured to train Random Forest models for multiple target variables, evaluate their performance, and store the results in dictionaries. The '**train_models**' function begins by initialising an empty dictionary '**random_forest_models**' to store Random Forest models for each target variable. It then loops through each target variable, prepares the data with the selected features, and splits it into training and testing sets. For each target variable, a Random Forest model '**rf_model**' is initialised with 100 estimators and a random state of 200. The model is trained on the training data, and predictions are made on the test data.

Finally, the trained model and its performance metrics are stored in dictionaries. The '**random_forest_models**' dictionary holds the trained Random Forest models for each target variable, while the '**performance_metrics_dict**' dictionary stores the performance metrics, including accuracy, precision, recall, F1-score, ROC AUC, and the predicted probabilities, for each target variable. The function returns these dictionaries for further use.

```python
def train_models(target_variables, selected_features_dict, scaling_df, df_final):
    # Dictionary to store Random Forest models for each target variable
    random_forest_models = {}

    # Dictionary to store performance metrics for each target variable
    performance_metrics_dict = {}

    # Loop through each target variable
    for target_variable in target_variables:
        # Prepare the data with selected features
        selected_features = selected_features_dict[target_variable]
        X_selected = scaling_df[selected_features]
        y = df_final[target_variable]

        # Split the data into training and testing sets
        X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=ts_size, random_state=random)

        # Initialize Random Forest model
        rf_model = RandomForestClassifier(n_estimators=n_est, random_state=random)

        # Fit the model
        rf_model.fit(X_train, y_train)

        # Make predictions
        y_pred = rf_model.predict(X_test)
        #print(y_pred)
        # Evaluate the model
        accuracy = accuracy_score(y_test, y_pred)

        precision = precision_score(y_test, y_pred, average='weighted')

        recall = recall_score(y_test, y_pred, average='weighted')

        f1 = f1_score(y_test, y_pred, average='weighted')
        y_pred_prob = rf_model.predict_proba(X_test)
```

```python
#---------------------------------------------------------------------------------------------------------
# Compute ROC AUC
        y_pred_prob = rf_model.predict_proba(X_test)
        n_classes = y_pred_prob.shape[1]
        #print(n_classes)

        if len(set(y)) > 2:  # Multiclass classification
            roc_auc = roc_auc_score(y_test, y_pred_prob, multi_class='ovr')
        else:  # Binary classification
            rf_prob = y_pred_prob[:, 1]
            roc_auc = roc_auc_score(y_test, rf_prob)

#---------------------------------------------------------------------------------------------------------

        #Store the trained model and performance metrics in dictionaries
        random_forest_models[target_variable] = rf_model
        performance_metrics_dict[target_variable] = {
            'Accuracy': accuracy,
            'Precision': precision,
            'Recall': recall,
            'F1-Score': f1,
            'ROC AUC': roc_auc,
            'y_test': y_test,
            'y_pred_prob': y_pred_prob,
        }
    return random_forest_models, performance_metrics_dict
```

Figure 5.20 Random Forest (RF) Model

The first part of coding, in Figure 5.21, trains Random Forest models using all available features. It initialises '**selected_features_all_dict**' to include all feature names for each target variable. The '**train_models**' function is then called with this dictionary, along with the scaling data '**scaling_df**' and the final dataset '**df_final**'. The resulting trained models and performance metrics are stored in '**random_forest_models_all**' and '**performance_metrics_dict_all**', respectively.

The second part focuses on training models using features selected based on Pearson correlation. It calls the '**train_models**' function with '**selected_features_corr_dict_pearson**', which contains the selected features for each target variable based on Pearson correlation. The trained models and their performance metrics are stored in '**random_forest_models_pearson**' and '**performance_metrics_dict_pearson**'.

The third part involves training models using features selected by Linear Discriminant Analysis (LDA). It uses '**selected_features_lda_dict**' to provide the selected features for each target variable. The '**train_models**' function is called with this dictionary, and the trained models and performance metrics are stored in '**random_forest_models_lda**' and '**performance_metrics_dict_lda**'.

```
# Train models using all features
print("Training models using all features:")
selected_features_all_dict = {target: data_names for target in target_variables}
random_forest_models_all, performance_metrics_dict_all = train_models(
    target_variables, selected_features_all_dict, scaling_df, df_final
)

# Train models using features selected by Pearson correlation
print("Training models using features selected by Pearson correlation:")
random_forest_models_pearson, performance_metrics_dict_pearson = train_models(
    target_variables, selected_features_corr_dict_pearson, scaling_df, df_final
)

# Train models using features selected by LDA
print("Training models using features selected by LDA:")
random_forest_models_lda, performance_metrics_dict_lda= train_models(
    target_variables, selected_features_lda_dict, scaling_df, df_final)
```

Figure 5.21 Training Models (RF)

After training, the performance metrics from each method are converted into data frames: '**rf_performance_df**' for all features, '**pca_rf_performance_df**' for Pearson correlation-selected features, and '**lda_rf_performance_df**' for LDA-selected features. These data frames are then printed to show the performance metrics of the trained models.

```
rf_performance_df = pd.DataFrame(performance_metrics_dict_all).T
pca_rf_performance_df = pd.DataFrame(performance_metrics_dict_pearson).T
lda_rf_performance_df = pd.DataFrame(performance_metrics_dict_lda).T
```

Figure 5.22 Converting Performance Metrics Results into Data Frame

For the training, converting the performance metrics to data frames is the same for Catboost and LSTM. The only difference is the model algorithm. For example, in Figure 5.21, the model name is train_models, while for Catboost, it is cb_train_models, and for LSTM, it is lstm_train_models.

Catboost

Figure 5.23 defines a function to train Catboost models for multiple target variables, evaluate their performance, and store the results in dictionaries. The '**cb_train_models**' function initialises an empty dictionary '**catboost_models**' to store Catboost models for each target variable. It iterates over each target variable, prepares the data with the selected features, and splits it into training and testing sets. A Catboost model '**catboost_model**' is initialised with the specified number of iterations and a random seed for reproducibility. The model is then trained on the training data, and predictions are made on the test data.

The trained model and its performance metrics are stored in dictionaries. The '**catboost_models**' dictionary holds the trained Catboost models for each target variable, while the '**performance_metrics_dict**' dictionary stores the performance metrics, including accuracy, precision, recall, F1-score, ROC AUC, and the predicted probabilities, for each target variable. The function returns these dictionaries for further use.

```
def cb_train_models(target_variables, selected_features_dict, scaling_df, df_final):
    # Dictionary to store CatBoost models for each target variable
    catboost_models = {}

    # Dictionary to store performance metrics for each target variable
    performance_metrics_dict = {}

    # Loop through each target variable
    for target_variable in target_variables:
        # Prepare the data with selected features
        selected_features = selected_features_dict[target_variable]
        X_selected = scaling_df[selected_features]
        y = df_final[target_variable]

        # Split the data into training and testing sets
        X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=ts_size, random_state=random)

        # Initialize CatBoost model
        catboost_model = CatBoostClassifier(iterations=n_est, random_seed=random, silent=True)

        # Fit the model
        catboost_model.fit(X_train, y_train)

        # Make predictions
        y_pred = catboost_model.predict(X_test)

        # Evaluate the model
        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred, average='weighted')
        recall = recall_score(y_test, y_pred, average='weighted')
        f1 = f1_score(y_test, y_pred, average='weighted')
```

```
#-----------------------------------------------------------------------------------------------------------
# Compute ROC AUC
        y_pred_prob = catboost_model.predict_proba(X_test)

        if len(set(y)) > 2:  # Multiclass classification
            roc_auc = roc_auc_score(y_test, y_pred_prob, multi_class='ovr')
        else:  # Binary classification
            y_prob = y_pred_prob[:, 1]
            roc_auc = roc_auc_score(y_test, y_prob)

#-----------------------------------------------------------------------------------------------------------

        # Store the trained model and performance metrics in dictionaries
        catboost_models[target_variable] = catboost_model
        performance_metrics_dict[target_variable] = {
            'Accuracy': accuracy,
            'Precision': precision,
            'Recall': recall,
            'F1-Score': f1,
            'ROC AUC': roc_auc,
            'y_test': y_test,
            'y_pred_prob': y_pred_prob,
        }
    return catboost_models, performance_metrics_dict
```

Figure 5.23 Catboost Model

LSTM

The '**lstm_train_models**' function initialises an empty dictionary '**lstm_models**' to store LSTM models for each target variable. It iterates over each target variable, preparing the data with the selected features and splitting it into training and testing sets. The features are scaled, and the data is reshaped to 3D for LSTM input. The target variable is converted to a categorical format. An LSTM model is created using a custom '**lstm_model**' function, trained on the training data, and used to make predictions on the test data.

The trained model and its performance metrics are stored in dictionaries. The '**lstm_models**' dictionary holds the trained LSTM models for each target variable, while the '**performance_metrics_dict**' dictionary stores the performance metrics, including accuracy, precision, recall, F1-score, ROC AUC, and the predicted probabilities, for each target variable. The function returns these dictionaries for further use.

```python
def lstm_train_models(target_variables, selected_features_dict, scaling_df, df_final):
    # Dictionary to store LSTM models for each target variable
    lstm_models = {}

    # Dictionary to store performance metrics for each target variable
    performance_metrics_dict = {}

    # Loop through each target variable
    for target_variable in target_variables:
        # Prepare the data with selected features
        selected_features = selected_features_dict[target_variable]
        X_selected = scaling_df[selected_features]
        y = df_final[target_variable]

        # Split the data into training and testing sets
        X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=ts_size, random_state=random)

        # Scale the features
        scaler = StandardScaler()
        X_train = scaler.fit_transform(X_train)
        X_test = scaler.transform(X_test)

        # Reshape the data to 3D for LSTM [samples, time steps, features]
        X_train = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
        X_test = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

        # Convert the target variable to categorical
        y_train = to_categorical(y_train)
        y_test = to_categorical(y_test)

        # Create LSTM model
        model = lstm_model((X_train.shape[1], X_train.shape[2]), y_train.shape[1])

        # Train the model
        model.fit(X_train, y_train, epochs=epoch, batch_size=batchSize, verbose=1)
```

```python
        # Make predictions
        y_pred = (model.predict(X_test) > 0.5).astype(int)
        y_pred_classes = np.argmax(y_pred, axis=1)
        y_test_classes = np.argmax(y_test, axis=1)

        # Evaluate the model
        accuracy = accuracy_score(y_test_classes, y_pred_classes)
        precision = precision_score(y_test_classes, y_pred_classes, average='weighted', zero_division= 0)
        recall = recall_score(y_test_classes, y_pred_classes, average='weighted', zero_division= 0)
        f1 = f1_score(y_test_classes, y_pred_classes, average='weighted', zero_division= 0)

#-----------------------------------------------------------------------------------------------------------
#CALCULATE ROC AUC
        y_prob = model.predict(X_test)
        roc_auc = roc_auc_score(y_test, y_prob, multi_class="ovr", average='micro')
#-----------------------------------------------------------------------------------------------------------

        # Store the trained model and performance metrics in dictionaries
        lstm_models[target_variable] = model
        performance_metrics_dict[target_variable] = {
            'Accuracy': accuracy,
            'Precision': precision,
            'Recall': recall,
            'F1-Score': f1,
            'ROC AUC': roc_auc,
            'y_test': y_test,
            'y_pred_prob': y_prob,
        }

    return lstm_models, performance_metrics_dict
```

Figure 5.24 LSTM Model

5.4 Data Evaluation

Figures 5.25 to 5.27 illustrate the evaluation metrics for different models: Random Forest (RF), Catboost, and Long Short-Term Memory (LSTM) neural network. For each model, the metrics calculated include accuracy, precision, recall, and F1-score, all using a weighted average to address class imbalances. Additionally, the ROC AUC score is computed to evaluate the models' performance. For multiclass classification, the roc_auc_score function uses a one-vs-rest (ovr) approach with a micro average method, while for binary classification, it focuses on the probability of the positive class. Specifically, the RF and Catboost models handle both binary and multiclass scenarios, whereas the LSTM model is tailored for multiclass classification.

```
        # Evaluate the model
        accuracy = accuracy_score(y_test, y_pred)

        precision = precision_score(y_test, y_pred, average='weighted')

        recall = recall_score(y_test, y_pred, average='weighted')

        f1 = f1_score(y_test, y_pred, average='weighted')
        y_pred_prob = rf_model.predict_proba(X_test)
#-------------------------------------------------------------------------------------------------
# Compute ROC AUC
        y_pred_prob = rf_model.predict_proba(X_test)
        n_classes = y_pred_prob.shape[1]
        #print(n_classes)

        if len(set(y)) > 2:  # Multiclass classification
            roc_auc = roc_auc_score(y_test, y_pred_prob, multi_class='ovr', average='micro')
        else:  # Binary classification
            rf_prob = y_pred_prob[:, 1]
            roc_auc = roc_auc_score(y_test, rf_prob)

#-------------------------------------------------------------------------------------------------
```

Figure 5.25 Evaluation Metrics RF

```
        # Evaluate the model
        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred, average='weighted')
        recall = recall_score(y_test, y_pred, average='weighted')
        f1 = f1_score(y_test, y_pred, average='weighted')

#-------------------------------------------------------------------------------------------------
# Compute ROC AUC
        y_pred_prob = catboost_model.predict_proba(X_test)

        if len(set(y)) > 2:  # Multiclass classification
            roc_auc = roc_auc_score(y_test, y_pred_prob, multi_class='ovr', average='micro')
        else:  # Binary classification
            y_prob = y_pred_prob[:, 1]
            roc_auc = roc_auc_score(y_test, y_prob)

#-------------------------------------------------------------------------------------------------
```

Figure 5.26 Evaluation Metrics Catboost

```
        # Evaluate the model
        accuracy = accuracy_score(y_test_classes, y_pred_classes)
        precision = precision_score(y_test_classes, y_pred_classes, average='weighted', zero_division= 0)
        recall = recall_score(y_test_classes, y_pred_classes, average='weighted', zero_division= 0)
        f1 = f1_score(y_test_classes, y_pred_classes, average='weighted', zero_division= 0)

#-----------------------------------------------------------------------------------------------------------
#CALCULATE ROC AUC
        y_prob = model.predict(X_test)
        roc_auc = roc_auc_score(y_test, y_prob, multi_class="ovr", average='micro')
#-----------------------------------------------------------------------------------------------------------
```

Figure 5.27 Evaluation Metrics LSTM

The function '**plot_roc_curves**' in Figure 5.28 is designed to plot ROC curves for machine learning models. It accepts a dictionary of performance metrics where each key represents a target variable, and the corresponding value contains the true labels '**y_test**' and predicted probabilities '**y_pred_prob**'. The function handles both binary and multiclass classification. For binary classification, it calculates the false positive rate FPR and true positive rate TPR for the positive class and plots the ROC curve along with the area under the curve (AUC). For multiclass classification, it binarises the true labels and plots the ROC curve for each class, including the AUC for each one. The function ensures robustness by checking for classes with only one unique label, which it skips to avoid plotting errors.

```
def plot_roc_curves(performance_metrics_dict):
    for target_variable, metrics in performance_metrics_dict.items():
        y_test = metrics['y_test']
        y_pred_prob = metrics['y_pred_prob']

        # Ensure y_test is a numpy array
        y_test = np.array(y_test)
        n_classes = y_pred_prob.shape[1]

        if n_classes > 2:  # Multiclass classification
            y_test_bin = label_binarize(y_test, classes=np.unique(y_test))

            plt.figure(figsize=(10, 8))
            for i in range(n_classes):
                if len(np.unique(y_test_bin[:, i])) == 1:  # Check if there's only one class
                    print(f"Class {i} in {target_variable} has only one class in y_test. Skipping.")
                    continue

                fpr, tpr, _ = roc_curve(y_test_bin[:, i], y_pred_prob[:, i])
                roc_auc = auc(fpr, tpr)
                plt.plot(fpr, tpr, label=f'Class {i} (area = {roc_auc:.7f})')

            plt.plot([0, 1], [0, 1], 'k--', lw=2)
            plt.xlim([0.0, 1.0])
            plt.ylim([0.0, 1.05])
            plt.xlabel('False Positive Rate')
            plt.ylabel('True Positive Rate')
            plt.title(f'Receiver Operating Characteristic (ROC) - Multiclass for {target_variable}')
            plt.legend(loc="lower right")
            plt.show()

        else:  # Binary classification
            if len(np.unique(y_test)) == 1:  # Check if there's only one class
                print(f"Binary classification in {target_variable} has only one class in y_test. Skipping.")
                continue

            fpr, tpr, _ = roc_curve(y_test, y_pred_prob[:, 1])
            roc_auc = auc(fpr, tpr)

            plt.figure(figsize=(10, 8))
            plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.7f})')
            plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
            plt.xlim([0.0, 1.0])
            plt.ylim([0.0, 1.05])
            plt.xlabel('False Positive Rate')
            plt.ylabel('True Positive Rate')
            plt.title(f'Receiver Operating Characteristic (ROC) - Binary for {target_variable}')
            plt.legend(loc="lower right")
            plt.show()
```

Figure 5.28 Function to Plot Roc for Machine Learning

In contrast, the function '**plot_roc_curvesLSTM**' in Figure 5.29 is tailored for deep learning models, specifically those using Long Short-Term Memory (LSTM) networks. This function also accepts a dictionary of performance metrics with true labels and predicted probabilities. Similar to the machine learning version, it handles both binary and multiclass classifications. For multiclass classification, it ensures that the true labels are binarised correctly and plots the ROC curves for each class, including the AUC. For binary classification, it flattens the true labels and predicted probabilities before calculating the FPR and TPR to plot the ROC curve and AUC. The function is designed to accommodate the specific requirements of deep learning models, such as handling multidimensional arrays and ensuring the correct format for the input data.

```python
def plot_roc_curvesLSTM(performance_metrics_dict):
    for target_variable, metrics in performance_metrics_dict.items():
        y_test = metrics['y_test']
        y_pred_prob = metrics['y_pred_prob']

        # Ensure y_test is a numpy array
        #y_test = np.array(y_test)
        n_classes = y_pred_prob.shape[1]

        if n_classes > 2:  # Multiclass classification
            # Ensure y_test is binarized for multiclass
            if y_test.ndim == 1 or y_test.shape[1] != n_classes:
                y_test_bin = label_binarize(y_test, classes=np.arange(n_classes))
            else:
                y_test_bin = y_test

            plt.figure(figsize=(10, 8))
            for i in range(n_classes):
                if len(np.unique(y_test_bin[:, i])) == 1:  # Check if there's only one class
                    #print(f"Class {i} in {target_variable} has only one class in y_test. Skipping.")
                    continue

                fpr, tpr, _ = roc_curve(y_test_bin[:, i], y_pred_prob[:, i])
                roc_auc = auc(fpr, tpr)
                plt.plot(fpr, tpr, label=f'Class {i} (area = {roc_auc:.2f})')

            plt.plot([0, 1], [0, 1], 'k--', lw=2)
            plt.xlim([0.0, 1.0])
            plt.ylim([0.0, 1.05])
            plt.xlabel('False Positive Rate')
            plt.ylabel('True Positive Rate')
            plt.title(f'Receiver Operating Characteristic (ROC) - Multiclass for {target_variable}')
            plt.legend(loc="lower right")
            plt.show()

        else:  # Binary classification
            if len(np.unique(y_test)) == 1:  # Check if there's only one class
                print(f"Binary classification in {target_variable} has only one class in y_test. Skipping.")
                continue

            # Flatten y_test and get probabilities for the positive class
            y_test_flat =  y_test[:, 1]
            y_pred_prob_flat = y_pred_prob[:, 1]

            fpr, tpr, _ = roc_curve(y_test_flat, y_pred_prob_flat)
            roc_auc = auc(fpr, tpr)

            plt.figure(figsize=(10, 8))
            plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
            plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
            plt.xlim([0.0, 1.0])
            plt.ylim([0.0, 1.05])
            plt.xlabel('False Positive Rate')
            plt.ylabel('True Positive Rate')
            plt.title(f'Receiver Operating Characteristic (ROC) - Binary for {target_variable}')
            plt.legend(loc="lower right")
            plt.show()
```

Figure 5.29 Function to Plot Roc for Deep Learning

As shown in Figure 5.30, to display the plotting charts, you need to call the function '**plot_roc_curves**' or '**plot_roc_curvesLSTM**' for plotting the roc graph for deep learning and pass the appropriate performance dictionaries. First, call the function with '**performance_metrics_dict_all**' to plot the ROC curves for all features using a Random Forest model. The same can be done for other algorithm models. Next, call the function '**performance_metrics_dict_pearson**' to plot the ROC curves for features selected by Pearson correlation. Finally, call the function '**performance_metrics_dict_lda**' to plot the ROC curves for features selected by Linear Discriminant Analysis (LDA). Each call will generate the respective ROC plots, organised and labelled for clarity with printed separators.

```
print('''
#--------------------------------------------------------------------------------------------------------------------------------
# All Features / Random Forest
#--------------------------------------------------------------------------------------------------------------------------------''')
plot_roc_curves(performance_metrics_dict_all)
print('''
#--------------------------------------------------------------------------------------------------------------------------------
# Features selected by Pearson / Random Forest
#--------------------------------------------------------------------------------------------------------------------------------''')
plot_roc_curves(performance_metrics_dict_pearson)
print('''
#--------------------------------------------------------------------------------------------------------------------------------
# Features selected by LDA / Random Forest
#--------------------------------------------------------------------------------------------------------------------------------''')
plot_roc_curves(performance_metrics_dict_lda)
```
✓ 2.1s

Figure 5.30 Function to Call for Plotting

Chapter 6: Results and Discussion

In this section, we test the machine and deep learning models, including hyperparameter tuning, to achieve the best accuracy. The hyperparameter tuning will focus on the number of estimators and batch size. The number of estimators refers to the base models in ensemble methods like Random Forests and Gradient Boosting Machines. Batch size refers to the number of training samples used in one iteration of training. Here is a table summarising the hyperparameter values we will test:

Table 6.1 Hyperparameter Tuning

Phase	Number of Estimators (ML)	Batch Size (DL)
Phase 1	100	16
Phase 2	200	32
Phase 3	300	48

6.1 Phase 1 of Testing

6.1.1 Random Forest (100 of Estimators)

- **No Feature Selection:**

Findings for the conditioning components are as follows: The Cooler component achieved perfect scores with accuracy, precision, recall, and an F1-score of 1.0 across the board, and the ROC curve also shows an area under the curve (AUC) of 1.0, indicating flawless classification performance. The Valve, Internal Pump Leakage, and Accumulator components also demonstrated high performance with accuracy, precision, recall, and F1-scores all above 0.99, and their ROC curves show AUC values very close to 1.0, reflecting excellent classification ability. The Stable Flag component had slightly lower, but still impressive, scores with accuracy, precision, recall, and F1-score around 0.98 and an AUC of 0.997, which indicates strong predictive performance but with slight room for improvement compared to the other components.

Table 6.2: Random Forest 100 Estimators no Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.993347	0.993197	0.993217	0.999956
Internal Pump Leakage	0.993197	0.993265	0.993197	0.993211
Accumulator	0.99093	0.990962	0.99093	0.990932
Stable Flag	0.984127	0.984497	0.984127	0.984028

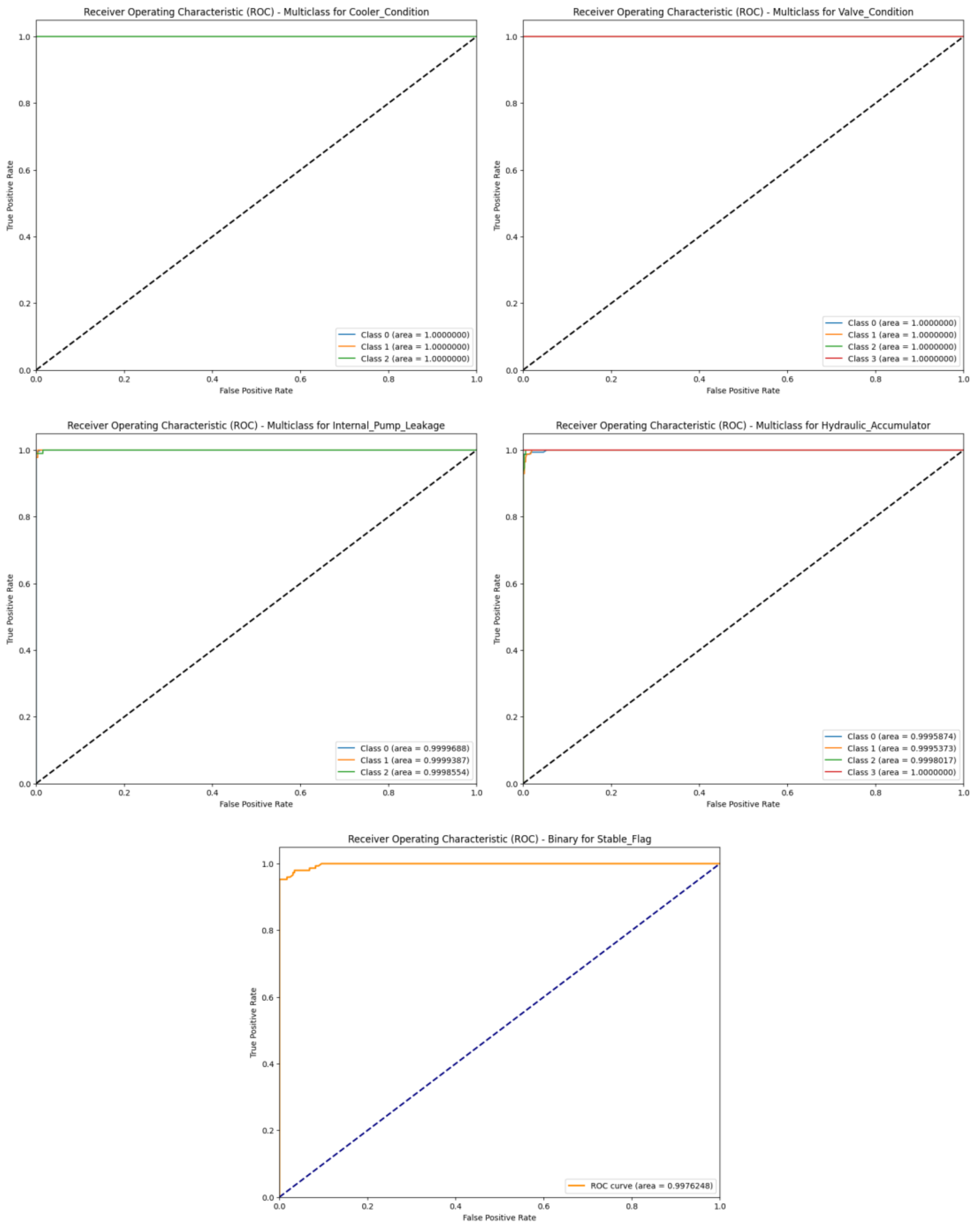


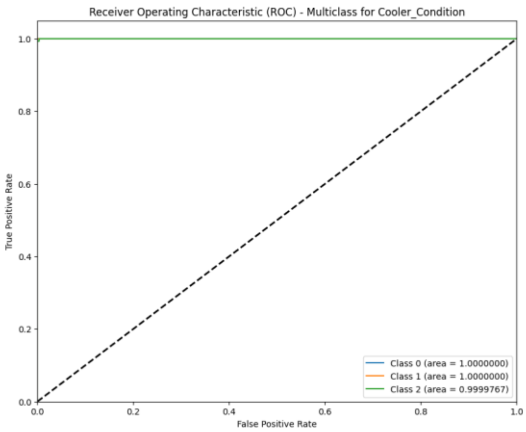
Figure 6.1 ROC AUC All Features (RF) – Phase 1

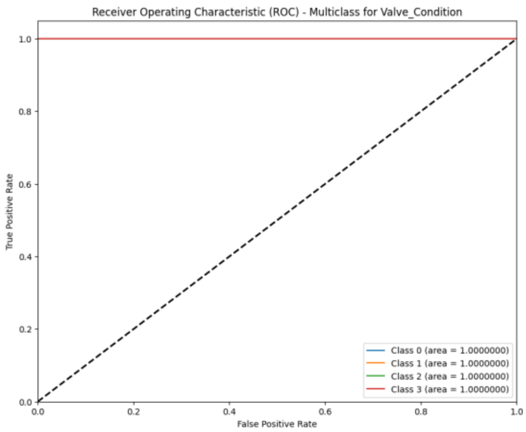
- **Pearson Correlation Analysis:**

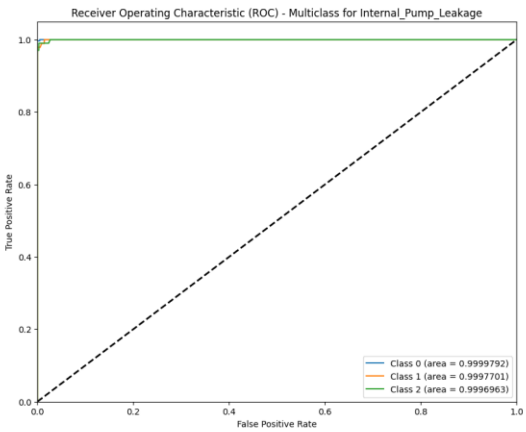
The analysis of the conditioning components using Pearson correlation analysis reveals that the Cooler component achieved an accuracy, precision, recall, and F1-score of approximately 0.998, indicating near-perfect performance. The Valve component followed closely with scores around 0.995 for all metrics, demonstrating excellent classification capability. The Internal Pump Leakage component showed strong results as well, with metrics around 0.989. The Accumulator component had slightly lower but still impressive scores around 0.982, while the Stable Flag component performed robustly with metrics near 0.980. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, highlighting the models' strong predictive performance.

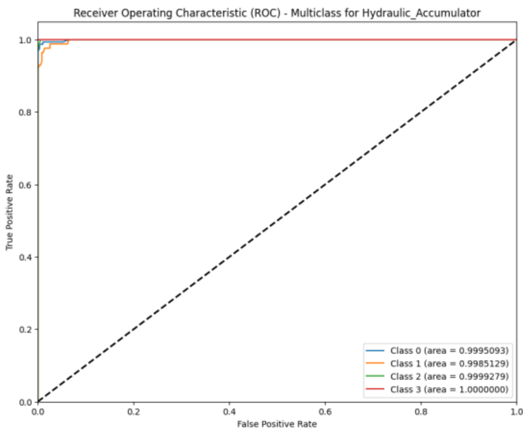
Table 6.3: Random Forest 100 Estimators PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	0.997732	0.997748	0.997732	0.997732
Valve	0.995465	0.995584	0.995465	0.995485
Internal Pump Leakage	0.988662	0.988897	0.988662	0.988717
Accumulator	0.981859	0.982176	0.981859	0.981829
Stable Flag	0.979592	0.9797	0.979592	0.979503









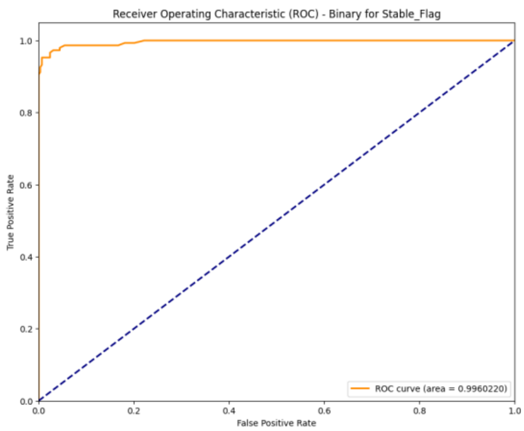


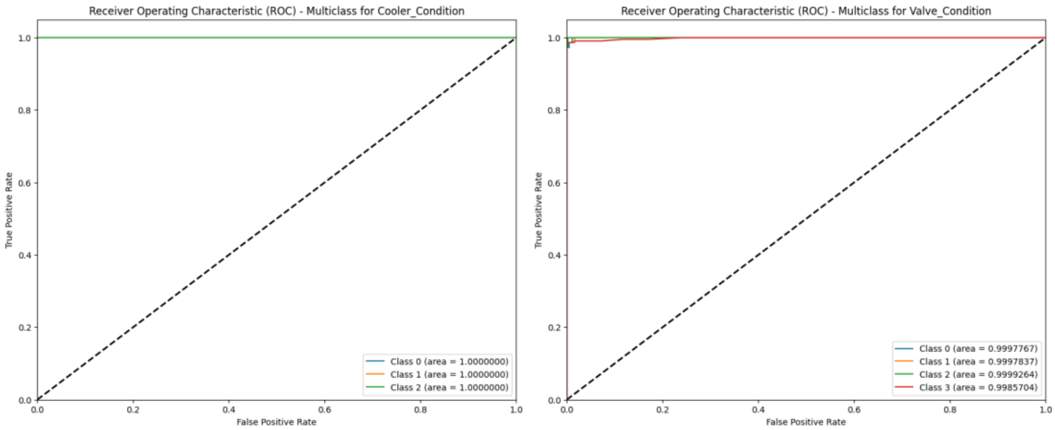
Figure 6.2 ROC AUC PCA (RF) – Phase 1

- **Linear Discriminant Analysis:**

The analysis of the conditioning components using Linear Discriminant Analysis (LDA) reveals that the Cooler component achieved an accuracy, precision, recall, and F1-score of approximately 0.998, indicating near-perfect performance. The Valve component had a lower but still high performance, with scores around 0.989 for all metrics. The Internal Pump Leakage component showed similar results with metrics around 0.989. The Accumulator component had slightly higher scores with accuracy, precision, recall, and F1-score around 0.991, while the Stable Flag component showed robust performance with metrics around 0.984. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.4: Random Forest 100 Estimators LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	0.997732	0.997748	0.997732	0.997732
Valve	0.988662	0.988843	0.988662	0.988692
Internal Pump Leakage	0.988662	0.988841	0.988662	0.988703
Accumulator	0.99093	0.990975	0.99093	0.990927
Stable Flag	0.984127	0.984497	0.984127	0.984028



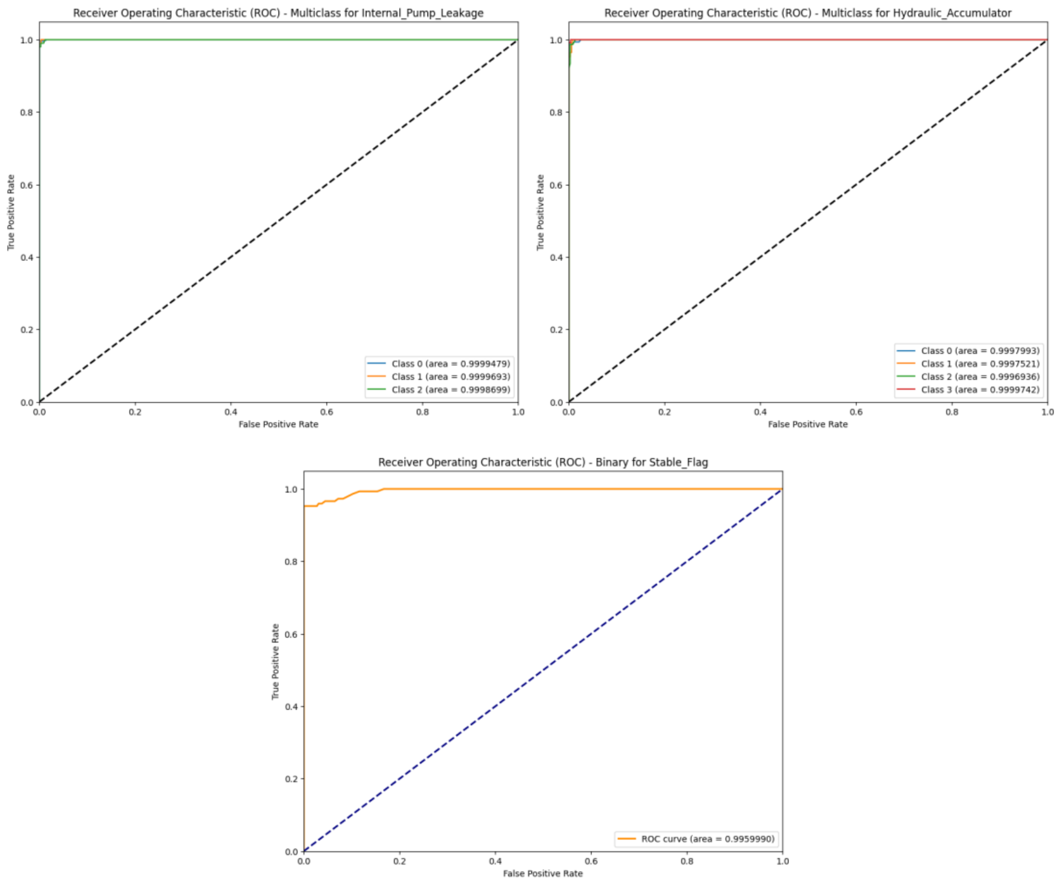


Figure 6.3 ROC AUC LDA (RF) – Phase 1

6.1.2 Catboost (100 of Estimators)

- **No Feature Selection:**

The analysis of the conditioning components using Catboost with no feature selection demonstrates exceptional performance. The Cooler and Valve components achieved perfect scores with accuracy, precision, recall, and an F1 score of 1.0. The Internal Pump Leakage component also performed admirably with metrics around 0.989. The Accumulator component had slightly lower but still high scores with accuracy, precision, recall, and F1-score around 0.977. The Stable Flag component showed robust performance with metrics around 0.991. The ROC curves for these components further support these findings, showing AUC values of 1.0 for the Cooler and Valve conditions and near-perfect AUC values close to 1.0 for the Internal Pump Leakage, Accumulator, and Stable Flag components, indicating strong predictive performance across all components.

Table 6.5: Catboost 100 Estimators No Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0

Valve	1.0	1.0	1.0	1.0
Internal Pump Leakage	0.988662	0.988793	0.988662	0.988697
Accumulator	0.977324	0.977615	0.977324	0.977283
Stable Flag	0.99093	0.991052	0.99093	0.990898

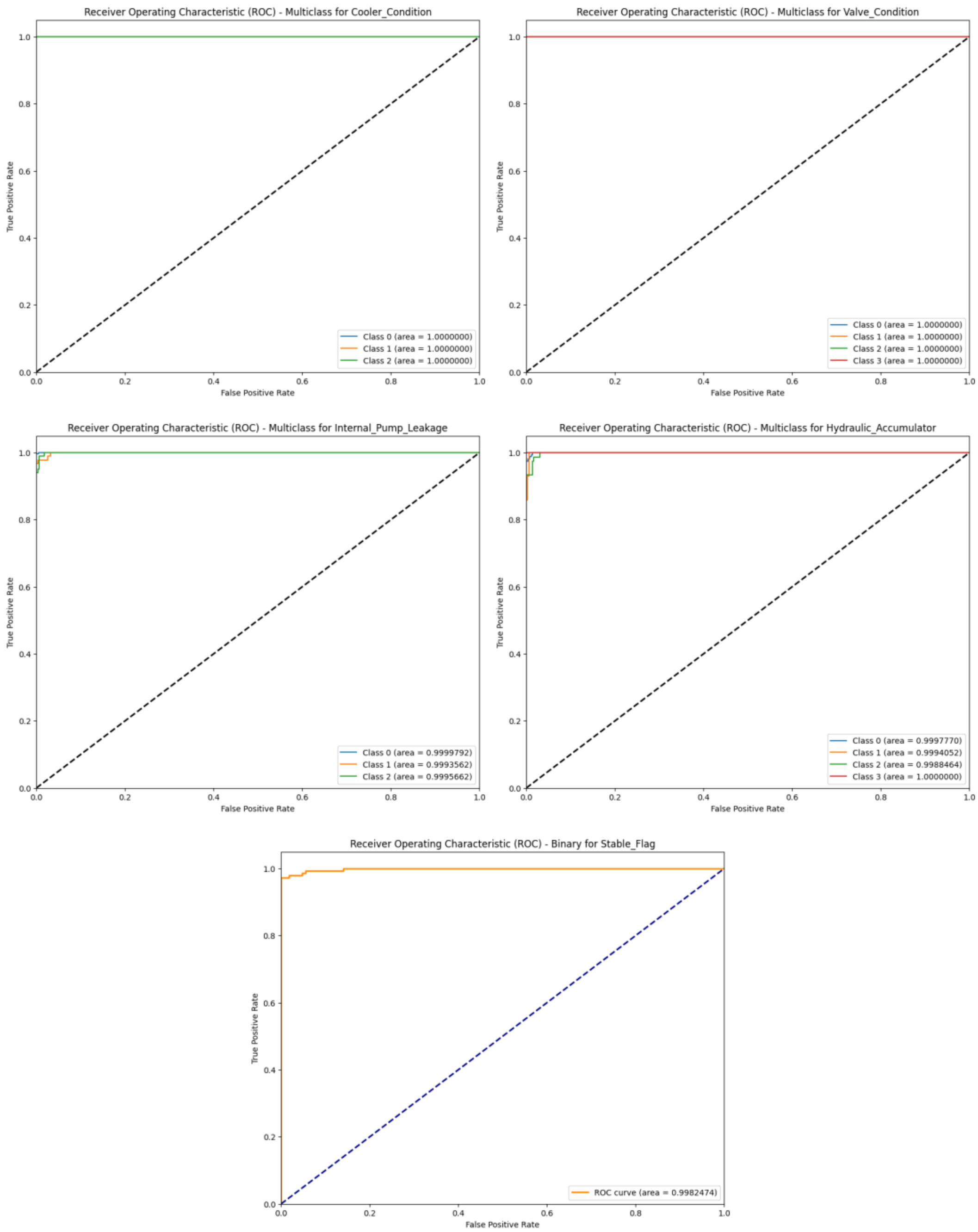


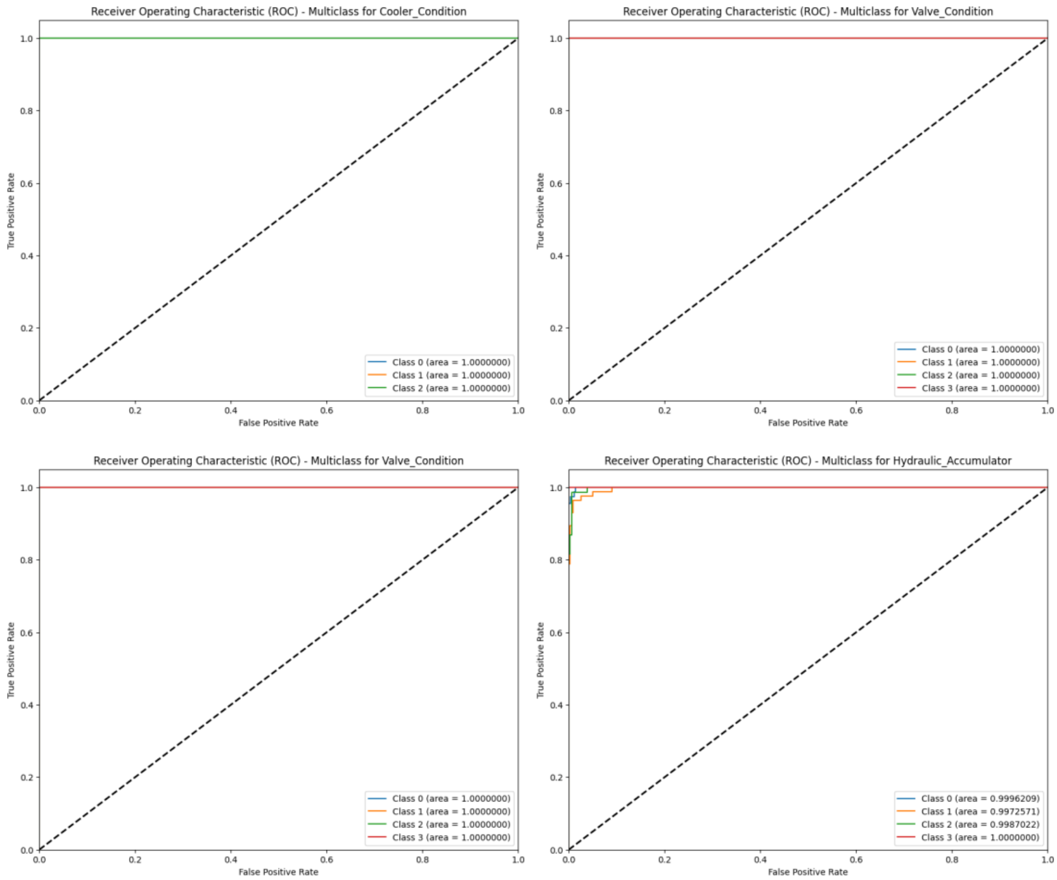
Figure 6.4 ROC AUC All Features (Catboost) – Phase 1

- **Pearson Correlation Analysis:**

The analysis of the conditioning components using Catboost with Pearson correlation analysis demonstrates exceptional performance. The Cooler and Valve components achieved perfect scores with accuracy, precision, recall, and an F1 score of 1.0. The Internal Pump Leakage component also performed admirably with metrics around 0.988662. The Accumulator component had slightly lower but still high scores with accuracy, precision, recall, and F1-score around 0.979592. The Stable Flag component showed robust performance with metrics around 0.984127. The ROC curves for these components further support these findings, showing AUC values of 1.0 for the Cooler and Valve conditions and near-perfect AUC values close to 1.0 for the Internal Pump Leakage, Accumulator, and Stable Flag components, indicating strong predictive performance across all components.

Table 6.6: Catboost 100 Estimators PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	1.0	1.0	1.0	1.0
Internal Pump Leakage	0.988662	0.988778	0.988662	0.988699
Accumulator	0.979592	0.979651	0.979592	0.979395
Stable Flag	0.984127	0.984112	0.984127	0.984114



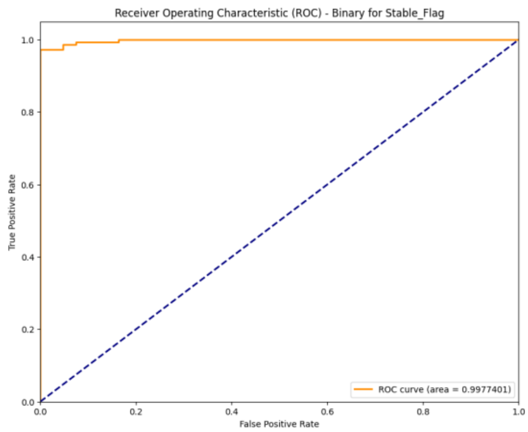


Figure 6.5 ROC AUC PCA (Catboost) – Phase 1

- **Linear Discriminant Analysis:**

The analysis of the conditioning components using Linear Discriminant Analysis (LDA) shows excellent performance. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component displayed high performance with an accuracy of 0.99093 and similar values for precision, recall, and F1-score. The Internal Pump Leakage component also performed well, with metrics around 0.986. The Accumulator component showed robust performance with accuracy, precision, recall, and an F1 score of around 0.984. The Stable Flag component had a slightly lower but still good performance, with metrics around 0.977. The ROC curves for these components support these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.7: Catboost 100 Estimators LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.99093	0.991228	0.99093	0.990975
Internal Pump Leakage	0.986395	0.986597	0.986395	0.986448
Accumulator	0.984127	0.984315	0.984127	0.984049
Stable Flag	0.977324	0.977497	0.977324	0.977205

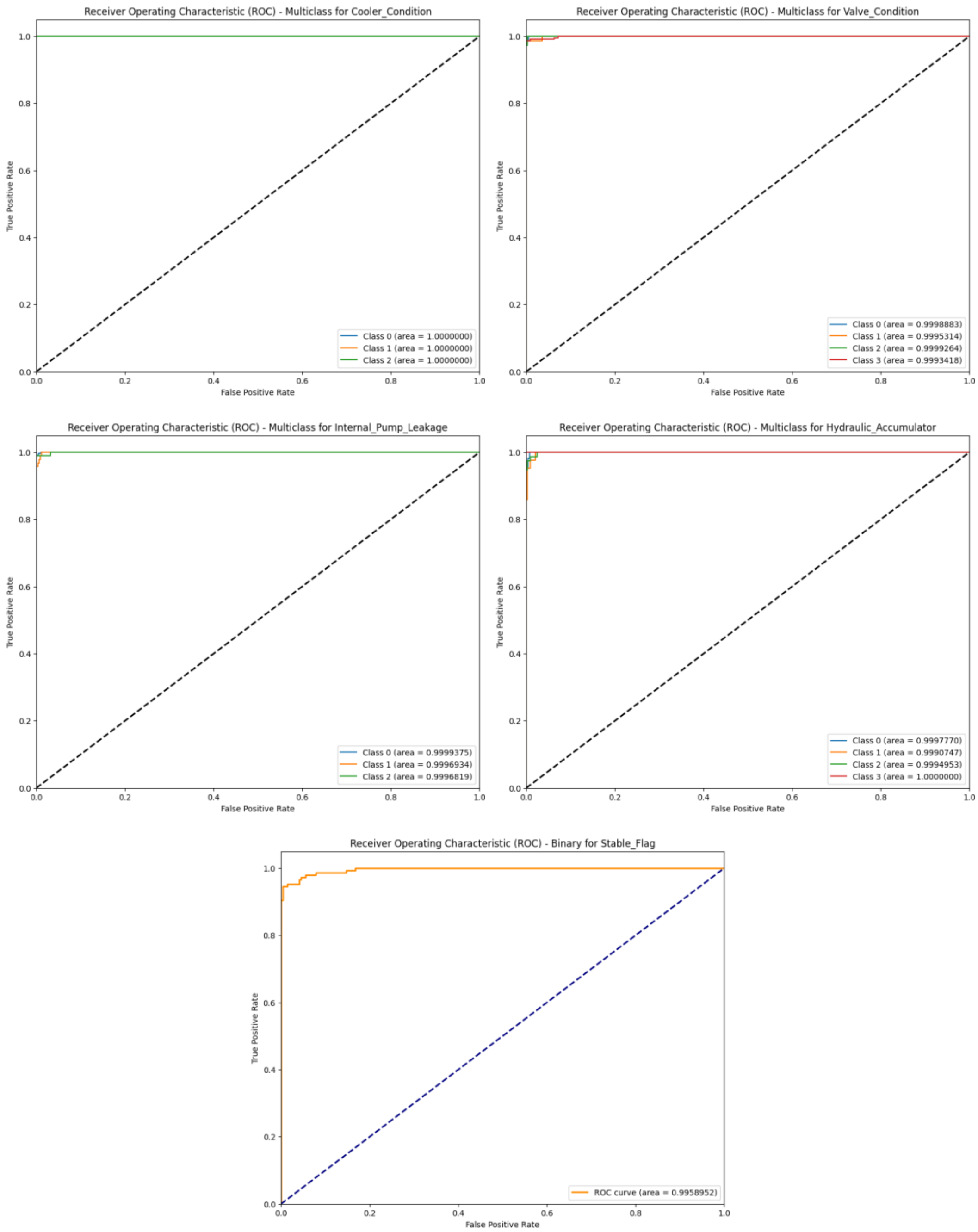


Figure 6.6 ROC AUC LDA (Catboost) – Phase 1

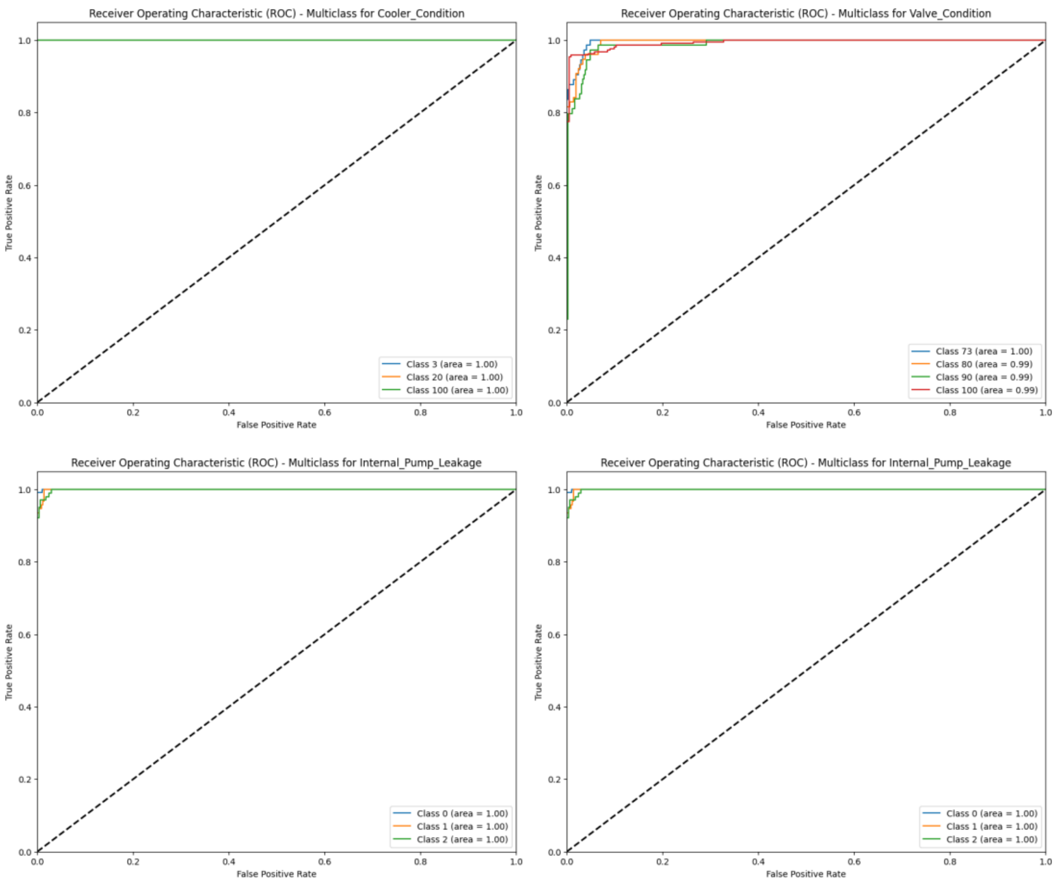
6.1.3 LSTM (16 Batch Size)

- **No Feature Selection:**

The analysis of the conditioning components using an LSTM model without feature selection demonstrates varied performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component had lower performance compared to other methods, with an accuracy of 0.895692 and corresponding precision, recall, and F1-score values around 0.923771. The Internal Pump Leakage component showed high performance with metrics around 0.979. The Accumulator component had slightly better scores with an accuracy of 0.909297 and corresponding precision, recall, and F1-score values around 0.91458. The Stable Flag component exhibited strong performance with an accuracy of 0.956916 and similar values for other metrics. The ROC curves for these components support these findings, showing high areas under the curve (AUC) values, close to 1.0 for the Cooler and Internal Pump Leakage components, and slightly lower but still high AUC values for the Valve, Accumulator, and Stable Flag components, indicating good predictive performance overall.

Table 6.7: LSTM 16 Batch Size No Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.895692	0.957366	0.895692	0.923771
Internal Pump Leakage	0.979592	0.980031	0.979592	0.979718
Accumulator	0.909297	0.921347	0.909297	0.91458
Stable Flag	0.956916	0.956998	0.956916	0.956952



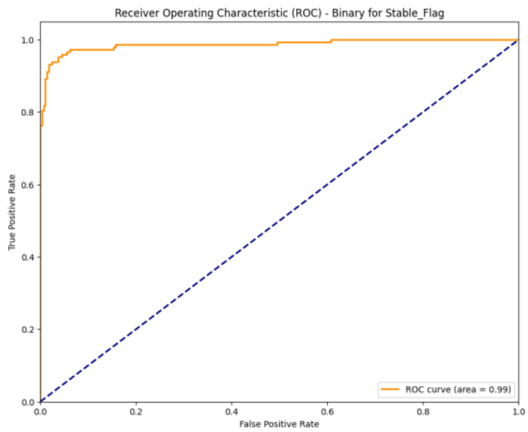


Figure 6.7 ROC AUC All Features (LSTM) – Phase 1

- **Pearson Correlation Analysis:**

The analysis of the conditioning components using Pearson Correlation Analysis demonstrates high performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component showed strong performance with an accuracy of 0.979022 and high precision, recall, and F1-score values around 0.999888. The Internal Pump Leakage component also performed well, with metrics around 0.959. The Accumulator component had slightly lower scores with an accuracy of 0.861678 and corresponding precision, recall, and F1-score values around 0.882659. The Stable Flag component exhibited robust performance with an accuracy of 0.956916 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.8: LSTM 16 Batch Size PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.979022	0.902494	0.937153	0.999888
Internal Pump Leakage	0.959184	0.965069	0.959184	0.959414
Accumulator	0.861678	0.907174	0.861678	0.882659
Stable Flag	0.956916	0.95753	0.956916	0.957089

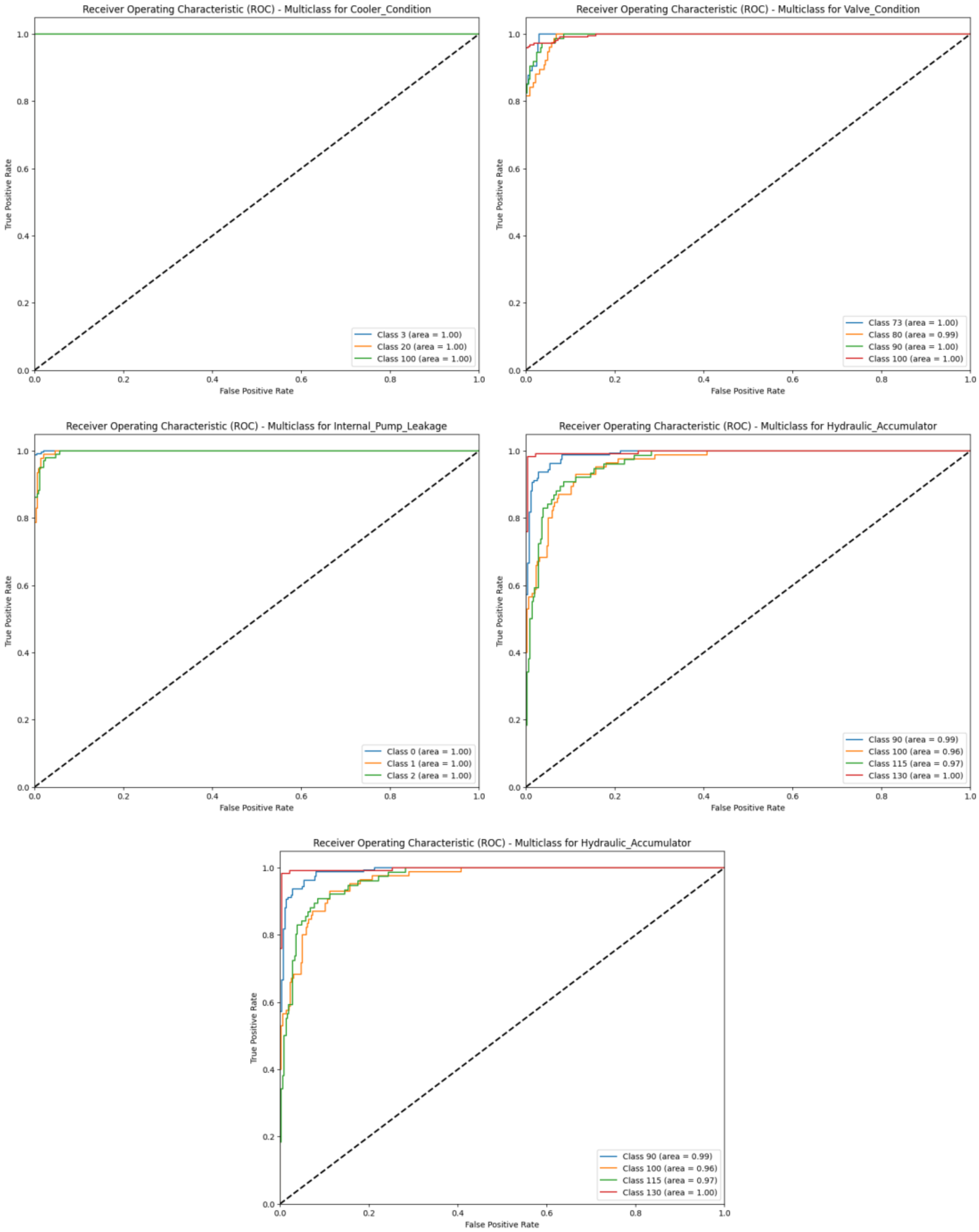


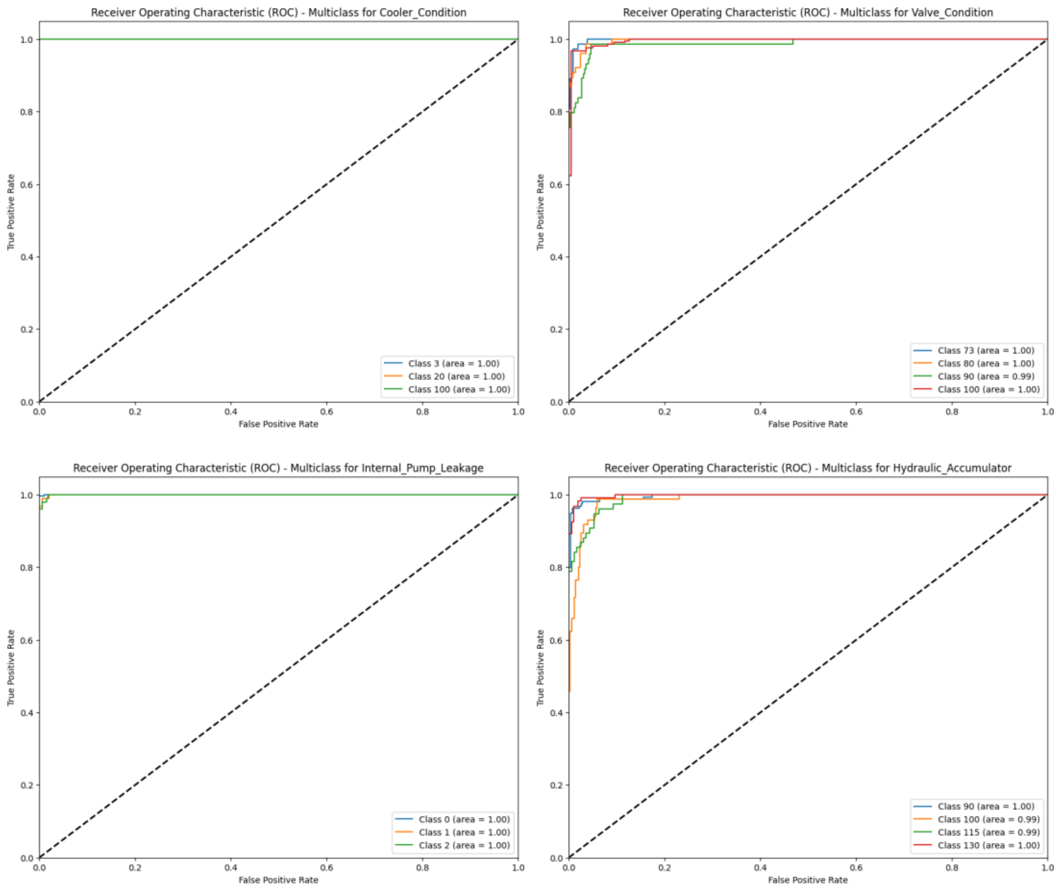
Figure 6.8 ROC AUC PCA (LSTM) – Phase 1

- **Linear Discriminant Analysis:**

The analysis of the conditioning components using Linear Discriminant Analysis (LDA) shows strong performance across various components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited high performance with an accuracy of 0.9161 and similar values for precision, recall, and F1-score around 0.936463. The Internal Pump Leakage component also showed excellent performance with metrics around 0.988662. The Accumulator component demonstrated robust performance with an accuracy of 0.931973 and corresponding precision, recall, and F1-score values around 0.933149. The Stable Flag component had slightly better scores with an accuracy of 0.970522 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.9: LSTM 16 Batch Size LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.9161	0.964052	0.9161	0.936463
Internal Pump Leakage	0.988662	0.988776	0.988662	0.988698
Accumulator	0.931973	0.934739	0.931973	0.933149
Stable Flag	0.970522	0.970713	0.970522	0.970339



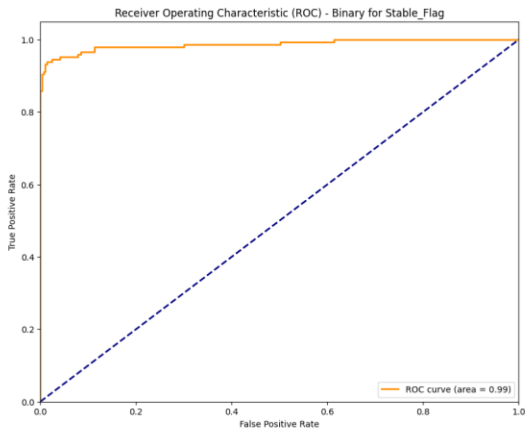


Figure 6.9 ROC AUC LDA (LSTM) – Phase 1

6.2 Phase Two of Testing

6.2.1 Random Forest (200 of Estimators)

- **No Feature Selection:**

The analysis of the conditioning components using a Random Forest model with 200 estimators and no feature selection reveals strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited high performance with an accuracy of 0.993197 and corresponding precision, recall, and F1-score values around 0.993217. The Internal Pump Leakage component also showed excellent performance with metrics around 0.99093. The Accumulator component demonstrated similar robust performance with an accuracy of 0.99093 and corresponding precision, recall, and F1-score values around 0.990957. The Stable Flag component had slightly lower scores with an accuracy of 0.981859 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.10: Random Forest 200 Estimators no Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.993197	0.993347	0.993197	0.993217
Internal Pump Leakage	0.99093	0.991061	0.99093	0.990964
Accumulator	0.99093	0.991041	0.99093	0.990957
Stable Flag	0.981859	0.982082	0.981859	0.981764

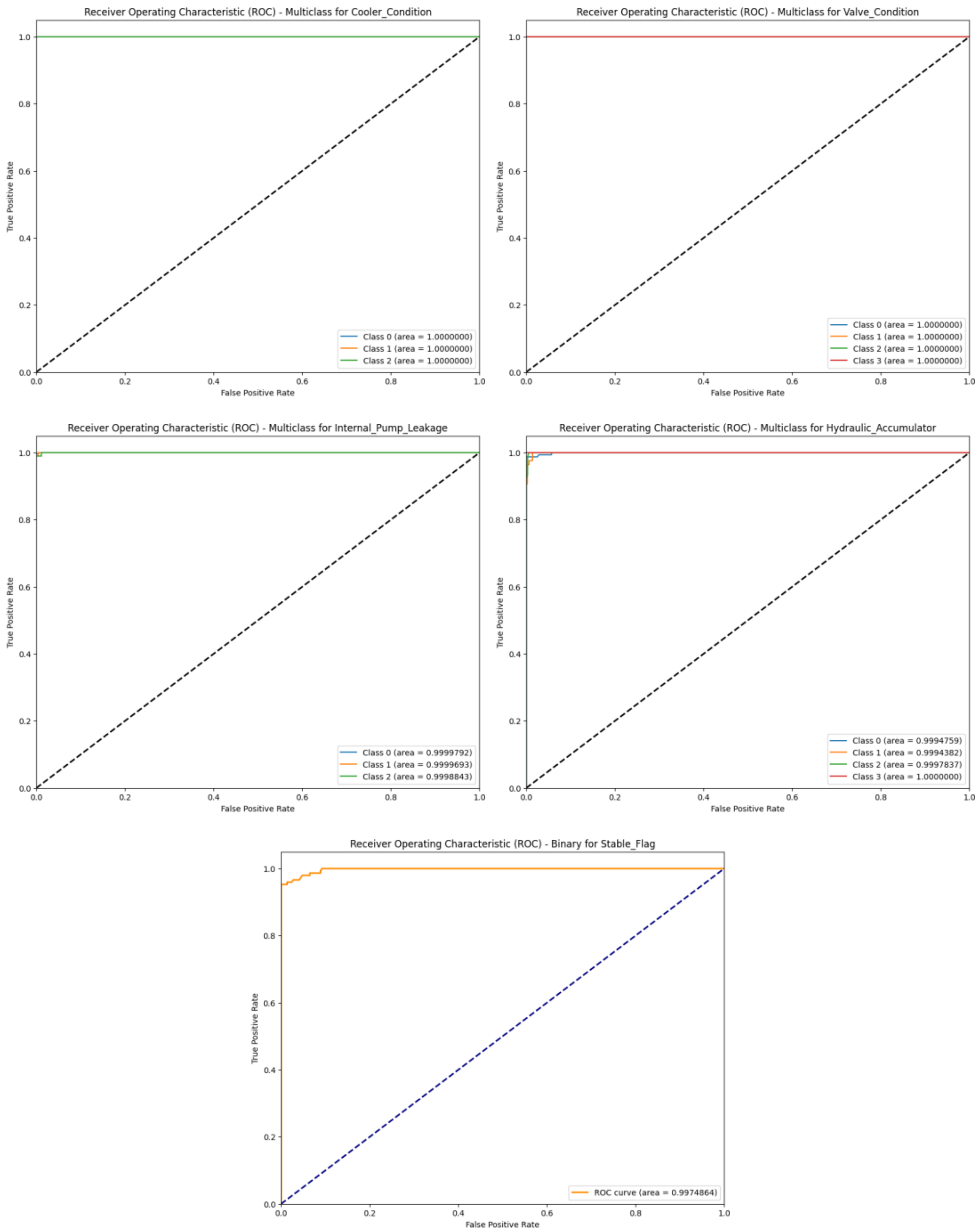


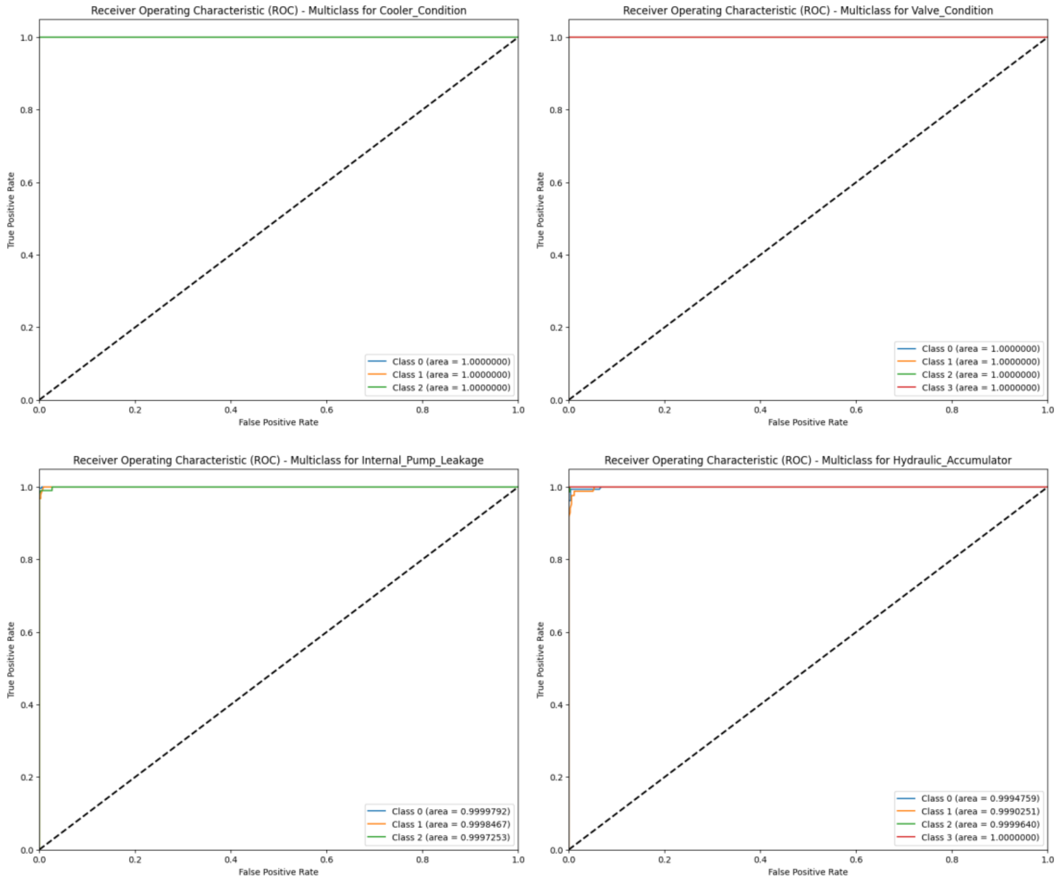
Figure 6.10 ROC AUC All Features (RF) – Phase 2

- **Pearson Correlation Analysis:**

The analysis of the conditioning components using Pearson Correlation Analysis reveals strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited high performance with an accuracy of 0.995465 and corresponding precision, recall, and F1-score values around 0.995485. The Internal Pump Leakage component also showed excellent performance with metrics around 0.988662. The Accumulator component demonstrated robust performance with an accuracy of 0.988662 and corresponding precision, recall, and F1-score values around 0.988696. The Stable Flag component had slightly lower scores with an accuracy of 0.979592 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.11: Random Forest 200 Estimators PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.995465	0.995584	0.995465	0.995485
Internal Pump Leakage	0.988662	0.988897	0.988662	0.988717
Accumulator	0.988662	0.98881	0.988662	0.988696
Stable Flag	0.979592	0.9797	0.979592	0.979503



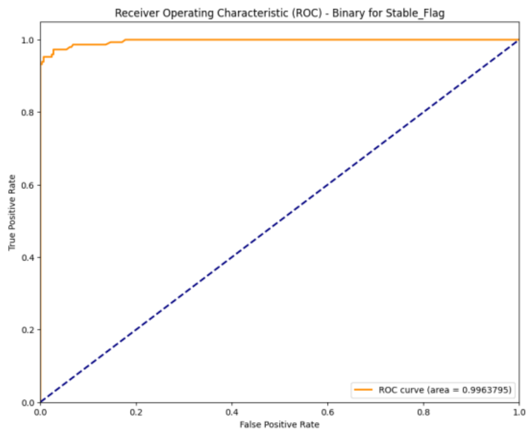


Figure 6.11 ROC AUC PCA (RF) – Phase 2

- **Linear Discriminant Analysis:**

The analysis of the conditioning components using Linear Discriminant Analysis (LDA) reveals that the Cooler component achieved perfect performance with accuracy, precision, recall, and an F1-score of 1.0. The Valve component demonstrated excellent performance with metrics slightly lower but still impressive, around 0.988 for accuracy, precision, recall, and F1-score. The Internal Pump Leakage component also performed well with metrics of approximately 0.991. The Accumulator component showed high accuracy, precision, recall, and F1-score around 0.991, indicating reliable performance. The Stable Flag component maintained robust performance with accuracy, precision, recall, and an F1-score of around 0.984. The ROC curves for these components confirm these results, displaying high AUC values close to 1.0 for all classes, signifying strong predictive capabilities across all components.

Table 6.12: Random Forest 200 Estimators LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.988843	0.988662	0.988692	0.999107
Internal Pump Leakage	0.99093	0.991061	0.99093	0.990964
Accumulator	0.99093	0.991041	0.99093	0.990957
Stable Flag	0.984127	0.984497	0.984127	0.984028

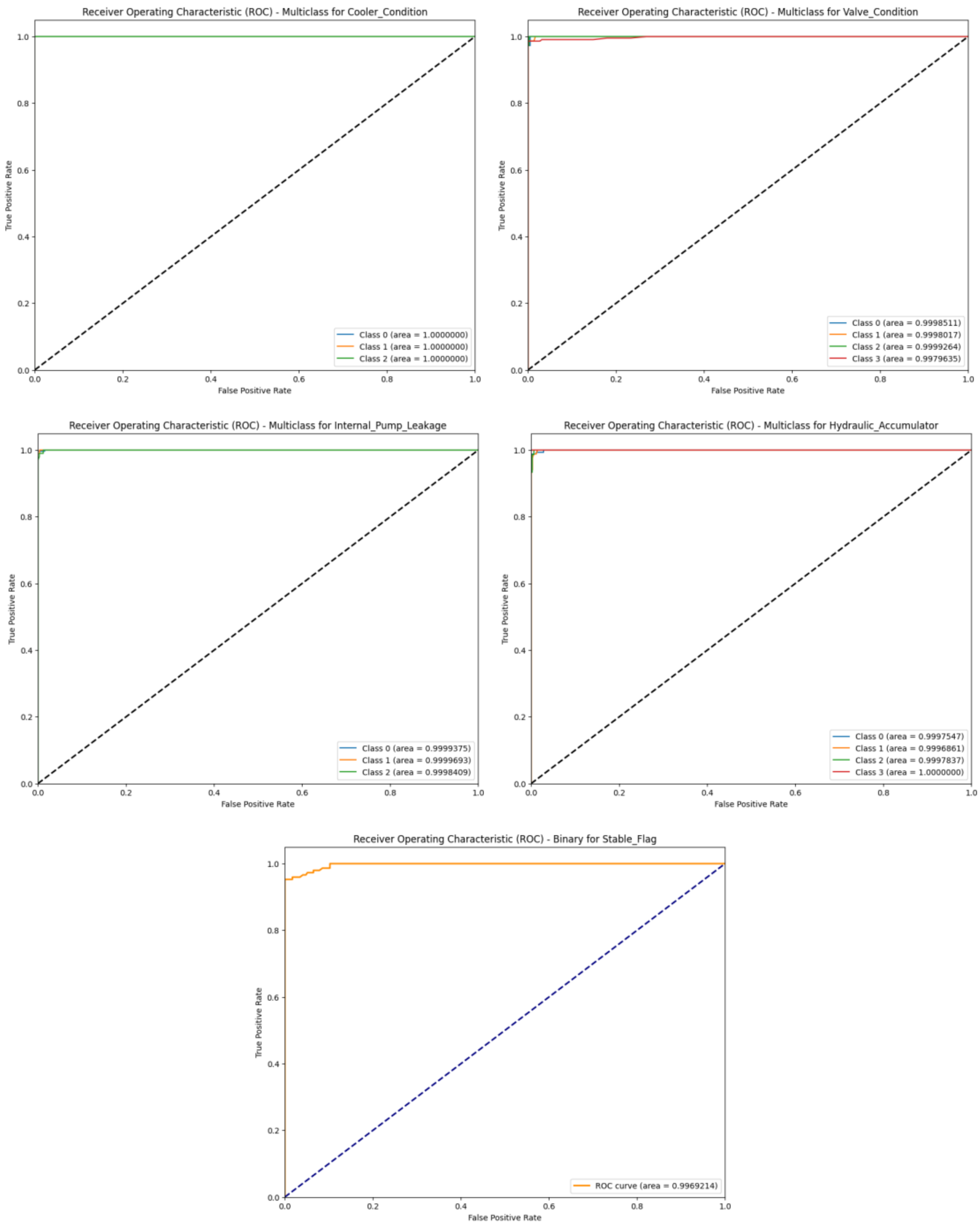


Figure 6.12 ROC AUC LDA (RF) – Phase 2

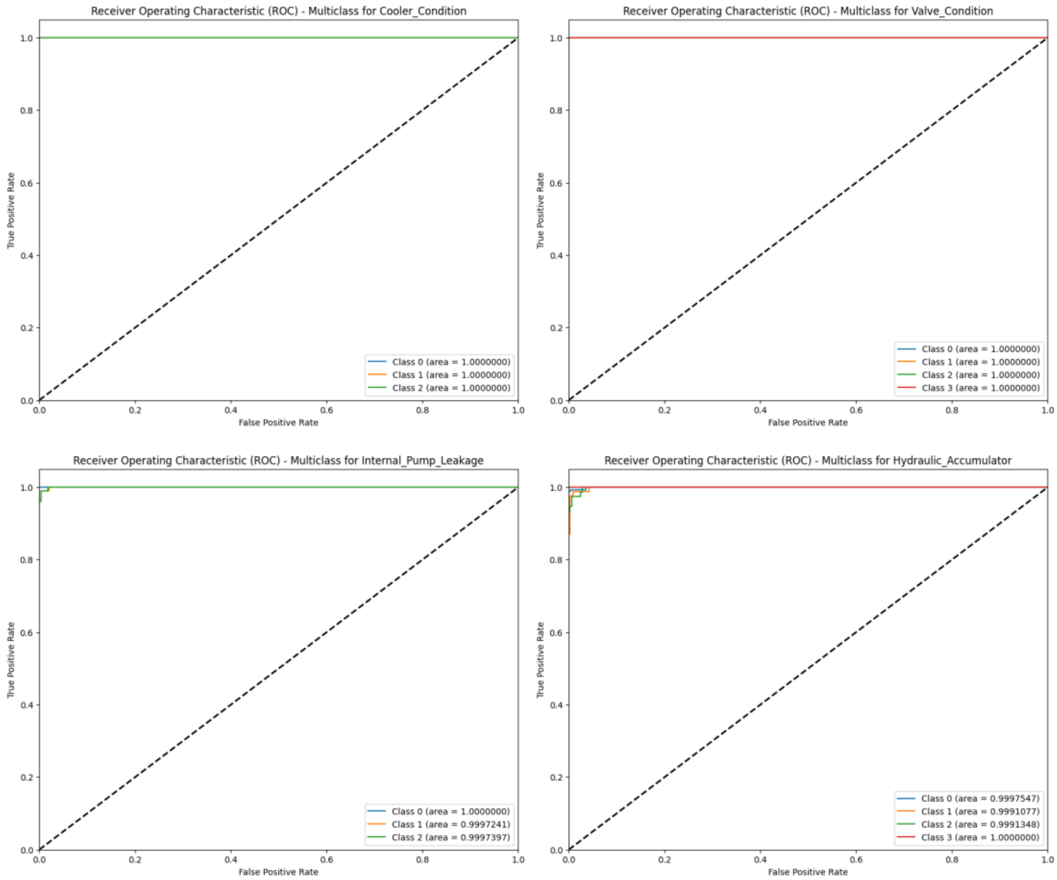
6.2.2 Catboost (200 of Estimators)

- **No Feature Selection:**

The analysis of the conditioning components using Catboost (200 Estimators) reveals that the Cooler and Valve components achieved perfect performance with accuracy, precision, recall, and an F1-score of 1.0. The Internal Pump Leakage component showed high performance with an accuracy of 0.988662 and an F1-score of 0.98873. Similarly, the Accumulator component maintained high performance with an accuracy of 0.988662 and an F1-score of 0.988662. The Stable Flag component demonstrated robust performance with an accuracy of 0.986395 and an F1-score of 0.986323. The ROC curves for these components confirm these results, displaying high AUC values close to 1.0 for all classes, signifying strong predictive capabilities across all components.

Table 6.13: Catboost 200 Estimators No Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	1.0	1.0	1.0	1.0
Internal Pump Leakage	0.988662	0.988943	0.988662	0.98873
Accumulator	0.988662	0.988662	0.988662	0.988662
Stable Flag	0.986395	0.986668	0.986395	0.986323



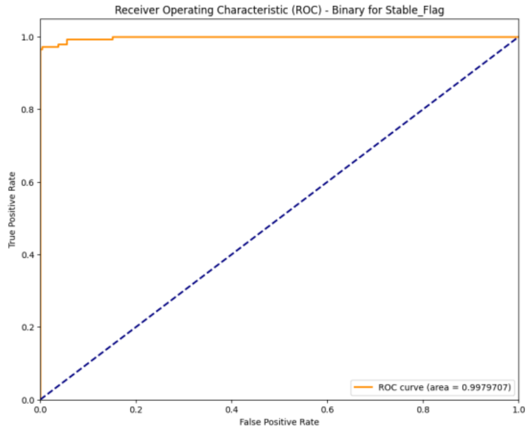


Figure 6.13 ROC AUC All Features (Catboost) – Phase 2

- **Pearson Correlation Analysis:**

The Pearson Correlation Analysis of the conditioning components reveals that the Cooler and Valve components both achieved perfect performance with accuracy, precision, recall, and an F1-score of 1.0. The Internal Pump Leakage component demonstrated high performance with an accuracy of 0.99093 and an F1-score of 0.990958. The Accumulator component also showed strong performance with an accuracy of 0.981859 and an F1-score of 0.98185. The Stable Flag component maintained high performance with an accuracy of 0.986395 and an F1-score of 0.986371. The ROC curves for these components confirm these results, displaying high AUC values close to 1.0 for all classes, signifying strong predictive capabilities across all components.

Table 6.14: Catboost 200 Estimators PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	1.0	1.0	1.0	1.0
Internal Pump Leakage	0.99093	0.991049	0.99093	0.990958
Accumulator	0.981859	0.982026	0.981859	0.98185
Stable Flag	0.986395	0.986394	0.986395	0.986371

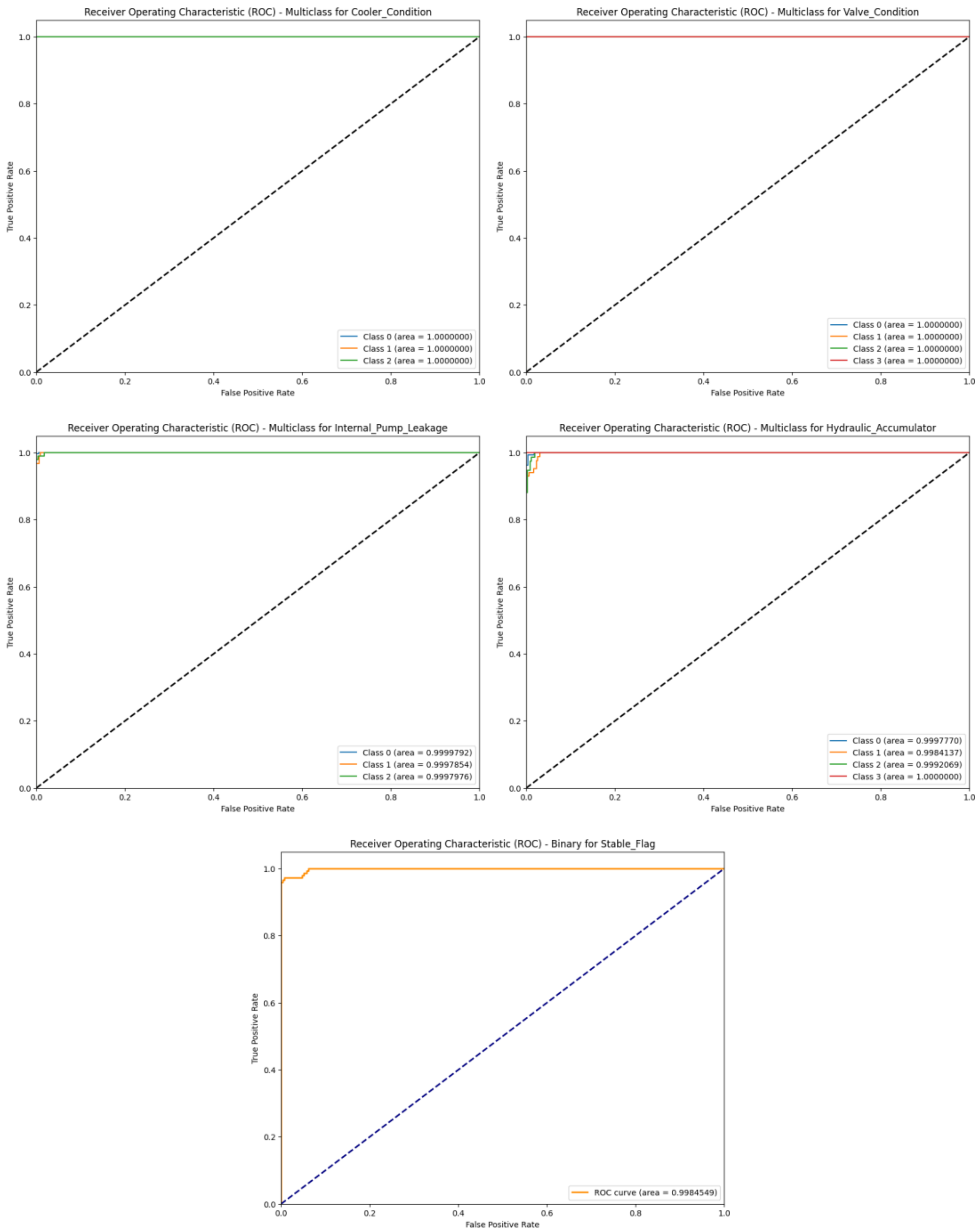


Figure 6.14 ROC AUC PCA (Catboost) – Phase 2

- **Linear Discriminant Analysis:**

The Linear Discriminant Analysis (LDA) results similarly reveal excellent performance. The Cooler component achieved perfect performance, with an accuracy, precision, recall, and F1-score of 1.0. The Valve component demonstrated high performance with an accuracy of 0.99093 and an F1-score of 0.99096. The Internal Pump Leakage component performed well, with an accuracy of 0.984127 and an F1-score of 0.984188. The Accumulator component showed strong performance with an accuracy of 0.981859 and an F1-score of 0.981903. The Stable Flag component maintained solid performance with an accuracy of 0.972789 and an F1-score of 0.972646. The ROC curves for these components corroborate these results, displaying high AUC values close to 1.0, indicating strong predictive capabilities across all components.

Table 6.15: Catboost 200 Estimators LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.99093	0.99117	0.99093	0.99096
Internal Pump Leakage	0.984127	0.984386	0.984127	0.984188
Accumulator	0.981859	0.982024	0.981859	0.981903
Stable Flag	0.972789	0.972911	0.972789	0.972646

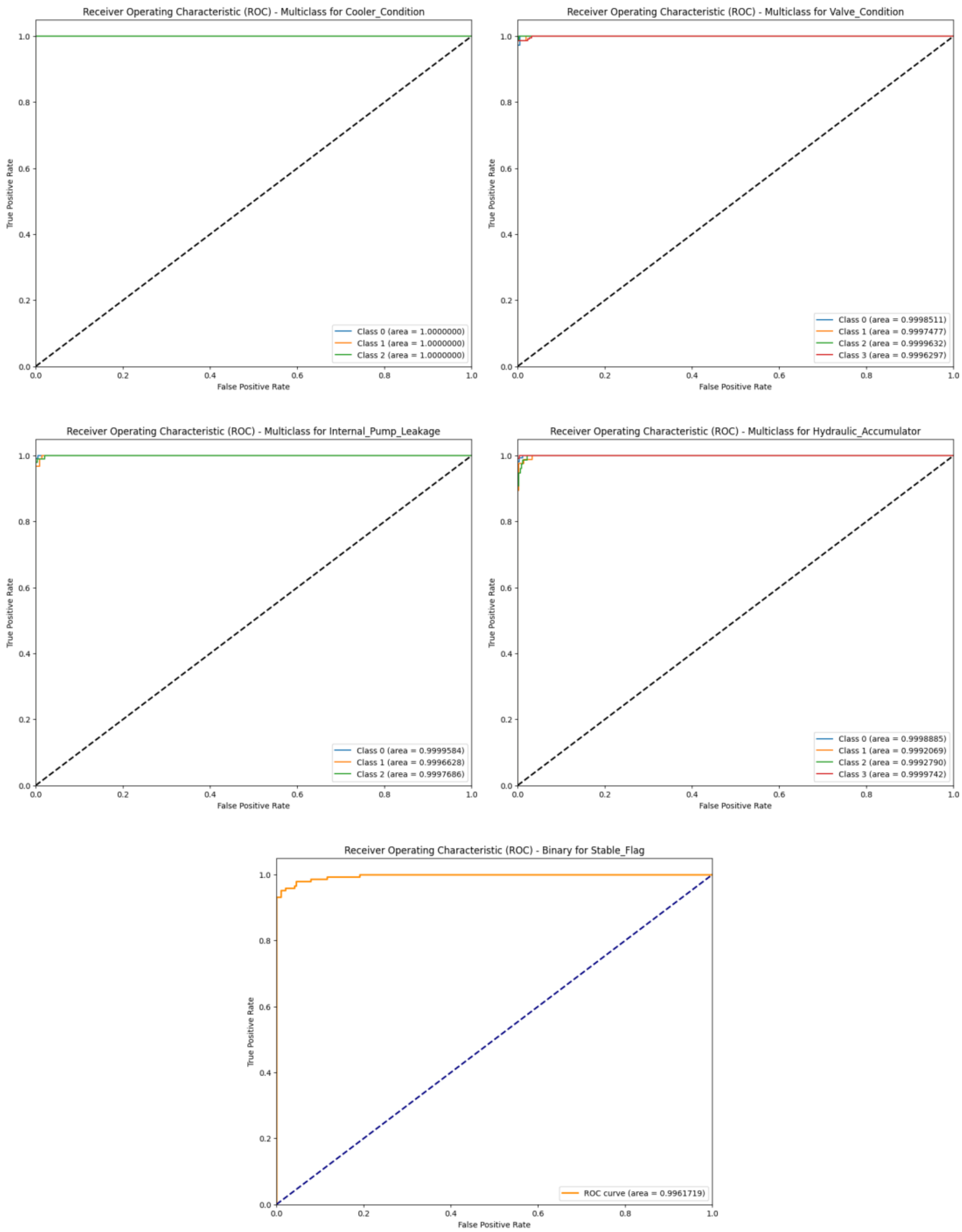


Figure 6.15 ROC AUC LDA (Catboost) – Phase 2

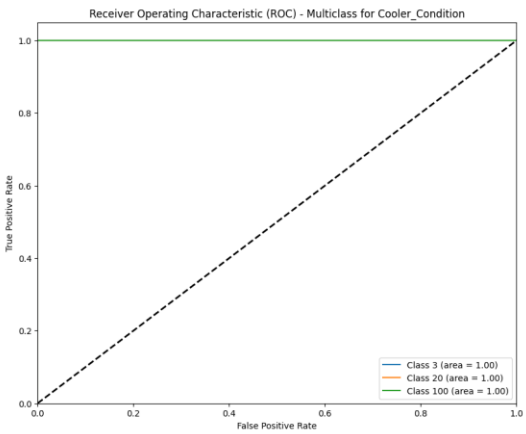
6.2.3 LSTM (32 Batch Size)

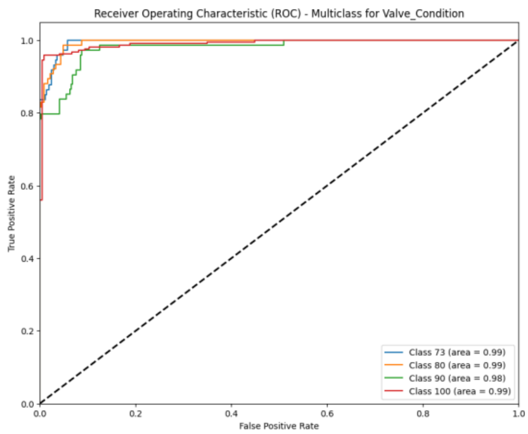
- **No Feature Selection:**

The LSTM analysis with a batch size of 32 and no feature selection shows that the Cooler component achieved perfect performance, with accuracy, precision, recall, and F1-score all being 1.0. The Valve component exhibited good performance with an accuracy of 0.886621 and an F1-score of 0.917169. The Internal Pump Leakage component performed well, with an accuracy of 0.968254 and an F1-score of 0.968319. The Accumulator component showed moderate performance, with an accuracy of 0.8322 and an F1-score of 0.86229. The Stable Flag component demonstrated strong performance, with an accuracy of 0.956916 and an F1-score of 0.95688. The ROC curves for these components support these findings, showing high AUC values and indicating strong predictive capabilities.

Table 6.16: LSTM 32 Batch Size No Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.886621	0.956094	0.886621	0.917169
Internal Pump Leakage	0.968254	0.969466	0.968254	0.968319
Accumulator	0.8322	0.900241	0.8322	0.86229
Stable Flag	0.956916	0.956854	0.956916	0.95688





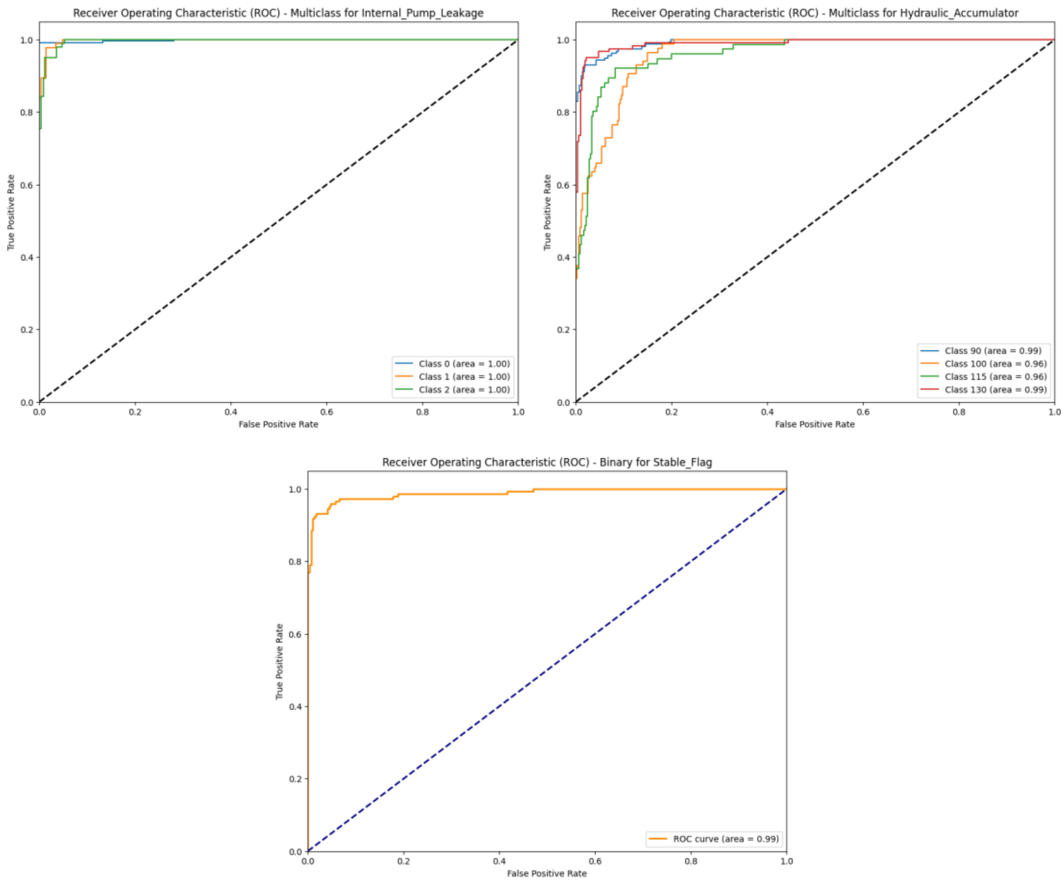


Figure 6.16 ROC AUC All Features (LSTM) – Phase 2

- **Pearson Correlation Analysis:**

The LSTM analysis with a batch size of 32 with Pearson correlation analysis shows that the Cooler component achieved perfect performance, with accuracy, precision, recall, and F1-score all being 1.0. The Valve component exhibited good performance with an accuracy of 0.911565 and an F1-score of 0.937869. The Internal Pump Leakage component performed well, with an accuracy of 0.977324 and an F1-score of 0. 977428. The Accumulator component showed the lowest performance, with an accuracy of 0. 634921 and an F1-score of 0.719172. The Stable Flag component demonstrated strong performance, with an accuracy of 0.952381 and an F1-score of 0.952497. The ROC curves for these components support these findings, showing high AUC values and indicating strong predictive capabilities.

Table 6.17: LSTM 32 Batch Size PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.911565	0.968978	0.911565	0.937869
Internal Pump Leakage	0.977324	0.977675	0.977324	0.977428
Accumulator	0.634921	0.902511	0.634921	0.719172
Stable Flag	0.952381	0.95271	0.952381	0.952497

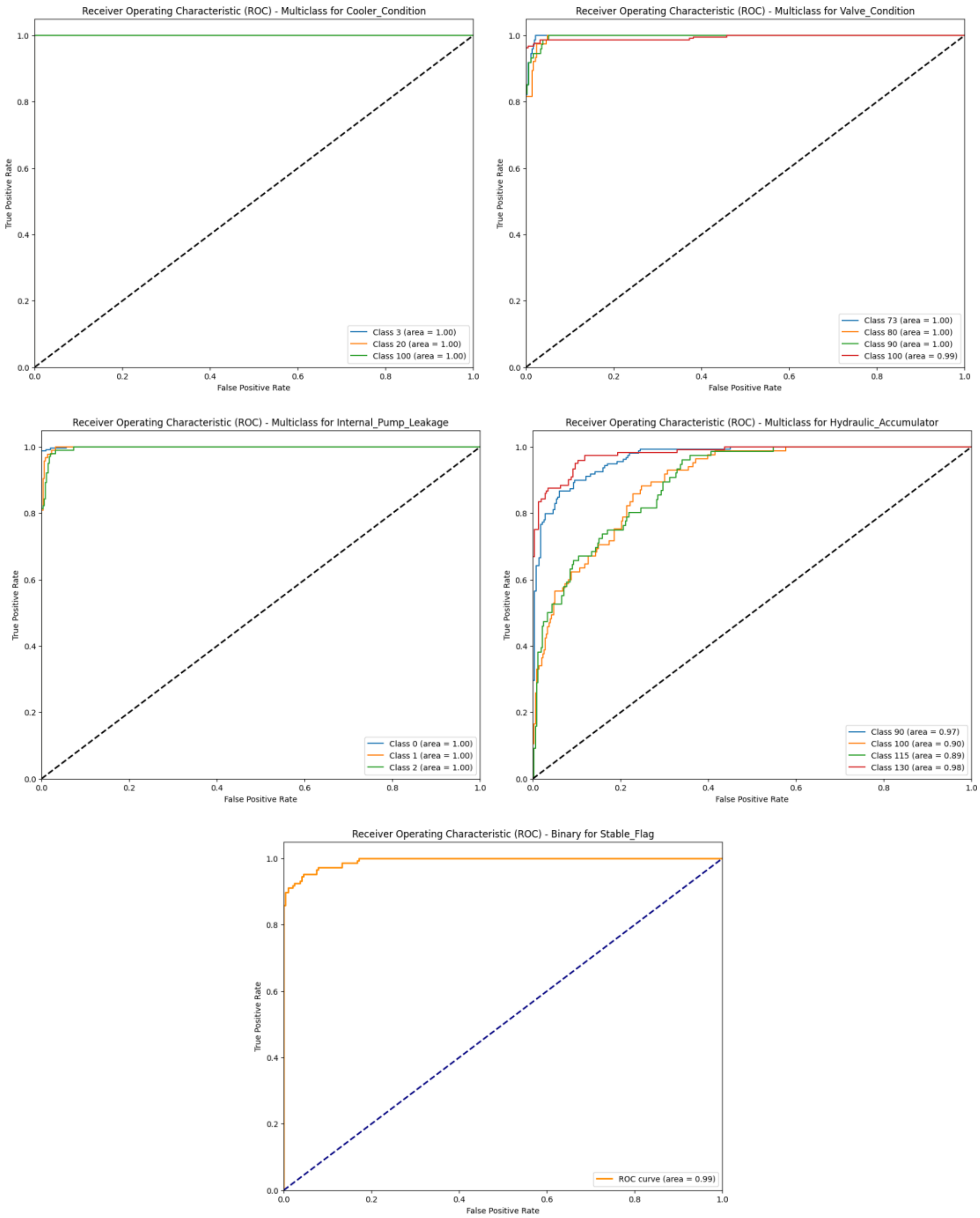


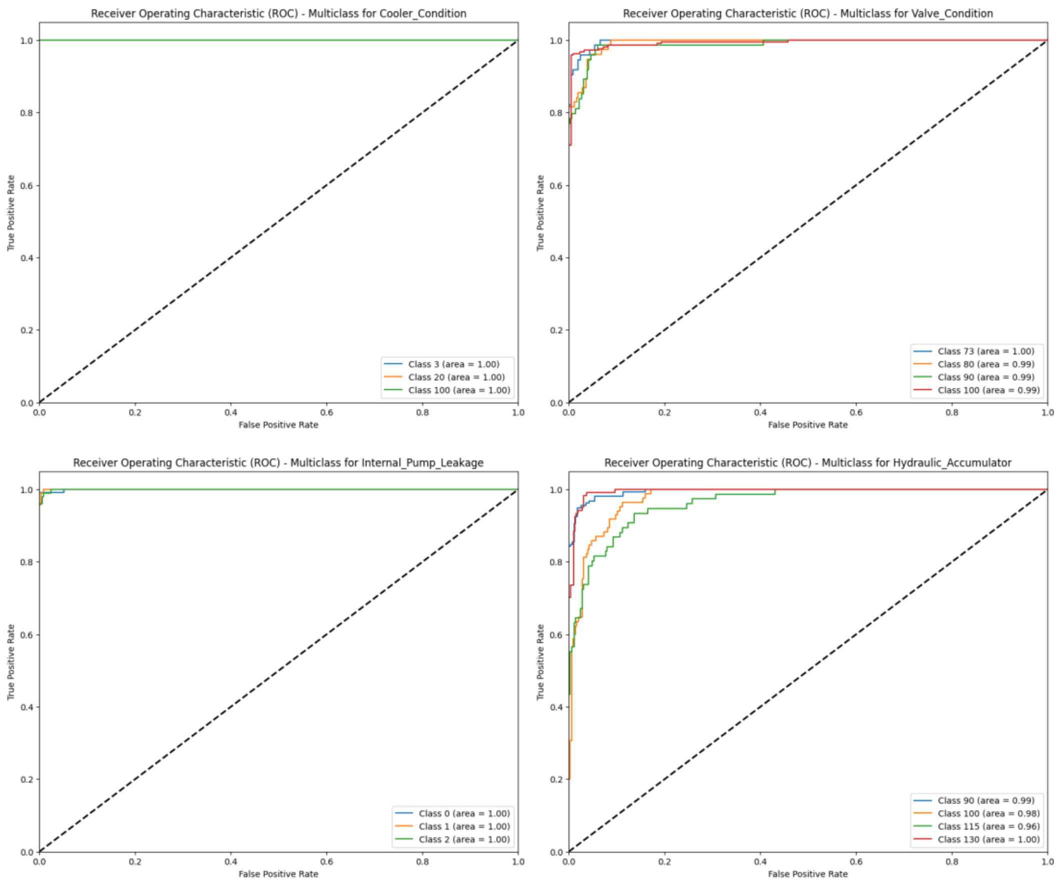
Figure 6.17 ROC AUC PCA (LSTM) – Phase 2

- **Linear Discriminant Analysis:**

The LSTM analysis with a batch size of 32 with linear discriminant analysis shows that the Cooler component achieved perfect performance, with accuracy, precision, recall, and F1-score all being 1.0. The Valve component exhibited good performance with an accuracy of 0. 902494 and an F1-score of 0. 93144. The Internal Pump Leakage component performed well, with an accuracy of 0.988662 and an F1-score of 0. 988717. The Accumulator component showed moderate performance, with an accuracy of 0.877551 and an F1-score of 0. 894171. The Stable Flag component demonstrated strong performance, with an accuracy of 0.963719 and an F1-score of 0.963459. The ROC curves for these components support these findings, showing high AUC values and indicating strong predictive capabilities.

Table 6.18: LSTM 32 Batch Size LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.902494	0.967989	0.902494	0.93144
Internal Pump Leakage	0.988662	0.988897	0.988662	0.988717
Accumulator	0.877551	0.912193	0.877551	0.894171
Stable Flag	0.963719	0.963924	0.963719	0.963459



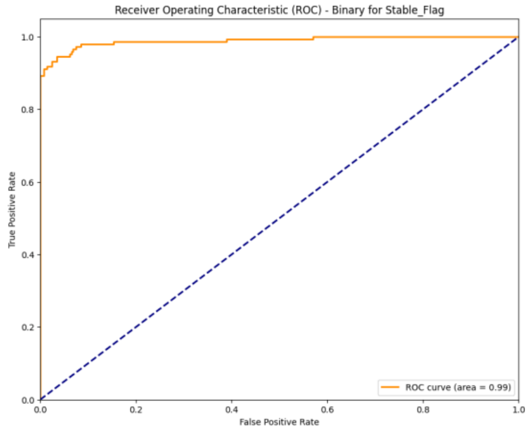


Figure 6.18 ROC AUC LDA (LSTM) – Phase 2

6.3 Phase Three of Testing

6.3.1 Random Forest (300 of Estimators)

- **No Feature Selection:**

The analysis of the conditioning components using a Random Forest model with 300 estimators and no feature selection reveals strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited high performance with an accuracy of 0.99093 and corresponding precision, recall, and F1-score values around 0.990974. The Internal Pump Leakage component also showed excellent performance with metrics around 0.99093. The Accumulator component demonstrated similar robust performance with an accuracy of 0.99093 and corresponding precision, recall, and F1-score values around 0.990957. The Stable Flag component had slightly lower scores with an accuracy of 0.981859 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.19: Random Forest 300 Estimators No Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.99093	0.991225	0.99093	0.990974
Internal Pump Leakage	0.99093	0.991061	0.99093	0.990964
Accumulator	0.99093	0.991041	0.99093	0.990957
Stable Flag	0.981859	0.982082	0.981859	0.981764

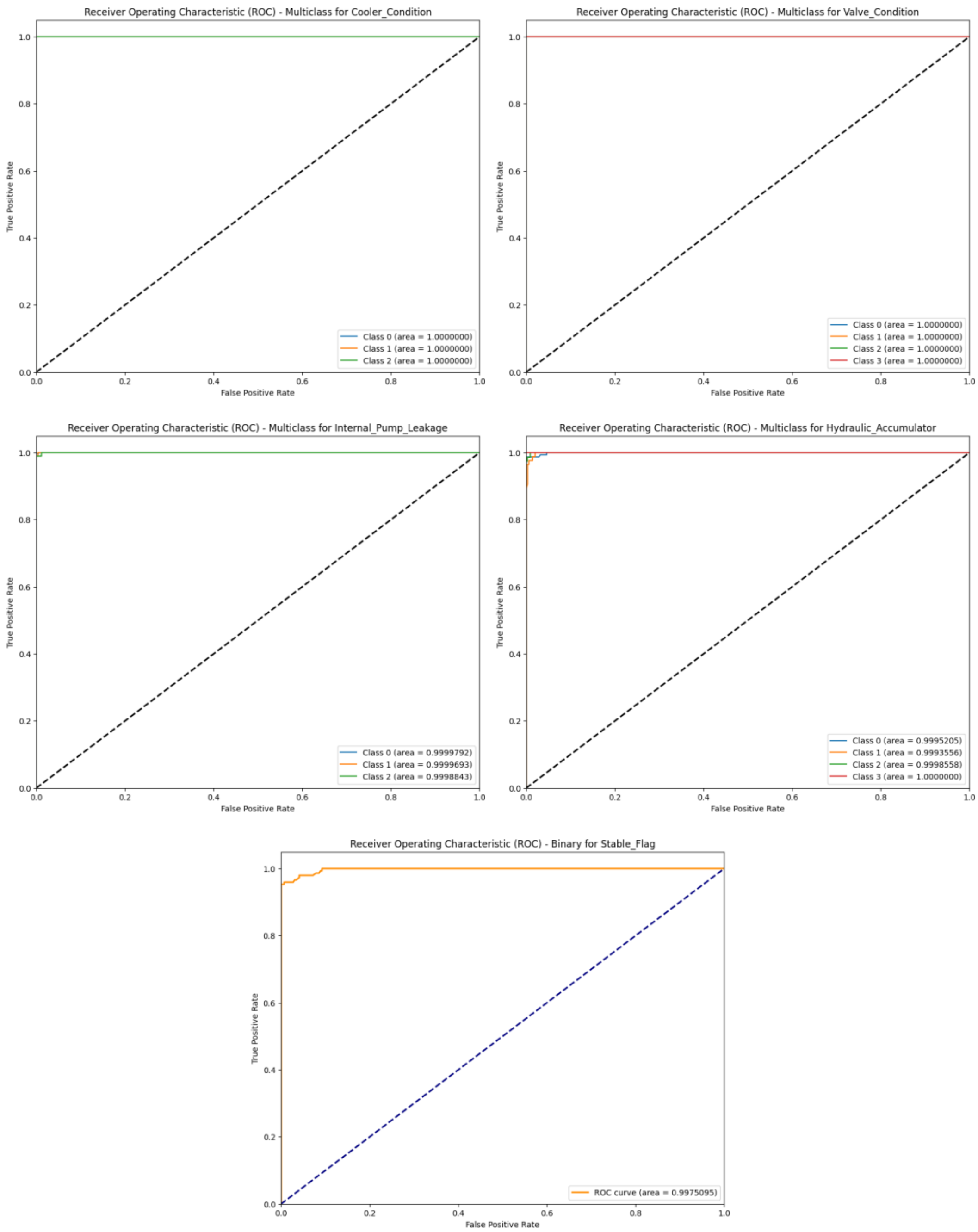


Figure 6.19 ROC AUC All Features (RF) – Phase 3

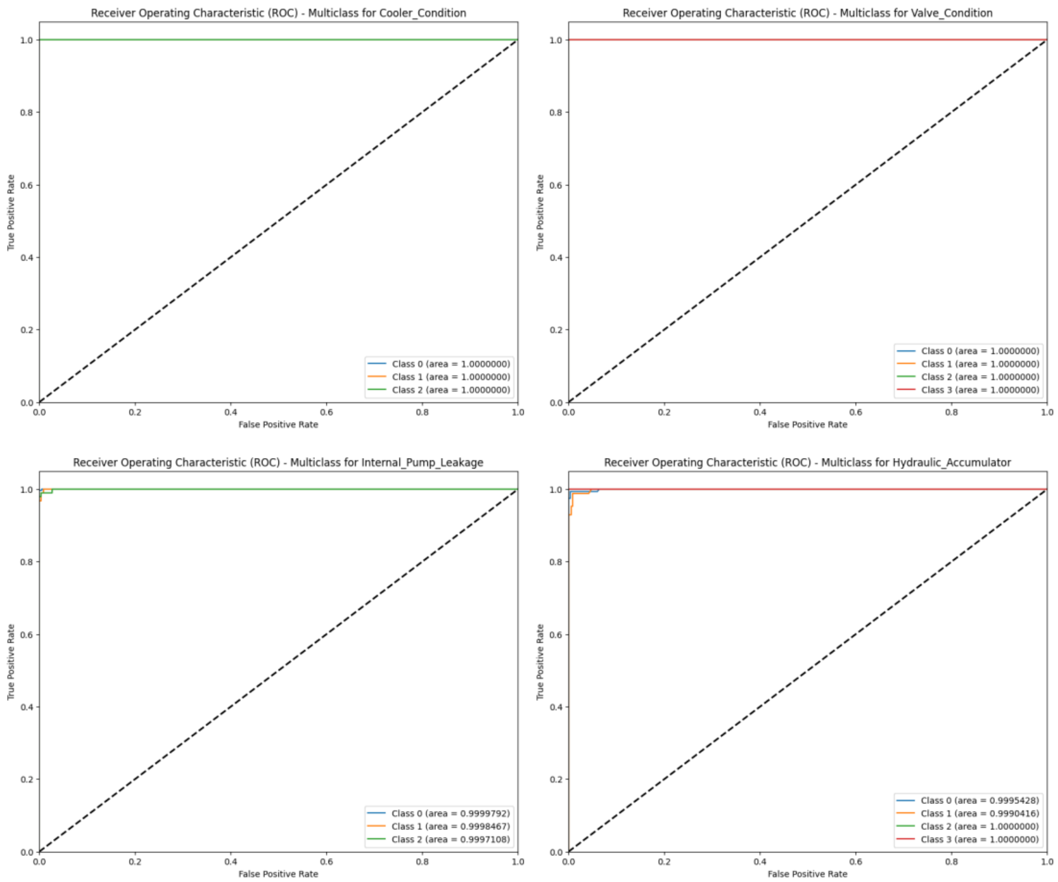
- **Pearson Correlation Analysis:**

The analysis of the conditioning components using a Random Forest model with 300 estimators and Pearson correlation analysis showed strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited high performance with an accuracy of 0.997732 and corresponding precision, recall, and F1-score values around 0.997737. The Internal Pump Leakage component also showed excellent performance with metrics around 0.988662. The Accumulator component demonstrated similar robust performance with an accuracy of 0.986395 and corresponding precision, recall, and F1-

score values around 0.986405. The Stable Flag component had slightly lower scores with an accuracy of 0.979592 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.20: Random Forest 300 Estimators PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.997732	0.997763	0.997732	0.997737
Internal Pump Leakage	0.988662	0.988897	0.988662	0.988717
Accumulator	0.986395	0.986543	0.986395	0.986405
Stable Flag	0.979592	0.9797	0.979592	0.979503



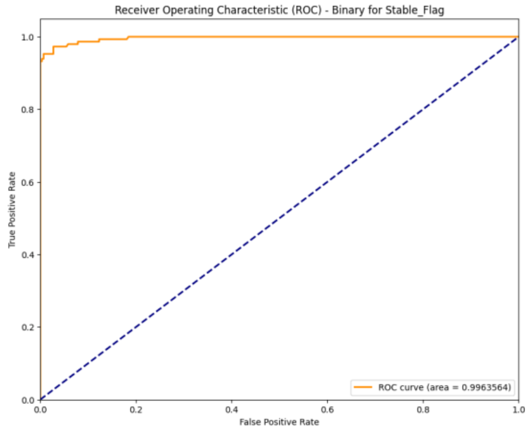


Figure 6.20 ROC AUC PCA (RF) – Phase 3

- **Linear Discriminant Analysis:**

The analysis of the conditioning components using a Random Forest model with 300 estimators and linear discriminant analysis showed strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited slightly lower performance with an accuracy of 0.986573 and corresponding precision, recall, and F1-score values around 0.998882. The Internal Pump Leakage component also showed excellent performance with metrics around 0.99093. The Accumulator component demonstrated similar robust performance with an accuracy of 0.99093 and corresponding precision, recall, and F1-score values around 0.990957. The Stable Flag component had slightly lower scores with an accuracy of 0.984127 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.21: Random Forest 300 Estimators LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.986573	0.986395	0.986423	0.998882
Internal Pump Leakage	0.99093	0.991061	0.99093	0.990964
Accumulator	0.99093	0.991041	0.99093	0.990957
Stable Flag	0.984127	0.984497	0.984127	0.984028

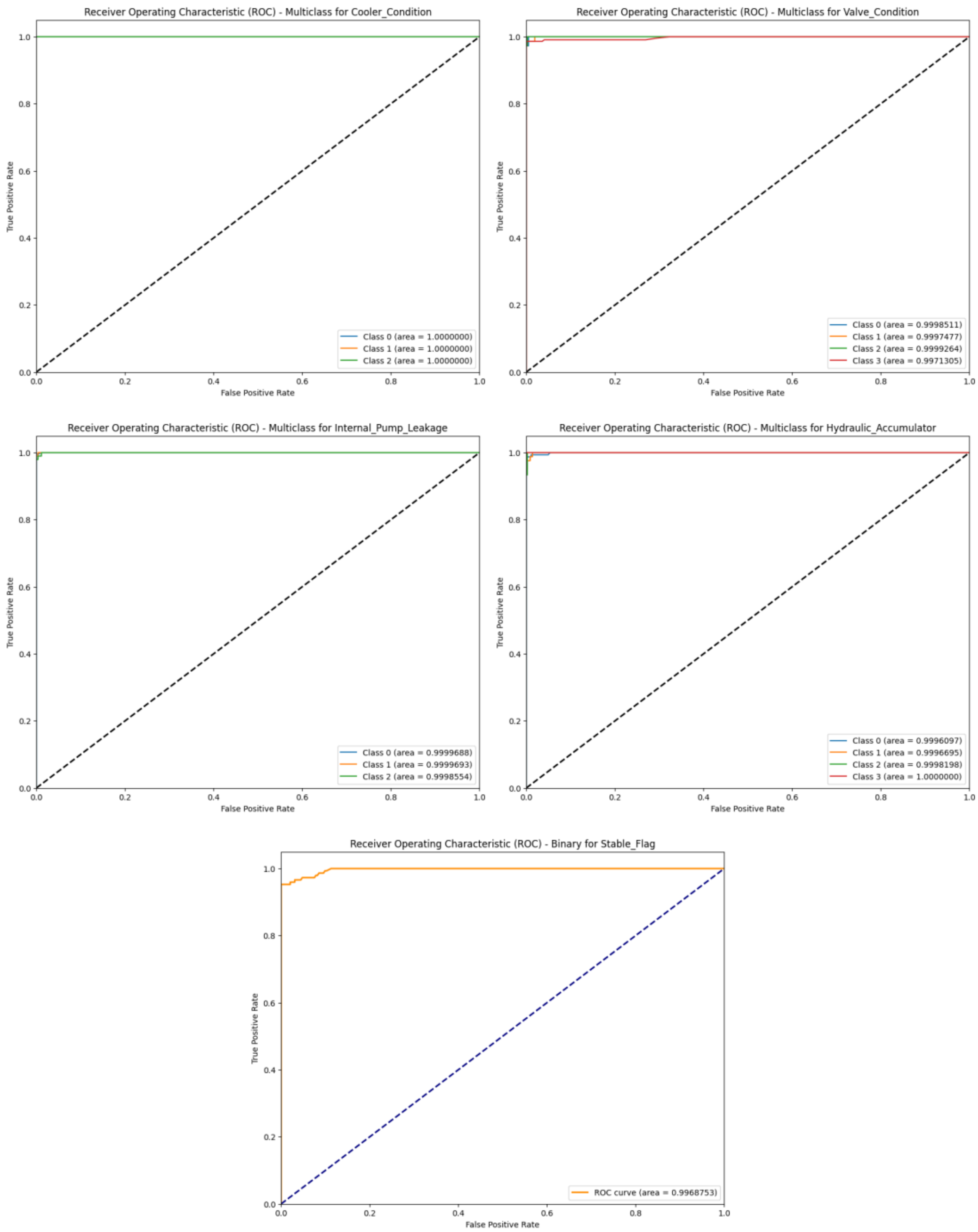


Figure 6.21 ROC AUC LDA (RF) – Phase 3

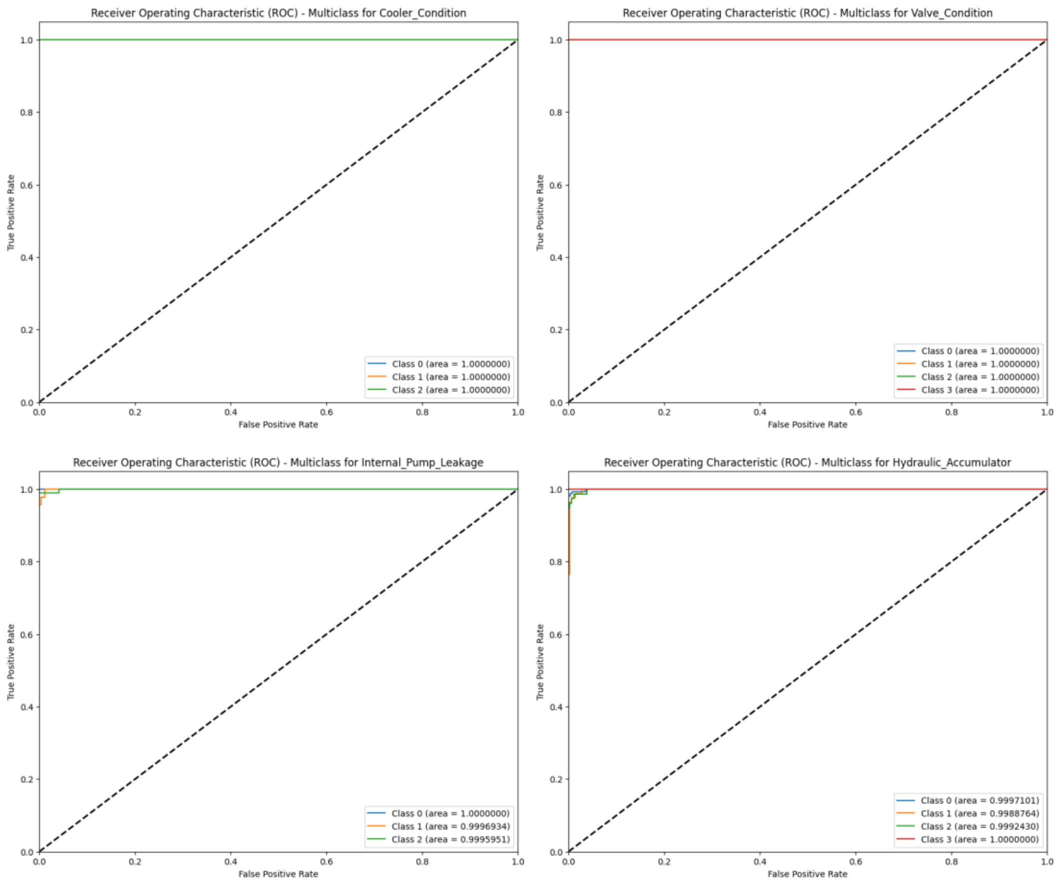
6.3.2 Catboost (300 of Estimators)

- **No Feature Selection:**

The analysis of the conditioning components using a Catboost model with 300 estimators and no feature selection showed strong performance across different components. The Cooler and Valve components achieved perfect scores with accuracy, precision, recall, and an F1 score of 1.0. The Internal Pump Leakage component also showed slightly lower performance with metrics around 0.988662. The Accumulator component demonstrated similar robust performance with an accuracy of 0.986395 and corresponding precision, recall, and F1-score values around 0.986409. The Stable Flag component had slightly lower scores with an accuracy of 0.988613 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.22: Catboost 300 Estimators No Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	1.0	1.0	1.0	1.0
Internal Pump Leakage	0.988662	0.988943	0.988662	0.98873
Accumulator	0.986395	0.986501	0.986395	0.986409
Stable Flag	0.988662	0.988852	0.988662	0.988613



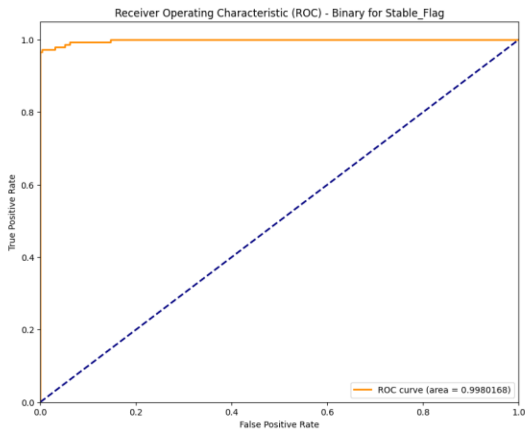


Figure 6.22 ROC AUC All Features (Catboost) – Phase 3

- **Pearson Correlation Analysis:**

The analysis of the conditioning components using a Catboost model with 300 estimators and Pearson correlation analysis showed strong performance across different components. The Cooler and Valve components achieved perfect scores with accuracy, precision, recall, and an F1 score of 1.0. The Internal Pump Leakage component also showed excellent performance with metrics around 0.99093. The Accumulator component demonstrated slightly lower performance with an accuracy of 0.977324 and corresponding precision, recall, and F1-score values around 0.986409. The Stable Flag component had slightly lower scores with an accuracy of 0.984127 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.23: Catboost 300 Estimators PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	1.0	1.0	1.0	1.0
Internal Pump Leakage	0.99093	0.990974	0.99093	0.990944
Accumulator	0.977324	0.977642	0.977324	0.977311
Stable Flag	0.984127	0.984112	0.984127	0.984114

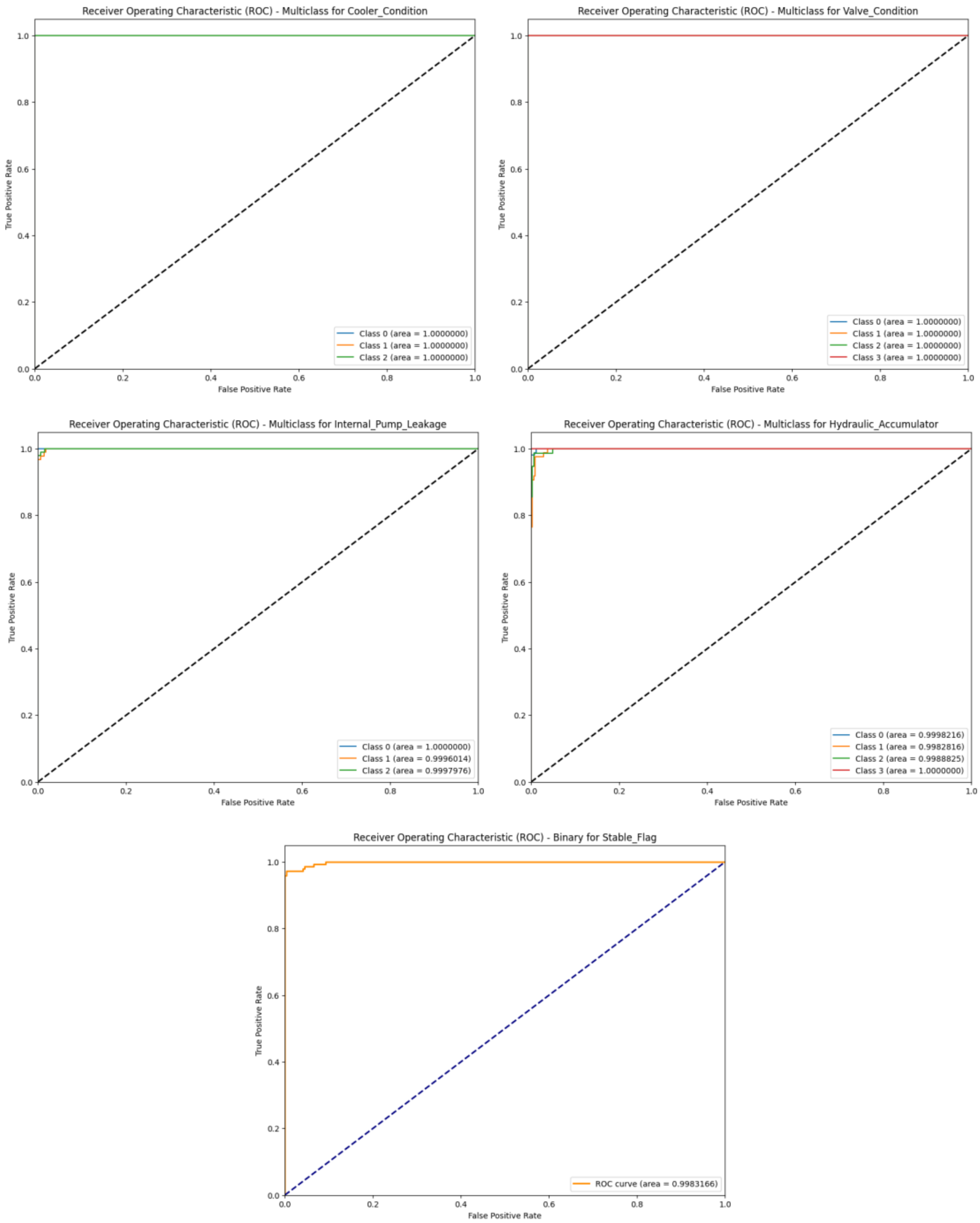


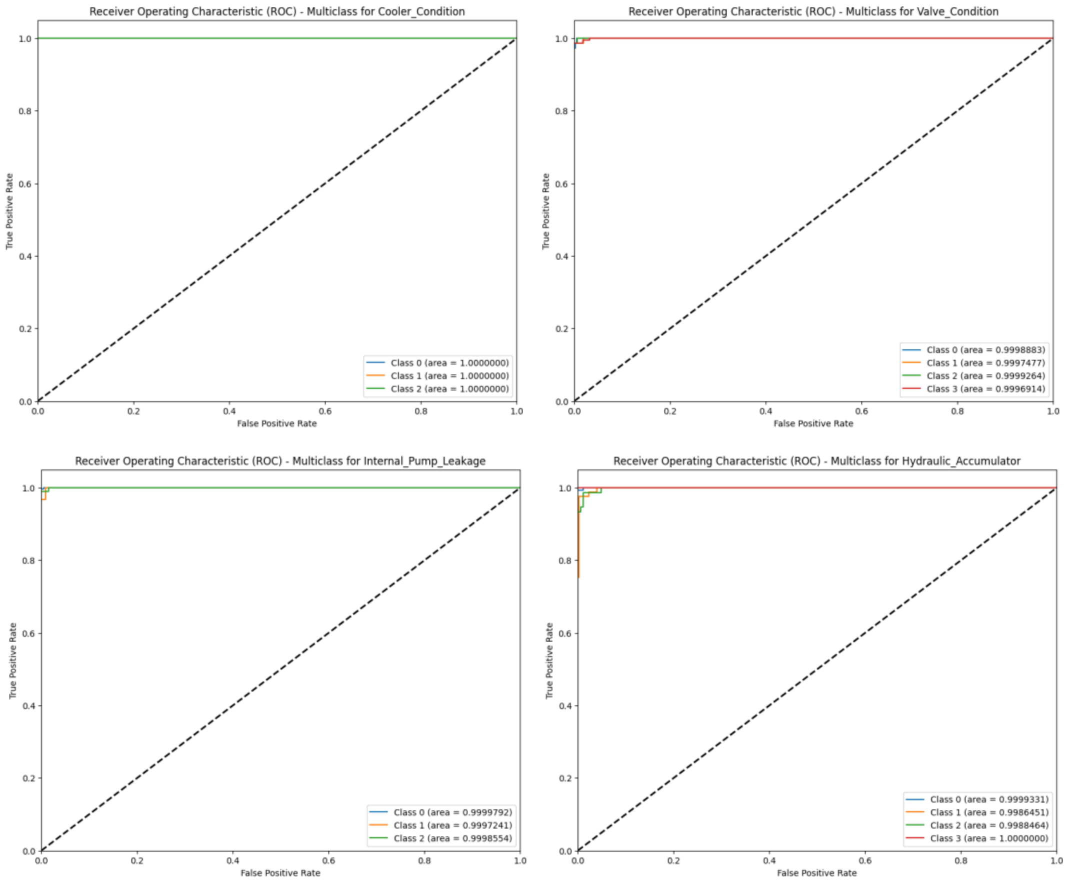
Figure 6.23 ROC AUC PCA (Catboost) – Phase 3

- **Linear Discriminant Analysis:**

The analysis of the conditioning components using a Catboost model with 300 estimators and linear discriminant analysis showed strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited slightly lower performance with an accuracy of 0.99093 and corresponding precision, recall, and F1-score values around 0.99096. The Internal Pump Leakage component also showed slightly lower performance with metrics around 0.988662. The Accumulator component demonstrated similar robust performance with an accuracy of 0.988662 and corresponding precision, recall, and F1-score values around 0.988743. The Stable Flag component had slightly lower scores with an accuracy of 0.977324 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.24: Catboost 300 Estimators LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.99093	0.99117	0.99093	0.99096
Internal Pump Leakage	0.988662	0.988778	0.988662	0.988699
Accumulator	0.988662	0.98898	0.988662	0.988743
Stable Flag	0.977324	0.977349	0.977324	0.977246



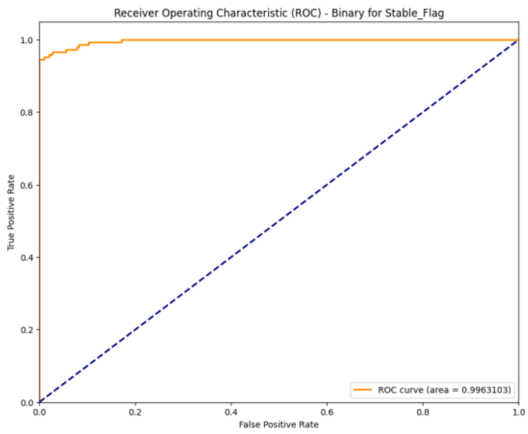


Figure 6.24 ROC AUC LDA (Catboost) – Phase 3

6.3.3 LSTM (48 Batch Size)

- **No Feature Selection:**

The analysis of the conditioning components using an LSTM model with 48 batch sizes and no feature selection showed strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited lower performance with an accuracy of 0.868481 and corresponding precision, recall, and F1-score values around 0.921549. The Internal Pump Leakage component also showed slightly lower performance with metrics around 0.975057. The Accumulator component demonstrated lower performance with an accuracy of 0.791383 and corresponding precision, recall, and F1-score values around 0.823836. The Stable Flag component had slightly lower scores with an accuracy of 0. 959184 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.25: LSTM 48 Batch Size No Feature Selection

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.868481	0.987622	0.868481	0.921549
Internal Pump Leakage	0.975057	0.975769	0.975057	0.975233
Accumulator	0.791383	0.87268	0.791383	0.823836
Stable Flag	0.959184	0.959089	0.959184	0.959114

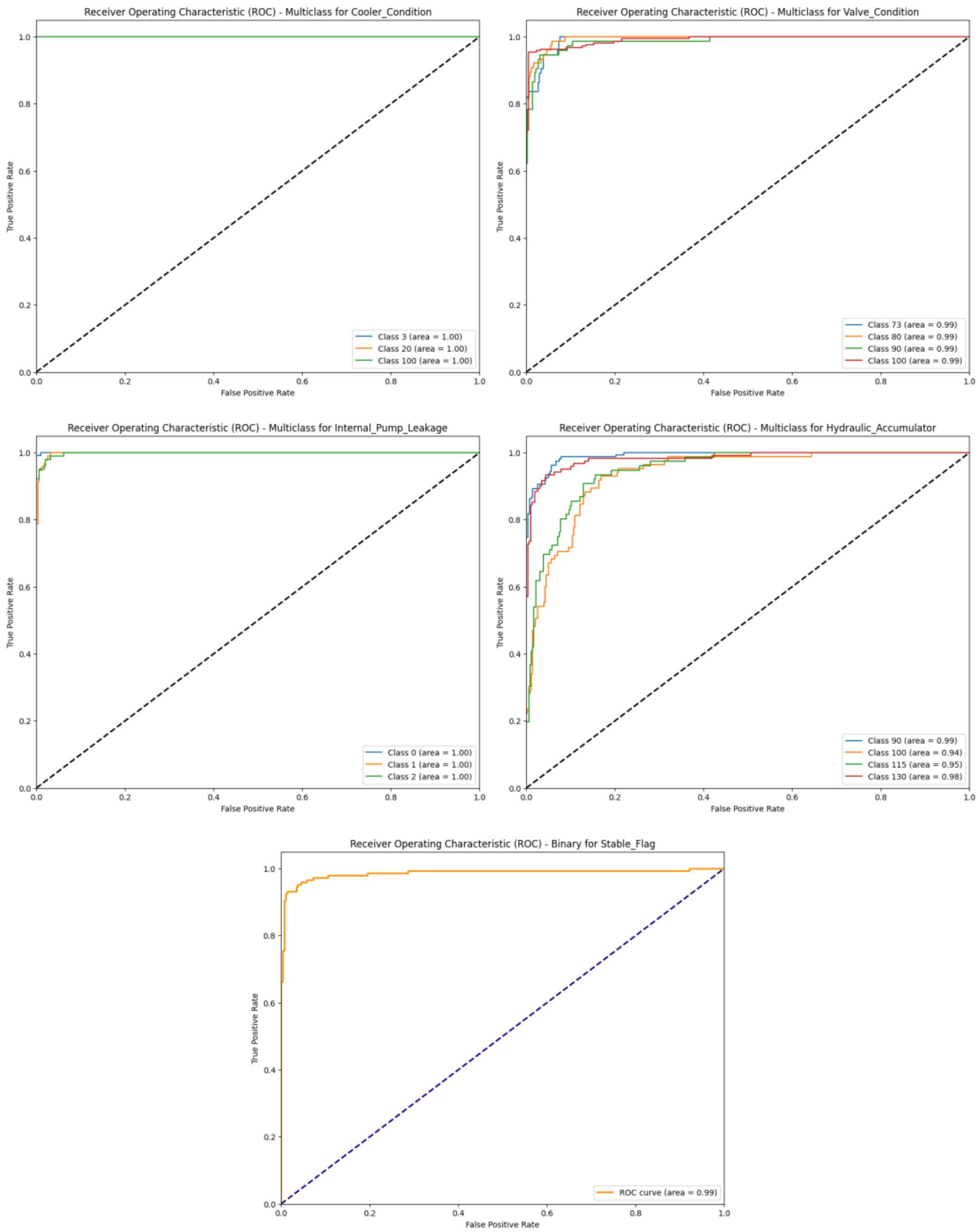


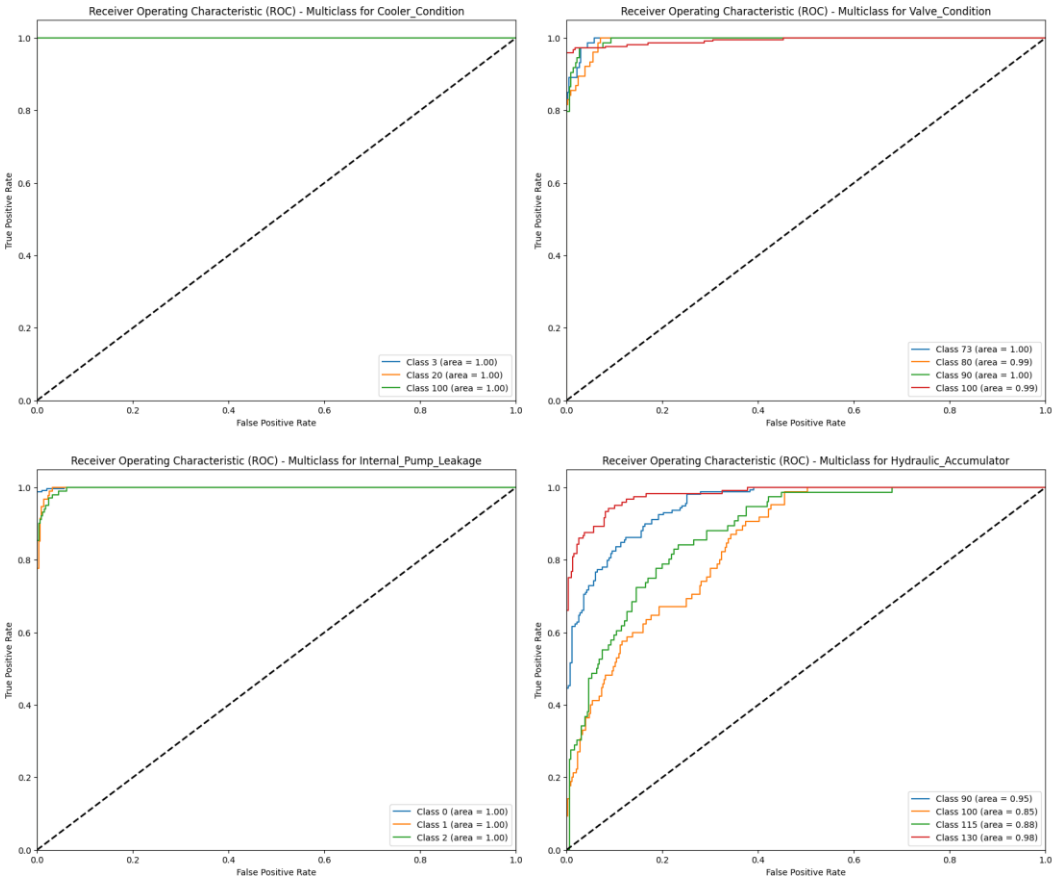
Figure 6.25 ROC AUC All Features (LSTM) – Phase 3

- **Pearson Correlation Analysis:**

The analysis of the conditioning components using an LSTM model with 48 batch sizes and Pearson correlation analysis showed strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited lower performance with an accuracy of 0.895692 and corresponding precision, recall, and F1-score values around 0.937596. The Internal Pump Leakage component also showed slightly lower performance with metrics around 0.972789. The Accumulator component demonstrated extremely low performance with an accuracy of 0.564626 and corresponding precision, recall, and F1-score values around 0.618207. The Stable Flag component had slightly lower scores with an accuracy of 0.959184 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components except for the accumulator. It displays some excellent and some good predictions for different classes.

Table 6.26: LSTM 48 Batch Size PCA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.895692	0.988218	0.895692	0.937596
Internal Pump Leakage	0.972789	0.973376	0.972789	0.972975
Accumulator	0.564626	0.878115	0.564626	0.618207
Stable Flag	0.959184	0.959089	0.959184	0.959114



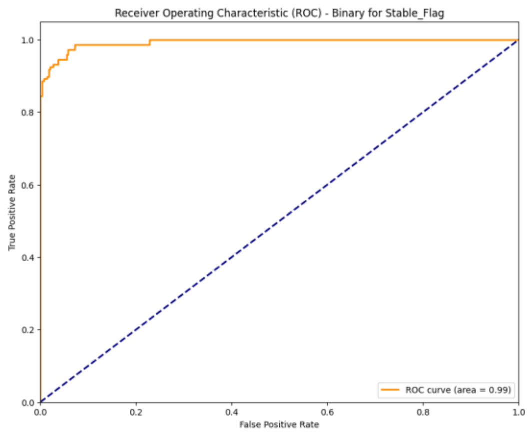


Figure 6.26 ROC AUC PCA (LSTM) – Phase 3

- **Linear Discriminant Analysis:**

The analysis of the conditioning components using an LSTM model with 48 batch sizes and linear discriminant analysis showed strong performance across different components. The Cooler component achieved perfect scores with an accuracy, precision, recall, and F1-score of 1.0. The Valve component exhibited lower performance with an accuracy of 0.861678 and corresponding precision, recall, and F1-score values around 0.908041. The Internal Pump Leakage component also showed slightly lower performance with metrics around 0.986395. The Accumulator component demonstrated lower performance with an accuracy of 0.820862 and corresponding precision, recall, and F1-score values around 0.850862. The Stable Flag component had slightly lower scores with an accuracy of 0.963719 and similar values for other metrics. The ROC curves for these components further validate these findings, showing high areas under the curve (AUC) values close to 1.0 for all classes, indicating strong predictive performance across all components.

Table 6.27: LSTM 48 Batch Size LDA

Conditioning Component	Accuracy	Precision	Recall	F1-Score
Cooler	1.0	1.0	1.0	1.0
Valve	0.861678	0.967279	0.861678	0.908041
Internal Pump Leakage	0.986395	0.986536	0.986395	0.986433
Accumulator	0.820862	0.904032	0.820862	0.850862
Stable Flag	0.963719	0.963719	0.963719	0.963719

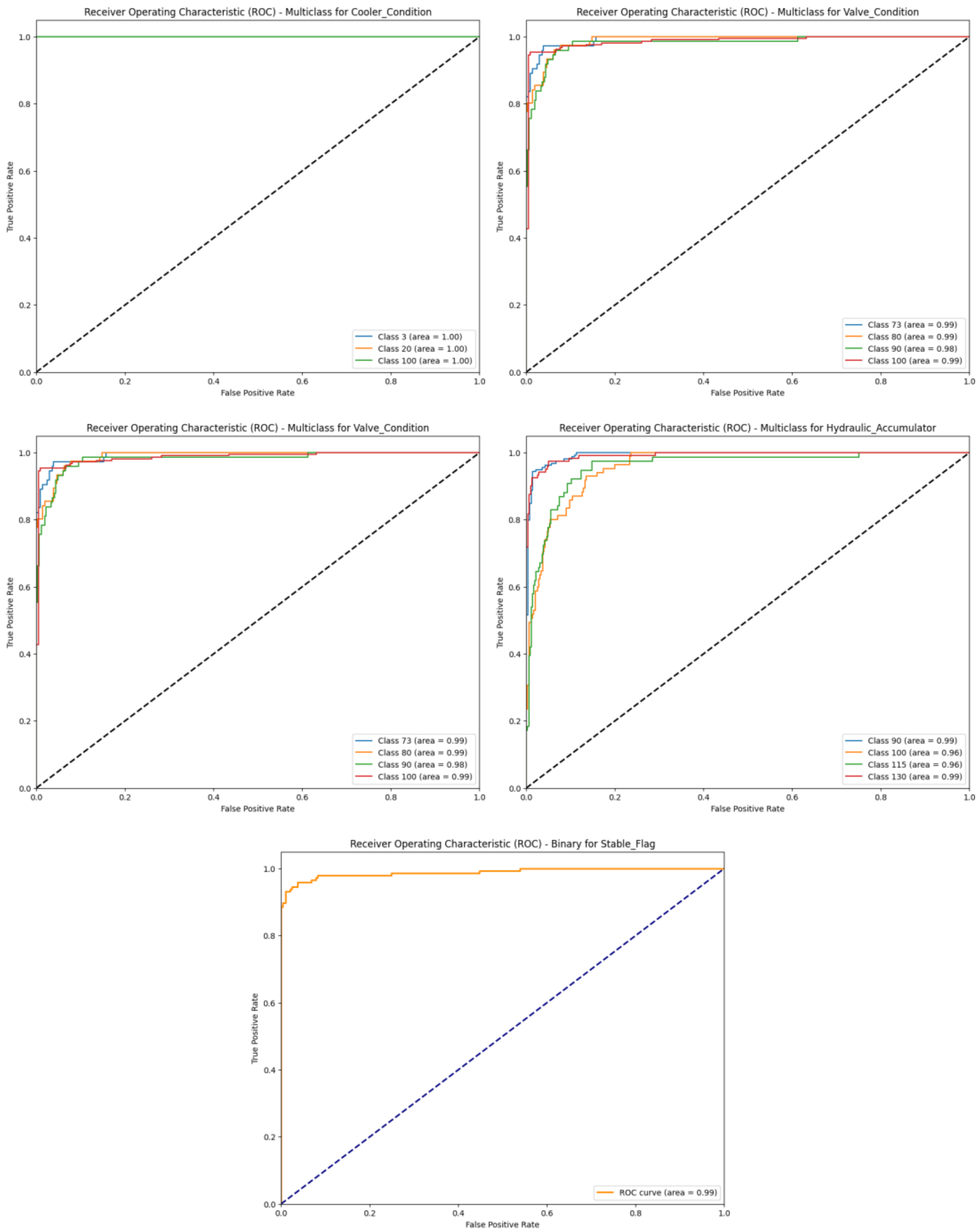


Figure 6.27 ROC AUC LDA (LSTM) – Phase 3

6.4 Results and Findings Summary

Figure 6.28 presents a bar chart comparing the performance of a Random Forest model across three phases with different numbers of estimators (100, 200, and 300) without feature selection. The chart shows that the Cooler component consistently achieves the highest performance, with the tallest bars across all phases. The Valve, Internal Pump Leakage, and Accumulator components also demonstrate high performance with only minor variations in bar height. The Stable Flag component has slightly shorter bars, indicating comparatively lower performance, but remains consistent across the phases.

Overall, the model performs robustly with minimal changes in performance across the different estimator numbers.

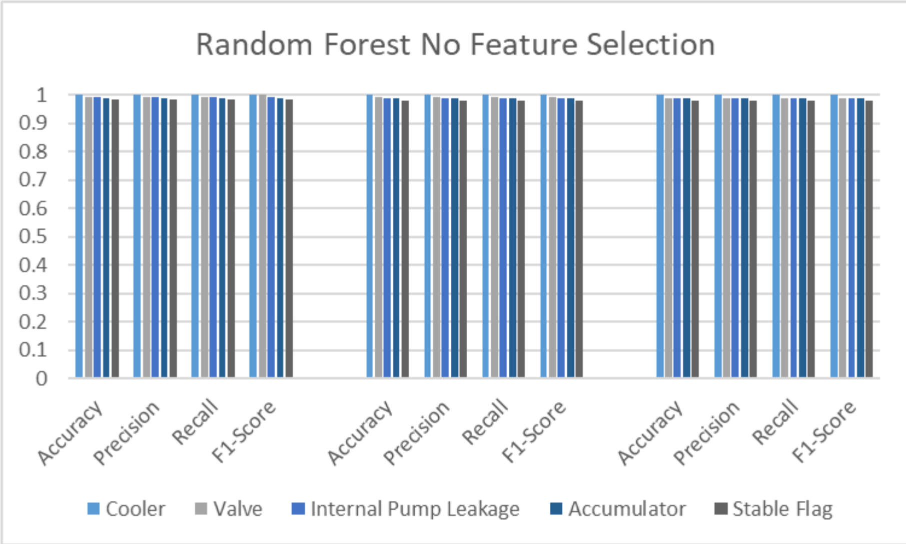


Figure 6.28 Random Forest Performance Comparison (No Feature Selection)

Figure 6.29 presents a bar chart comparing the performance of a Random Forest model across three phases with different numbers of estimators (100, 200, and 300) using Principal Correlation Analysis (PCA). The chart shows that in Phase 1, the Cooler component achieves the highest performance, indicated by the tallest bars, while the other components, such as the Valve, Internal Pump Leakage, Accumulator, and Stable Flag, have slightly shorter bars. In Phase 2, the Cooler component reaches perfect performance, with the tallest bars across all metrics, while the other components maintain high performance with minor variations. In Phase 3, the Cooler component continues to achieve perfect scores. Valve, Internal Pump Leakage, Accumulator, and Stable Flag components exhibit consistent performance across all phases, as indicated by the relatively stable bar heights.

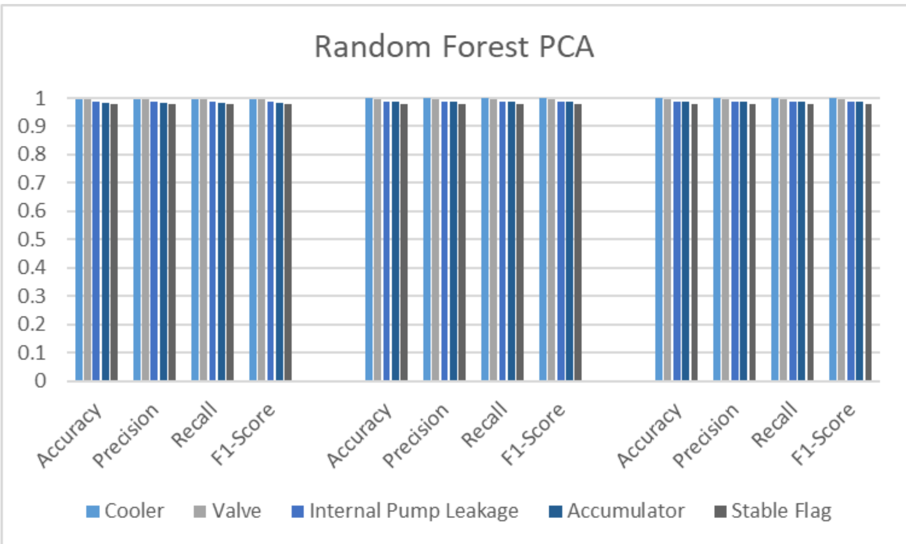


Figure 6.29 Random Forest Performance Comparison (PCA)

Figure 6.30 presents a bar chart comparing the performance of a Random Forest model across three phases with different numbers of estimators (100, 200, and 300) using Linear Discriminant Analysis (LDA). In Phase 1, the bars for the Cooler component are the tallest, indicating the highest performance, while the Valve, Internal Pump Leakage, Accumulator, and Stable Flag components have slightly shorter bars. In Phase 2, the Cooler component reaches perfect performance, with the tallest bars across all metrics, and other components such as the Valve, Internal Pump Leakage, and Accumulator also show improved performance with slightly taller bars. In Phase 3, the Cooler component maintains its perfect performance. Internal Pump Leakage, Accumulator, and Stable Flag components exhibit consistent performance with stable bar heights across all phases.

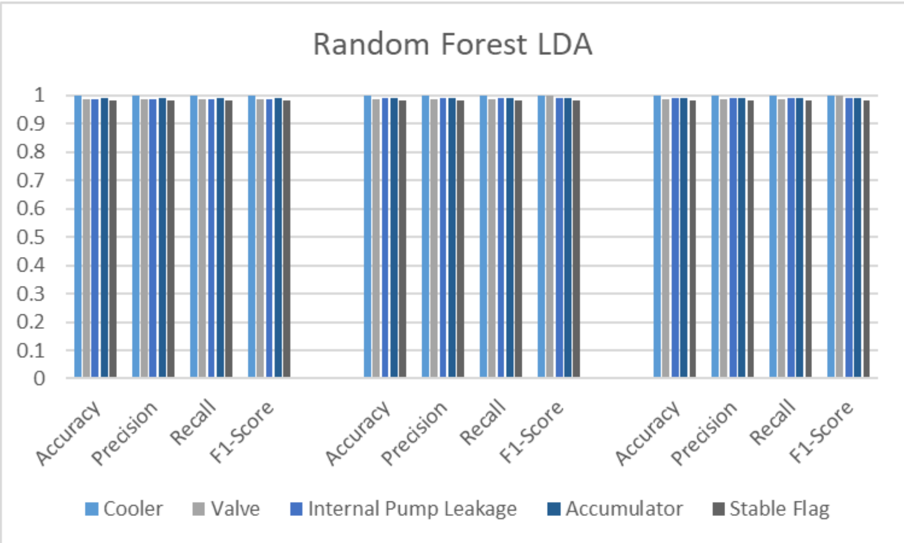


Figure 6.30 Random Forest Performance Comparison (LDA)

Figure 6.31 presents a bar chart comparing the performance of a Catboost model across three phases with different numbers of estimators (100, 200, and 300) with no feature selection. The chart shows that the Cooler and Valve components achieve perfect performance across all phases, indicated by the tallest bars. The Internal Pump Leakage component maintains high performance with consistently tall bars. The Accumulator component has the shortest bars, indicating relatively lower performance. The Stable Flag component exhibits high performance. Overall, the chart highlights the robust performance of the Catboost model, particularly for the Cooler and Valve components, with other components maintaining high and stable performance.

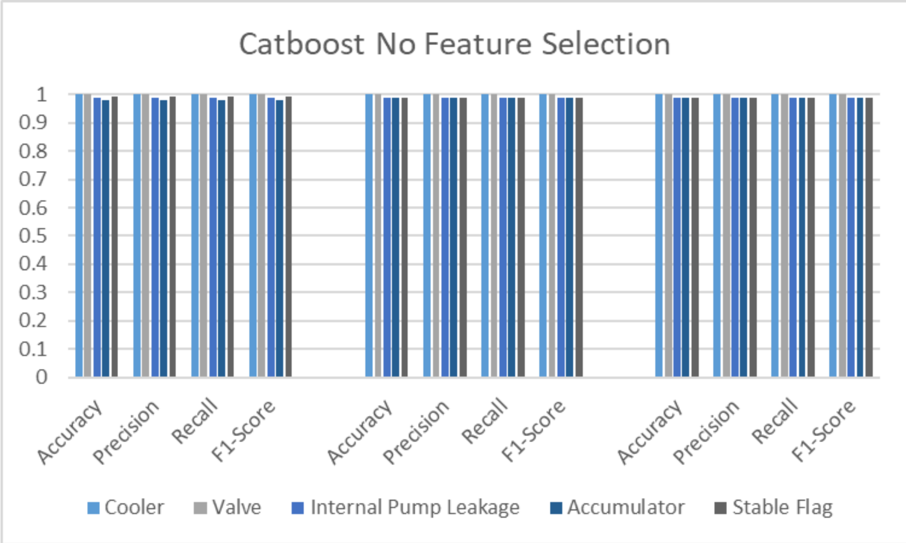


Figure 6.31 Catboost Performance Comparison (No Feature Selection)

Figure 6.32 presents a bar chart comparing the performance of a Catboost model across three phases with different numbers of estimators (100, 200, and 300) using Pearson Correlation Analysis. The chart shows that the Cooler and Valve components achieve perfect performance across all phases, indicated by the tallest bars. The Internal Pump Leakage component maintains high performance with consistently tall bars. The Accumulator component has shorter bars, indicating relatively lower performance. The Stable Flag component exhibits high performance with stable bar heights. Overall, the chart highlights the robust performance of the Catboost model, particularly for the Cooler and Valve components, with other components maintaining high and stable performance.

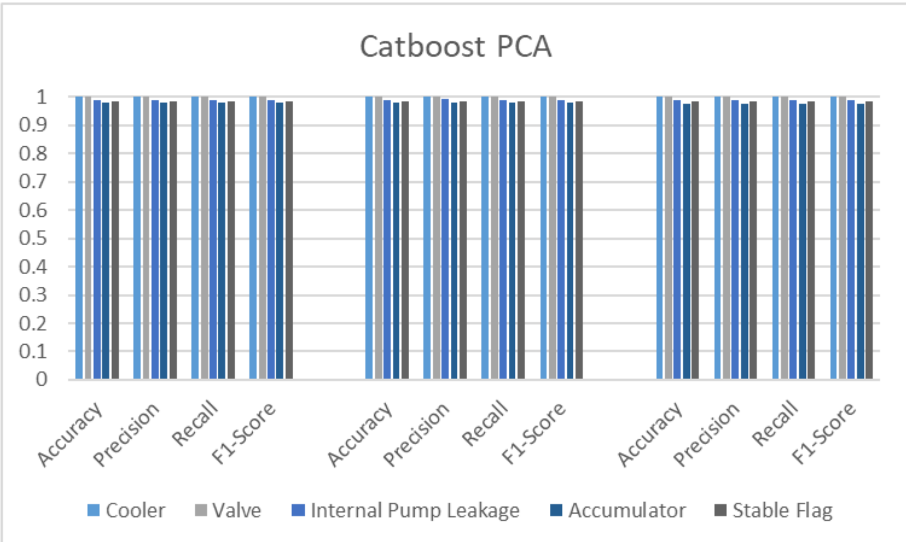


Figure 6.32 Catboost Performance Comparison (PCA)

Figure 6.33 presents a bar chart comparing the performance of a Catboost model across three phases with different numbers of estimators (100, 200, and 300) using Linear Discriminant Analysis (LDA). The chart shows that the Cooler component consistently achieves perfect performance across all phases, indicated by the tallest bars. The Valve component maintains high performance with stable bar heights across all phases. The Internal Pump Leakage component exhibits high performance with slight improvements in Phase 3. The Accumulator component shows high performance with shorter bars compared to the Cooler component but remains consistent across all phases. The Stable Flag component has the shortest bars, indicating comparatively lower performance. Overall, the chart highlights the robust performance of the Catboost model, particularly for the Cooler component, with other components maintaining high and stable performance.

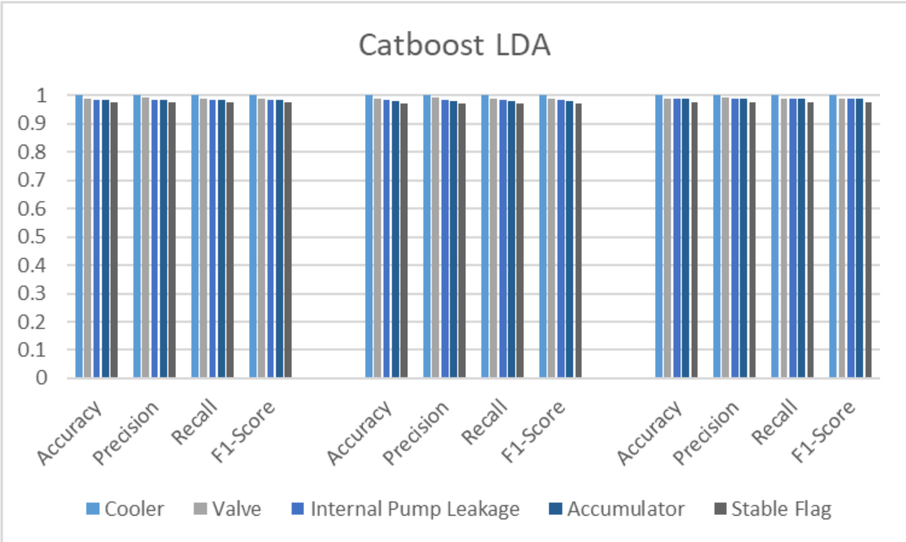


Figure 6.33 Catboost Performance Comparison (LDA)

Figure 6.34 compares the performance of an LSTM model without feature selection across three phases with different batch sizes (16, 32, and 48). In each phase, the model's performance metrics—accuracy, precision, recall, and F1-score—are evaluated for various conditioning components. The Cooler component consistently achieves perfect scores across all metrics and phases. The Valve and Internal Pump Leakage components show high performance, with slight variations in metrics across phases. The Accumulator and Stable Flag components exhibit more variability, particularly in the Accumulator's performance, which decreases with larger batch sizes.

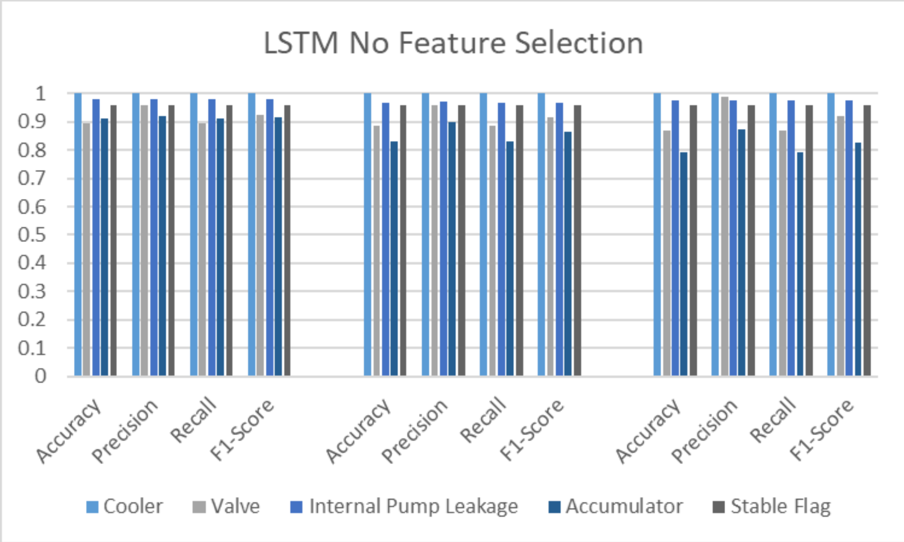


Figure 6.34 LSTM Performance Comparison – (No Feature Selection)

Figure 6.35 illustrates the LSTM model's performance with Pearson correlation analysis across three phases with batch sizes of 16, 32, and 48. Each phase evaluates the model on various conditioning components using metrics such as accuracy, precision, recall, and F1-score. The Cooler component consistently achieves perfect scores across all metrics in every phase. The Valve component shows high but variable performance, with the highest precision in the third phase. The Internal Pump Leakage component maintains strong and stable performance across phases. In contrast, the Accumulator component exhibits significant performance variability, particularly in phases two and three, where it shows a noticeable drop in metrics. The Stable Flag component consistently performs well, with only minor fluctuations across different batch sizes.

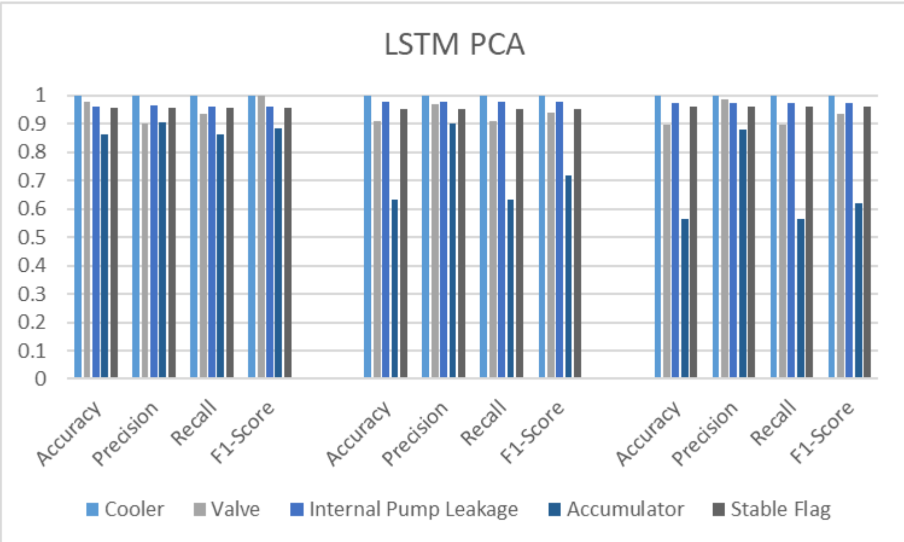


Figure 6.35 LSTM Performance Comparison (PCA)

Figure 6.36 presents the performance comparison of the LSTM model using Linear Discriminant Analysis (LDA) across three phases with batch sizes of 16, 32, and 48. Each phase evaluates key conditioning components—Cooler, Valve, Internal Pump Leakage, Accumulator, and Stable Flag—using metrics such as accuracy, precision, recall, and F1-score. The Cooler component consistently achieves perfect scores in all metrics across every phase. The Valve component shows high performance, with the highest precision but variable recall, particularly dropping in the third phase. The Internal Pump Leakage component maintains excellent and stable performance, with all metrics close to perfect across phases. The Accumulator component displays more variability, with its performance declining notably in the third phase. The Stable Flag component maintains consistently high performance across all phases with only minor fluctuations.

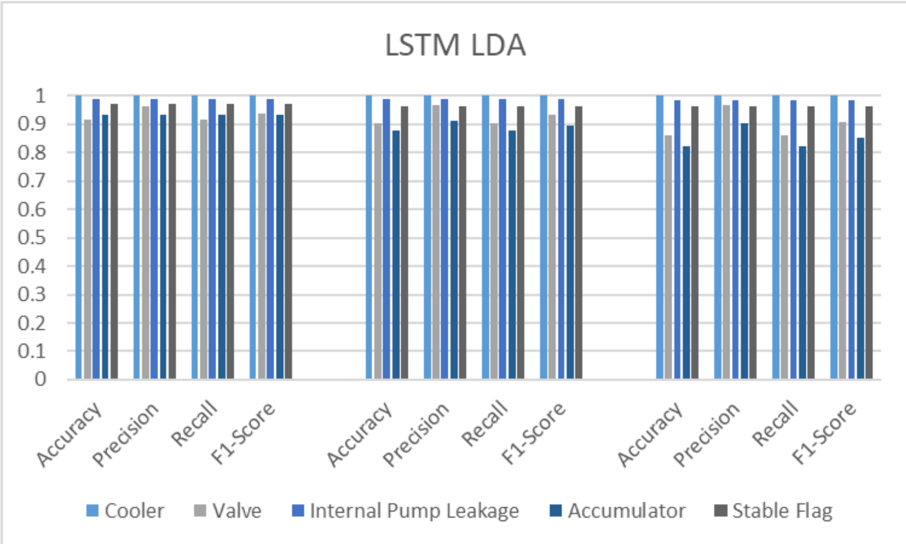


Figure 6.36 LSTM Performance Comparison (LDA)

The testing phase systematically evaluated the performance of the machine and deep learning models by tuning hyperparameters across three phases, focusing on the number of estimators for Random Forest and Catboost models and batch sizes for LSTM models. Each phase involved testing without feature selection, using Pearson Correlation Analysis, and employing Linear Discriminant Analysis (LDA).

Random Forest models, tested with 100, 200, and 300 estimators, consistently exhibited strong performance, particularly for the Cooler component, which achieved perfect scores (accuracy, precision, recall, F1-score of 1.0) in all scenarios. The Valve, Internal Pump Leakage, and Accumulator components also performed well, with scores typically above 0.99, and the Stable Flag component showed a slightly lower but still robust performance of around 0.98. Catboost models similarly demonstrated high performance, with the Cooler and Valve components achieving perfect scores across all phases and feature selection methods. The Internal Pump Leakage and Accumulator components maintained high scores (~0.98-0.99), while the Stable Flag component showed slightly lower but consistent performance. LSTM models were evaluated with batch sizes of 16, 32, and 48 and displayed more variability in performance. The Cooler component consistently achieved perfect scores, but the Valve and Accumulator components showed noticeable declines in performance at larger batch sizes, with accuracy dropping as low as 0.86 for Valve and 0.56 for Accumulator in certain scenarios. The Stable Flag component generally maintained strong performance across all phases.

Feature selection methods, including Pearson Correlation Analysis and LDA, generally enhanced or stabilised model performance, with slight improvements observed in some components. Overall, the testing phase highlighted the robustness and high predictive capabilities of Random Forest and Catboost models while identifying areas for improvement in LSTM models, particularly in handling larger batch sizes.

6.5 Dashboard

Using the dataset provided by ZeMA gGmbH a dashboard is created to display the charts of the condition components of the hydraulic system. The time column was manually added by calculating it. The calculation was done by the information provided by the dataset. Each cycle takes 60 seconds which produces a record and in total there are 2205 records. Here is the total time the test rig was running for:

$$
\begin{aligned}
\text{Total Time} &= 60 \text{ seconds} * 2205 \\
&= 132,300 \text{ seconds} \\
&= 2205 \text{ minutes} \\
&= 36 \text{ hours and } 45 \text{ minutes} \\
&= 1 \text{ day } 12 \text{ hours } 45 \text{ minutes}
\end{aligned}
$$

Figure 6.37 displays the "Overall Performance" page of the Hydraulic System Equipment Maintenance dashboard. The page is divided into two primary sections: the "Overall Stability of Hydraulic System" and the "Performance Valve Condition." The "Overall Stability of Hydraulic System" graph depicts the system's stability over time, with the stability flag fluctuating between 0 (stable conditions) and 1 (conditions potentially not stable). The stability data is plotted across a timeline highlighting periods of stability and instability. The "Performance Valve Condition" graph shows the valve condition percentage over the same timeline, where the valve condition ranges from 100% (optimal switching behaviour) to 73% (close to total failure). This section allows for monitoring the valve's performance, indicating periods of small lag, severe lag, and near-total failure. As seen in Figure 7.1, the relationship between the charts lies in their synchronized timelines, allowing for simultaneous comparison of system stability and valve performance. By examining these charts together, one can observe how fluctuations in valve condition is correlated with periods of instability in the hydraulic system.

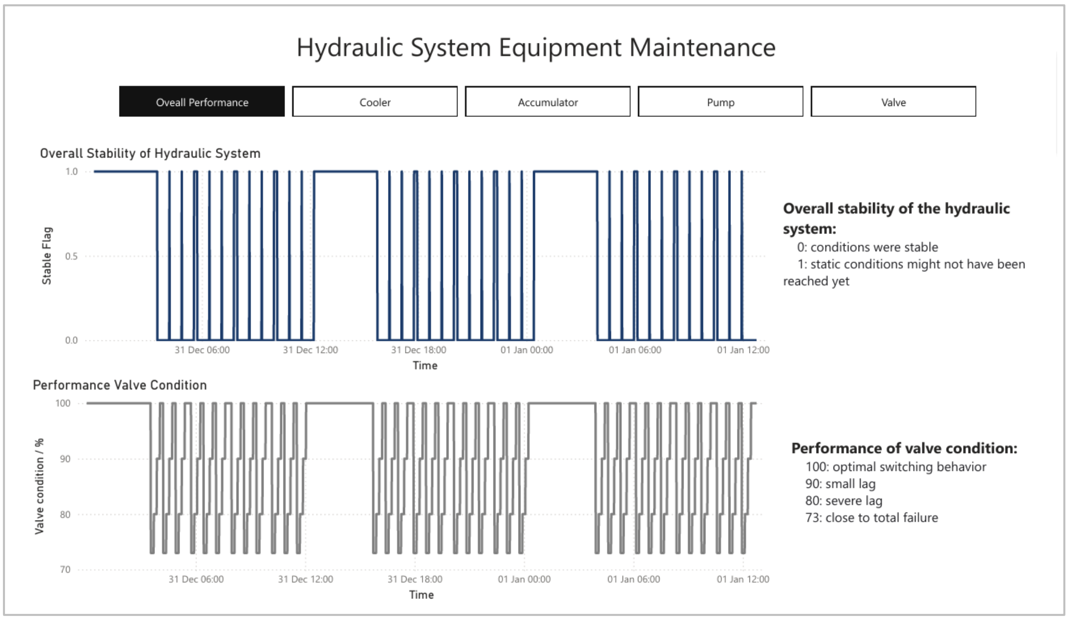


Figure 6.37 Overall Performance Page

Figure 6.38 presents the "Cooler Performance" page of the Hydraulic System Equipment Maintenance dashboard. This page is organized into multiple sections to provide a detailed analysis of the cooler's performance over a specified timeframe. The top-left chart, "Performance of Cooler Condition," shows the cooler condition percentage, indicating the cooler's efficiency levels over time, with values ranging from 3% (close to total failure) to 100% (full efficiency). The chart shows significant improvement in cooler conditions starting from around midnight on January 1st. The relationship among these charts is evident in the synchronization of their timelines. Improvements in the cooler condition and cooling efficiency coincide with increased cooling power and a distinct pattern in pressure sensor readings.

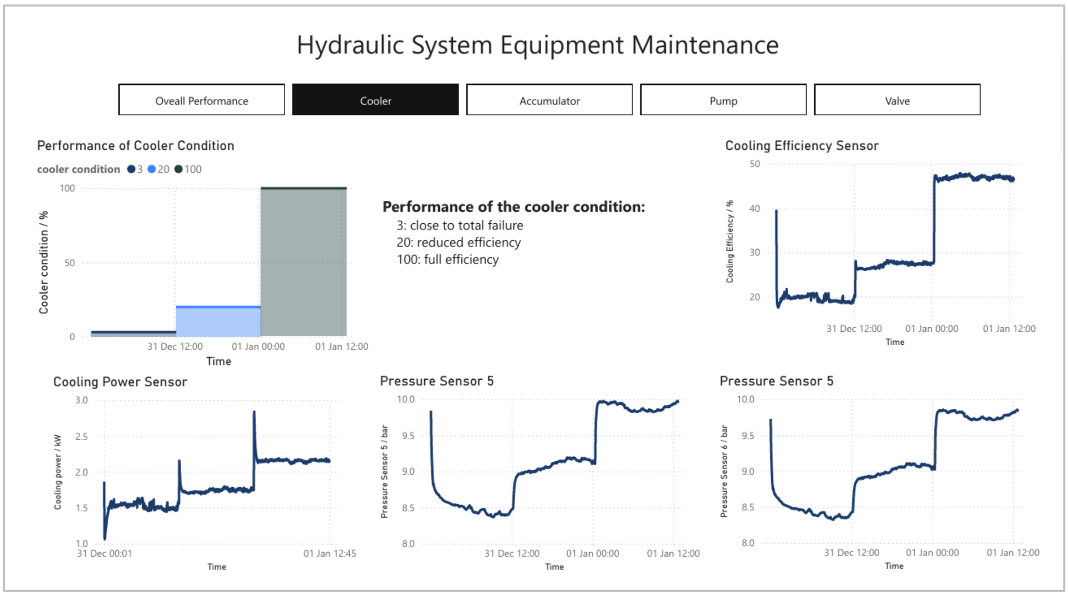


Figure 6.38 Cooler Performance Page

Figure 6.39 illustrates the "Accumulator Performance" page of the Hydraulic System Equipment Maintenance dashboard. This page focuses on monitoring the performance of the hydraulic accumulator, a crucial component in maintaining the system's pressure balance. "Performance of Hydraulic Accumulator," displays the accumulator's pressure over time, measured in bar. The pressure values range from 130 bar (indicating optimal pressure) to 90 bar (indicating a state close to total failure). The timeline spans from December 31st to January 1st, showcasing fluctuations in pressure levels throughout this period. The chart reveals a cyclic pattern in the accumulator's performance, where the pressure repeatedly drops from optimal levels (130 bar) to significantly lower levels (around 90 bar) before recovering. These fluctuations suggest periods of reduced efficiency and potential system stress. By observing the pressure trends, maintenance personnel can identify critical periods where the accumulator's pressure deviates from the optimal range, signalling the need for intervention or further investigation.

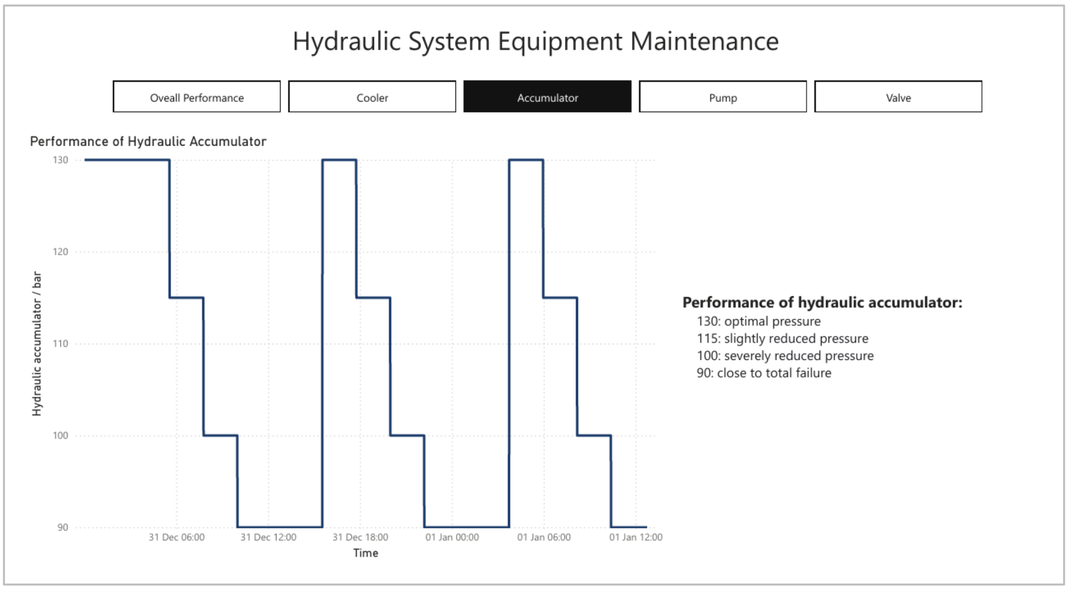


Figure 6.39 Accumulator Performance Page

Figure 6.40 presents the "Internal Pump Performance" page of the Hydraulic System Equipment Maintenance dashboard. This page is dedicated to monitoring the performance of the internal pump within the hydraulic system, specifically focusing on the leakage levels. The main chart, titled "Performance of the Internal Pump," displays the internal pump leakage over time, measured on a scale from 0 to 2. A value of 0 indicates no leakage, 1 represents weak leakage, and 2 signifies severe leakage. The chart shows a repetitive pattern where the leakage alternates between 0 (no leakage) and 2 (severe leakage), indicating a recurring issue with the internal pump's. This cyclical pattern suggests periods where the pump is functioning correctly, followed by intervals of severe leakage, which could indicate potential maintenance issues or the need for further inspection. By analysing the fluctuations in leakage levels, maintenance personnel can identify critical times when the pump's performance deviates from normal, enabling them to take measures to address the underlying issues.

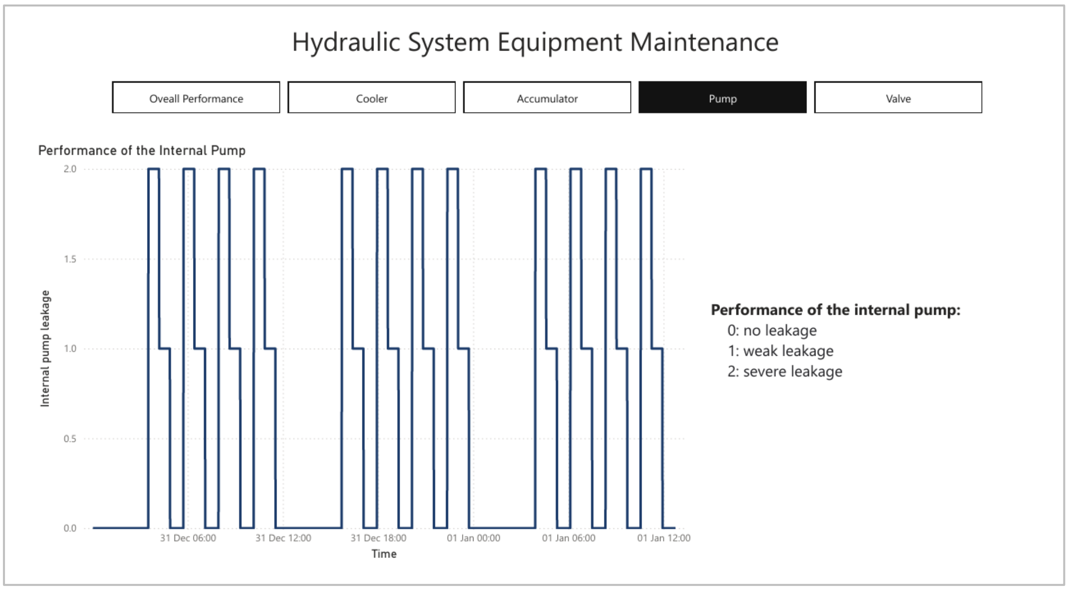


Figure 6.40 Internal Pump Performance Page

Figure 6.41 illustrates the "Valve Performance" page of the Hydraulic System Equipment Maintenance dashboard. This page is designed to provide detailed insights into the performance of the valve within the hydraulic system. "Performance Valve Condition," displays the valve condition as a percentage over time. The performance is measured from 70% to 100%, where 100% indicates optimal switching behaviour, 90% represents small lag, 80% signifies severe lag, and 73% indicates a state close to total failure. The chart shows periodic fluctuations in valve performance, with the condition oscillating between approximately 75% and 100%. This pattern suggests recurring issues with the valve's performance, transitioning from optimal to near-failure states repeatedly. By analysing the periodic dips in performance, maintenance personnel can identify critical times when the valve condition deteriorates, which may require immediate attention or scheduled maintenance.

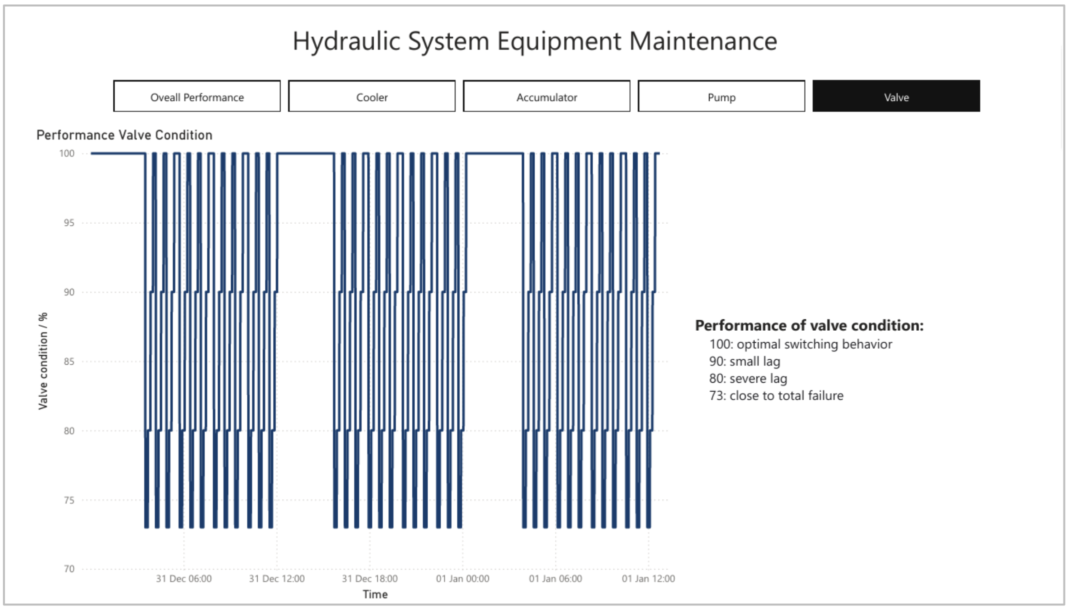


Figure 6.41 Valve Performance Page

Chapter 7: Conclusion

This research project consists of six chapters that comprehensively analyse predictive maintenance for hydraulic systems using machine learning and deep learning techniques. The project reviewed the background research, presented the theoretical framework and proposed a research methodology to construct the predictive model. The project began with an introduction in Chapter 1, providing an overview of the research topic, problem statement, aim, objectives, scope, and significance. Chapter 2 included a detailed literature review of relevant papers on hydraulic systems, condition monitoring, predictive maintenance and previous research. The theoretical framework in Chapter 3 includes the CRISP-DM process model and its six stages. Chapter 4 outlined the research methodology, including data collection, data understanding, data preparation, modelling, evaluation and deployment. The implementation plan for the project was described in detail in Chapter 5, including data understanding, data preparation and future work. The present study aimed to build a predictive model that forecasts the maintenance requirements of a hydraulic system and identifies the patterns of decline in sensors and conditions variables. Chapter 6 detailed the testing phase, which systematically evaluated the performance of machine learning and deep learning models, such as Random Forest (RF), Catboost, and Long Short-Term Memory (LSTM), through hyperparameter tuning and feature selection methods. The results highlighted that RF and Catboost models exhibited strong and consistent performance, particularly for the Cooler component, while LSTM models showed more variability. Feature selection methods like Pearson Correlation Analysis and Linear Discriminant Analysis generally improved model performance. Chapter 7 presented the deployment of a Power BI dashboard. Based on the literature review, the project selected the most suitable data preparation methods and machine learning algorithms for the modelling stage. To conclude, this project demonstrated the feasibility and importance of using machine learning and deep learning techniques for the predictive maintenance of hydraulic systems and a Power BI dashboard. The project's results will contribute to improving condition monitoring and maintenance engineering.

References

Askari, B., Cavone, G., Carli, R., Grall, A., & Dotoli, M. (2023, August 26). A semi-supervised learning approach for fault detection and diagnosis in complex mechanical systems. *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*. http://dx.doi.org/10.1109/case56687.2023.10260469

Beebe, R. S. (2004). Predictive maintenance of pumps using condition monitoring. Elsevier.

Berghout, T., Benbouzid, M., Muyeen, S. M., Bentrcia, T., & Mouss, L.-H. (2021). Auto-NAHL: A neural network approach for condition-based maintenance of complex industrial systems. *IEEE Access*, *9*, 152829–152840. https://doi.org/10.1109/access.2021.3127084

Buabeng, A., Simons, A., Frempong, N. K., & Ziggah, Y. Y. (2023). Hybrid intelligent predictive maintenance model for multiclass fault classification. *Soft Computing*. https://doi.org/10.1007/s00500-023-08993-1

Chekkala, V. (2020). Predictive Maintenance for Fault Diagnosis and Failure Prognosis in Hydraulic System.

Chen, C.-H., Chan, Y.-K., & Yu, S.-S. (2023, June). Hydraulic system failure prediction method with limited failure data. *2023 Sixth International Symposium on Computer, Consumer and Control (IS3C)*. http://dx.doi.org/10.1109/is3c57901.2023.00053

Das, S. (2022). Artificial intelligence in highway safety. CRC Press.

Gaurkar, S., Kotalwar, A., & Gabale, S. (2021). Predictive Maintenance of Industrial Machines using Machine Learning. *International Research Journal of Engineering and Technology (IRJET)*, *8*(11).

Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation. Departmental Technical Reports (CS). 1209. https://digitalcommons.utep.edu/cgi/viewcontent.cgi?article=2202&context=cs_techrep

Graves, A. (2012). Supervised sequence labelling with recurrent neural networks. Springer.

Guo, P., Wu, J., Xu, X., Cheng, Y., & Wang, Y. (2019, October). Health condition monitoring of hydraulic system based on ensemble support vector machine. *2019 Prognostics and System Health Management Conference (PHM-Qingdao)*. http://dx.doi.org/10.1109/phm-qingdao46334.2019.8942981

Helwig, N., Pignanelli, E., & Schutze, A. (2015, May). Condition monitoring of a complex hydraulic system using multivariate statistics. *2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*. http://dx.doi.org/10.1109/i2mtc.2015.7151267

IBM documentation. (n.d.). Retrieved January 29, 2024, from https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview

Keleko, A. T., Kamsu-Foguem, B., Ngouna, R. H., & Tongne, A. (2023). Health condition monitoring of a complex hydraulic system using Deep Neural Network and DeepSHAP explainable XAI. *Advances in Engineering Software*, *175*, 103339. https://doi.org/10.1016/j.advengsoft.2022.103339

Khan, H. A., Bhatti, U., Kamal, K., Alkahtani, M., Abidi, M. H., & Mathavan, S. (2023). Fault classification for cooling system of hydraulic machinery using AI. *Sensors*, *23*(16), 7152. https://doi.org/10.3390/s23167152

Kim, D., & Heo, T.-Y. (2022). Anomaly detection with feature extraction based on machine learning using Hydraulic System IoT Sensor Data. *Sensors*, *22*(7), 2479. https://doi.org/10.3390/s22072479

Kim, K., & Jeong, J. (2022, July). Multi-layer stacking ensemble for fault detection classification in hydraulic system. *2022 26th International Conference on Circuits, Systems, Communications and Computers (CSCC)*. http://dx.doi.org/10.1109/cscc55931.2022.00066

Levitt, J. (2003). Complete guide to preventive and predictive maintenance. Industrial Press Inc.

Liu, J., Xu, H., Peng, X., Wang, J., & He, C. (2023). Reliable composite fault diagnosis of hydraulic systems based on linear discriminant analysis and multi-output hybrid kernel extreme learning machine. *Reliability Engineering & System Safety*, *234*, 109178. https://doi.org/10.1016/j.ress.2023.109178

Loh, W. (2011). Classification and regression trees. *WIREs Data Mining and Knowledge Discovery*, *1*(1), 14–23. https://doi.org/10.1002/widm.8

Ma, X., Wang, P., Zhang, B., & Sun, M. (2021). A multi-rate sensor information fusion strategy for multitask fault diagnosis based on convolutional neural network. *Journal of Sensors*, *2021*, 1–17. https://doi.org/10.1155/2021/9952450

Mallak, A., & Fathi, M. (2021). Sensor and component fault detection and diagnosis for hydraulic machinery integrating LSTM autoencoder detector and diagnostic classifiers. *Sensors*, *21*(2), 433. https://doi.org/10.3390/s21020433

Peng, Z., Zhang, K., & Chai, Y. (2020, July). Multiple fault diagnosis for hydraulic systems using Nearest-centroid-with-DBA and Random-Forest-based-time-series-classification. *2020 39th Chinese Control Conference (CCC)*. http://dx.doi.org/10.23919/ccc50068.2020.9189401

Quatrini, E., Costantino, F., Pocci, C., & Tronci, M. (2020). Predictive model for the degradation state of a hydraulic system with dimensionality reduction. *Procedia Manufacturing*, *42*, 516–523. https://doi.org/10.1016/j.promfg.2020.02.039

Schneider, T., Klein, S., & Bastuck, M. (2018, April 26). *Condition monitoring of hydraulic systems Data Set at ZeMA*. Zenodo. https://zenodo.org/records/1323611

Skinner, S. (2014). Hydraulic fluid power - A historical timeline. Lulu.com.

Soh, J., & Kim, D. (2021, October 31). Condition monitoring with time series data based on probabilistic model. *2021 24th International Conference on Electrical Machines and Systems (ICEMS)*. http://dx.doi.org/10.23919/icems52562.2021.9634481

Wu, J., Guo, P., Cheng, Y., Zhu, H., Wang, X.-B., & Shao, X. (2020). Ensemble generalized multiclass support-vector-machine-based health evaluation of complex degradation systems. *IEEE/ASME Transactions on Mechatronics*, *25*(5), 2230–2240. https://doi.org/10.1109/tmech.2020.3009449

Xu, Z., Yu, H., Tao, J., & Liu, C. (2021). Compound fault diagnosis in hydraulic system with multi-output SVM. *CSAA/IET International Conference on Aircraft Utility Systems (AUS 2020)*. http://dx.doi.org/10.1049/icp.2021.0470

Yugapriya, M., Judeson, A. K. J., & Jayanthy, S. (2022, March 16). Predictive Maintenance of Hydraulic System using Machine Learning Algorithms. *2022 International Conference on Electronics and Renewable Systems (ICEARS)*. http://dx.doi.org/10.1109/icears53579.2022.9751840

Zhang, C., & Ma, Y. (2012). *Ensemble machine learning: Methods and applications*. Springer Science & Business Media.

Zhang, P., Hu, W., Cao, W., Chen, L., & Wu, M. (2022, May 4). Multi-Fault diagnosis of hydraulic systems based on fully convolutional networks. *2022 13th Asian Control Conference (ASCC)*. http://dx.doi.org/10.23919/ascc56756.2022.9828182

Zhang, Q. (2019). Basics of Hydraulic Systems, second edition. CRC Press.

APPENDICES

Leave this box blank

Evaluating Machine Learning and Deep Learning Algorithms for Predictive Maintenance of Hydraulic Systems

Ayat Al-Khulaqi[#], Naveen Palanichamy [#], Su-Cheng Haw [#]

[#] *Faculty of Computing and Informatics, Multimedia University, Persiaran Multimedia, 63100, Cyberjaya, Selangor, Malaysia*
E-mail: p.naveen@mmu.edu.my

***Abstract*— Hydraulic systems are critical in industries like aerospace and petroleum. However, equipment degradation can lead to failures over time, resulting in costly downtime. Condition monitoring and predictive maintenance can be implemented to predict the equipment's failure before the machine's total failure. Current data-driven methods for predicting faults in hydraulic systems are insufficient due to inaccurate predictions. Our primary objective is to investigate the potential of different classifier models' predictive capabilities in enhancing the reliability of hydraulic systems. This paper implements two machine learning models, random forest (RF) and categorical boost (Catboost), and one deep learning model, long-short-term memory (LSTM), to predict the maintenance needs of hydraulic system equipment using the data from ZeMA gGmbH. The results of the models are evaluated through different metrics, including Precision, Recall, F1-Score, and Accuracy. The outcomes of the conducted experiments validate the paramount importance of the RF model, which has proven to be the most efficient and successful in accurately predicting instances of equipment failure prior to the occurrence of total system failure. Across critical hydraulic system condition components revealed their varying performance across different components, with LSTM excelling in predictiveness of the valve, RF dominating Pump predictions, and overall reliability observed for Accumulator and Stable Flag. The experimental findings demonstrate that the proposed method for predicting the state of hydraulic systems outperforms alternative approaches.**

***Keywords*— Hydraulic systems, Condition monitoring, Predictive maintenance, Machine learning, Deep Learning**

I. Introduction

Hydraulic systems are used in numerous industries, including the oil and gas industry, air transport, construction, mobile vehicles, and factory equipment. Hence, many of the installations/mechanical components used in these applications need frequent service to continue operating flawlessly. As an innovative approach within industries, predictive maintenance (PdM), through its capability to detect potential equipment breakdown through sensors embedded in them, forms part of the Internet of Things (IoT). The conventional approach to machine maintenance costs a lot and mainly responds to breakdowns. Machine Learning (ML) and Deep Learning (DL) algorithms are used for predicting equipment failures, which have shown promising results in predicting them before they occur. This evolution includes increased operational efficiency, decreased downtime, and better practice in predictive maintenance [1].

Condition monitoring (CM) is the initial step or act that should be done to look for possible conditions and predict maintenance to stop machines from being down for a long time and reduce maintenance expenses [2]. It involves sampling process data, such as temperature, pressure, and so forth, through sampling equipment connected to a computer system [3]. The use of CM is an essential aspect of PdM. PdM entails continuous monitoring of machines for their health status and working conditions, which can lead to quality improvement of products, increased output rates and overall factory performance [4].

Hydraulic systems consist of five conditioning components: a cooler, which manages the fluid temperature and prevents overheating; the valve controls fluid flow and direction; the pump provides the necessary pressure to circulate fluid; the accumulator stores energy and compensates for fluid leakage; and the stable flag serves as an overall health indicator. Previous studies have adopted diverse methodologies to conduct hydraulic system fault analysis, including ML, DL, or combining both. Some researchers have concentrated on the failure of the entire hydraulic system ([5]-[11]), while others have focused on specific hydraulic system conditions such as valves, pumps, coolers, accumulators, and stable flags ([2], [12]-[22]). Certain studies have aimed at diagnosing multiple faults and identifying different degrees of single faults ([23] -[24]). A significant portion of the existing literature focuses on identifying the failure of the entire machinery or four specific condition components. However, studies investigating the failure of all five condition components are relatively limited. This paper aims to bridge this gap by incorporating all five condition components that predict the maintenance of hydraulic systems. Also, to explore the predictive maintenance of various ML and DL models to improve the reliability of hydraulic systems.

The research is structured as follows: Section II reviews prior studies on ML and DL for fault analysis and predictive maintenance. Section III details the data sourcing, pre-processing, modelling, and evaluation process. Section IV presents comprehensive results and analyses. The final section, Section V, concludes the study, discussing limitations and future research directions.

II. Related Works

Hydraulic systems necessitate regular maintenance to ensure optimal performance. Implementing Predictive Maintenance (PdM) can significantly enhance the efficiency of these systems. By utilising Machine Learning (ML) or Deep Learning (DL) algorithms, it is possible to predict the degradation state of the conditioning components. In past research papers, some implemented ML, some DL, and others, under sections A, B, and C, further explain the researchers' proposed techniques.

A. Machine Learning

This paper proposes a technique to improve the fault tolerance and accuracy of a hydraulic system by using logistic regression, K-nearest Neighbor, decision tree, RF, and naïve Bayes to predict faults. The proposed technique is implemented using the Spyder IDE software tool on a Raspberry Pi 3 Model B+ controller [5]. The authors proposed a new design of multi-layer stacking ensemble models to enhance the fault detection of manufacturing plants through the use of data from hydraulic systems by combining five ML algorithms and LDA together for better classification performance than the traditional stacking ensemble methods [25]. Proposed for hydraulic system failure prediction enhancement was a Time-based Imbalanced Data Synthesis Technique (TIDS) process and an XGboost classifier used to generate time domain features from data and to synthesize minority samples to address data imbalance [7].

The authors found faults in the pneumatic system. They predicted them with a hybrid semi supervised learning model combined with traditional classification methods such as SVM, LR, DT, NB, and RF [9]. On the other hand, different research proposes a probability-based algorithm for analysing the time-series data of hydraulic systems and evaluating multiple conditions using Gaussian Mixture Model (GMM) for high accuracy [17]. The authors presented a method to diagnose various faults in the condition components of a hydraulic system based on principal component analysis and a multi-output, multi-class SVM for effective fault identification [19].

B. Deep Learning

One proposed data-driven approach focuses on deep neural networks (DNN) for multi-class classification degradation levels of each state of the hydraulic system [23]. The authors have described a strategy designed to diagnose many faults in hydraulic systems using time-series representations with FCN to acquire instantaneous features throughout multi-rate data [24]. The authors presented an advanced neural network model, Auto-NAHL, with automated hyperparameter tuning through Particle Swarm Optimization (PSO) for predicting maintenance in hydraulic systems [16]. Aa study focused on AI technique based on a Residual Network (ResNet-18) is presented for the high-detection classification of faulted cooling circuitry in hydraulic systems [26]. Researchers proposed a Multirate Sensor Information Fusion Strategy (MRSIFS) for fault diagnosis under multitask conditions, which would address the condition of the hydraulic system. The method proposed in this study uses multidimensional convolutional blocks and integrates multisource information fusion into the architecture of CNN [22].

C. Machine Learning and Deep Learning

A study focusing on predictive maintenance was conducted to diagnose faults and predict the condition of the components in hydraulic systems using LR, RF, ANN, LightGBM, and Catboost [12]. A web application was also developed to demonstrate the exploratory data analysis of the system’s condition. Researchers suggested a method for monitoring the health condition of hydraulic systems. Ensemble learning was used to improve the predictive precision of SVM classifiers, forming this method's essence. Besides, LDA and ANN were incorporated to compare the results against ensemble SVM [13]. A proposed method using machine and DL algorithms for fault detection and tolerance in each condition of the hydraulic systems. These algorithms include Logistic Regression, K Nearest Neighbor (KNN), Decision Tree, RF, and Naïve Bayes [2].

This includes the predictive study of industrial machine mechanical part conditions by using machine and deep learning algorithms, particularly Long Short-Term Memory (LSTM) and RF algorithms [14]. In another study, the authors presented an approach to monitoring the conditions of hydraulic systems using LDA, ANN, Linear SVM, and RBF SVM algorithms. The proposed technique applies multivariate statistics in sensor data analysis and fault detection. The data extracted from this is used in the training [15]. A proposed method for fault classification in hydraulic systems using a combination of Nearest Centroid (NC) with Dtw Barycenter Averaging (DBA) and RF algorithm to enhance the accuracy and speed of diagnosis [8]. A study on the predictive maintenance system on the innovative health assessment framework for hydraulic systems is undertaken using ensemble general multiclass support vector machines (EGMSVM) for stacking several GMSVMs as submodels and one RF as a metamodel [18].

In this work, the authors presented algorithms such as LDA, RF, GMSVM, E-SVMs, LSTM, 1D-CNN, and heterogeneous stacking. This paper proposed a technique in hydraulic system fault diagnosis based on multiple output classification. They combined Linear Discriminant Analysis and Hybrid Kernel Extreme Learning Machine to get high classification accuracy using MO-RPELM kernel. On the other hand, its authors compared performance in different models with MO-DSAE, MO-RF, MO-SVM, and MO-KNN, as shown in [20]. The present proposed hybrid artificial intelligence technique for predictive maintenance of the hydraulic system is composed of some algorithms like LSSVM, LDA SVM, and ANN, along with the combination of two additional feature selection techniques with ICEEMDAN-PCA or PCA without ICEEMDAN in combination with these algorithms [10].

A study was carried out on the anomaly detection system for the condition of hydraulic machinery signals, using eight algorithms: LDA, LR, SVC, DT, RF, XGBoost, LightGBM, and Multi-layer Perceptron, along with three feature selection methods in the form of Spearman's Rank Correlation Coefficient, Pearson Correlation Coefficient, and the Boruta Algorithm [21]. It studies the fault detection and diagnosis framework for hydraulic machinery with nine algorithms: LR, LDA, KNN, CART, NB, SVM, RF, CNN, and LSTM. The algorithms are implemented for four feature selections—Feature Importance (FI), RkSE, Time Domain Features, and Principal Component Analysis—without feature selection to compare different results [11].

TABLE I
MACHINE AND DEEP LEARNING ALGORITHMS

	[2]	[5]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[18]	[20]	[21]	[22]	[23]	[24]	[25]
LR	/	/		/		/	/							/				/
KNN		/				/												
DT		/	/	/										/				
RF		/	/	/		/	/		/			/		/				
NB		/		/		/												
ANN					/		/	/		/								
XGBOOST														/				/
LIGHTGBM							/							/				
CATBOOST							/											
LDA					/	/		/		/		/		/				
Ensemble SVM								/										
DNN																/		
NN	/																	
SVM	/			/	/	/												
DF	/																	
LSTM						/			/		/	/						
SVM (Linear)			/							/								
SVM (RBF)										/								
SVM (Gaussian)			/															
Softmax			/															
FCN																	/	
Auto-NAHL											/							
1D - CNN											/	/						
DBN											/							
GMM																		
GMSVM												/						
E-SVM												/						
EGMSVMs												/						
Heterogeneous Stacking												/						
MO-RPELM													/					
MO-DSAE													/					
MO-RF													/					
MO-SVM													/					
MO-KNN													/					
CNN						/									/			
LSVM					/													
SVC														/				
Multi-layer Perceptron														/				
CART						/												

To conclude, most authors implemented fault analysis on the degradation states of the hydraulic system condition components since monitoring the health is crucial. However, a comprehensive analysis of all five condition components is often overlooked. This study aims to fill this void, thereby maintaining energy efficiency and material savings and enhancing quality. In previous research papers, most implemented both ML and DL algorithms; in our research paper, the implementation of the ML model will be based on the most and least frequent ones: RF and Catboost. Catboost achieved high accuracy for all the hydraulic system conditions compared to other ML algorithms [12].

For DL, the most frequent will be implemented, where LSTM is chosen over ANN due to LSTM performing better in previous papers for predicting the degradation states of the hydraulic system conditions. For example, LSTM achieved high accuracy in predicting the conditions of the hydraulic system with results of cooler (100%), valve (95%), pump (99%), and accumulator (97%) [14]. In another research paper, LSTM achieved results for coolers (100%), valves (100%), pumps (100%), and accumulators (73%) [18]. Compared to ANN results of cooler (100%), valve (100%),15 pump (80%), and accumulator (50.4%) [15]. In another paper, ANN had better results due to the feature selection technique, which resulted in a cooler (100%), valve (97%), pump (94%), and accumulator (92%) [12].

This study presents a predictive maintenance framework that significantly advances from current methods. Most of the earlier studies concentrated on either one or a few condition components of the hydraulic systems, which may affect their predictive abilities. In contrast, our proposed approach consists of all five condition components of hydraulic systems, offering a comprehensive and integrated solution. Therefore, this paper aims to develop and test a predictive model for the degradation state of critical components of a hydraulic system using ML and DL algorithms. This approach bridges the gap in current predictive maintenance strategies and sets a new standard for hydraulic systems reliability improvement.

III. Research Methodology

The methodology proposed includes the ZeMA gGmbH dataset, data pre-processing, modelling, and evaluation of the hydraulic system's predictive maintenance; the methodology flow chart is shown in Fig. 1.

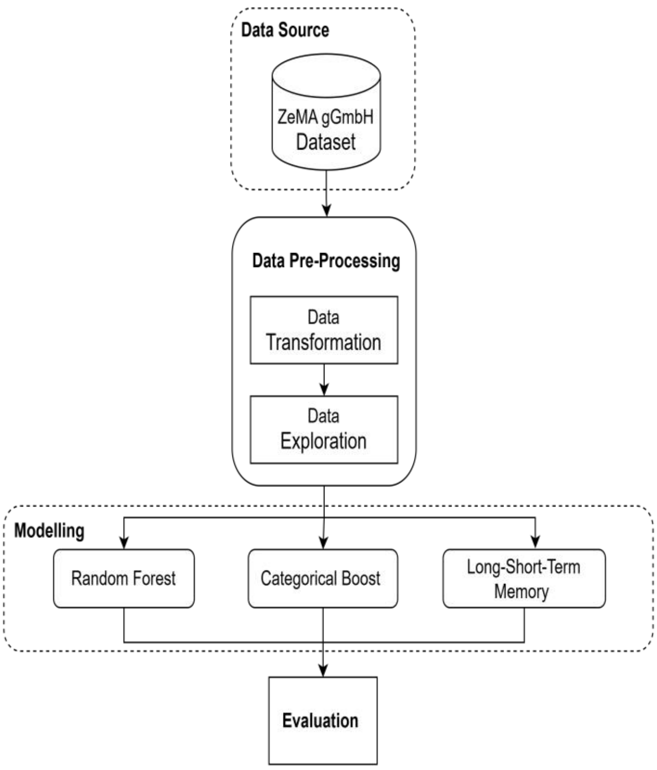


Fig. 1 Hydraulic System Methodology Chart

A. Data Source

The hydraulic system health condition data was sourced from a hydraulic test rig by the Centre for Mechatronics and Automation Technology (ZeMA), Saarbruecken, Germany [27]. The dataset encapsulates sensor data from the test rig, designed to simulate four distinct types of faults and system stability, each with varying severity levels. The test rig comprises a primary working circuit and a secondary cooling-filtration circuit interconnected through an oil tank. The working circuit, powered by main pump MP1 (with an electrical motor power of 3.3kw), undergoes cyclic repetition of different load levels regulated by the proportional pressure relief valve.

The system is designed to cyclically record process values from all sensors every 60 seconds and measures process values such as pressures, volume flows and temperatures. At the same time, the condition of five hydraulic components (cooler, valve, pump, accumulator, and stable flag) is quantitatively varied. The dataset contains 2205 instances, providing substantial data for analysis. It includes multiple feature types, such as multivariate time-series data with numerical and categorical attributes. It consists of raw sensor data in matrix form, where every row is a cycle while columns are the data points within a cycle.

The test rig was designed to emulate many fault scenarios, the details of which are shown in Table II. The test rig is equipped with sensors that record a variety of process values, including pressure (PS1-6), motor power (EPS1), volume flow (FS1-2), temperature (TS1-5), vibration (VS1), cooling efficiency (CE), cooling power (CP), and efficiency factor (SE) as demonstrated in Fig. 2 [15]. These sensors operate independently, meaning their readings remain unaffected by the system's state. Instead, they serve the crucial function of monitoring the system's condition.

TABLE II
Sensor Data (Independent Variables)

Physical Dimension	Sensor	Measuring Value	Units	SF
Pressure	PS1-PS6	Pressure	bar	100 Hz
Motor Power	ESP1	Motor Power	W	100 Hz
Flow Rate	FS1-FS2	Volume Flow	1/min	10 Hz
Temperature	TS1-TS5	Temperature	°C	1 Hz
Vibration	VS1	Vibration	mm/s	1 Hz
Cooling Efficiency	CE	Cooling Efficiency(virtual)	%	1 Hz
Cooling Power	CP	Cooling Power (virtual)	kW	1 Hz
System Efficiency	SE	Efficiency Factor	%	1 Hz

The control parameters in this hydraulic system are deemed dependent variables, as their conditions are directly influenced by the independent variables within the system. These dependent variables include the cooler, valve, pump, and hydraulic accumulator, which are crucial to the system’s operation and are detailed in Table III. The cooler regulates the temperature, the valve controls the fluid flow, the pump maintains pressure, and the accumulator compensates for pressure fluctuations.

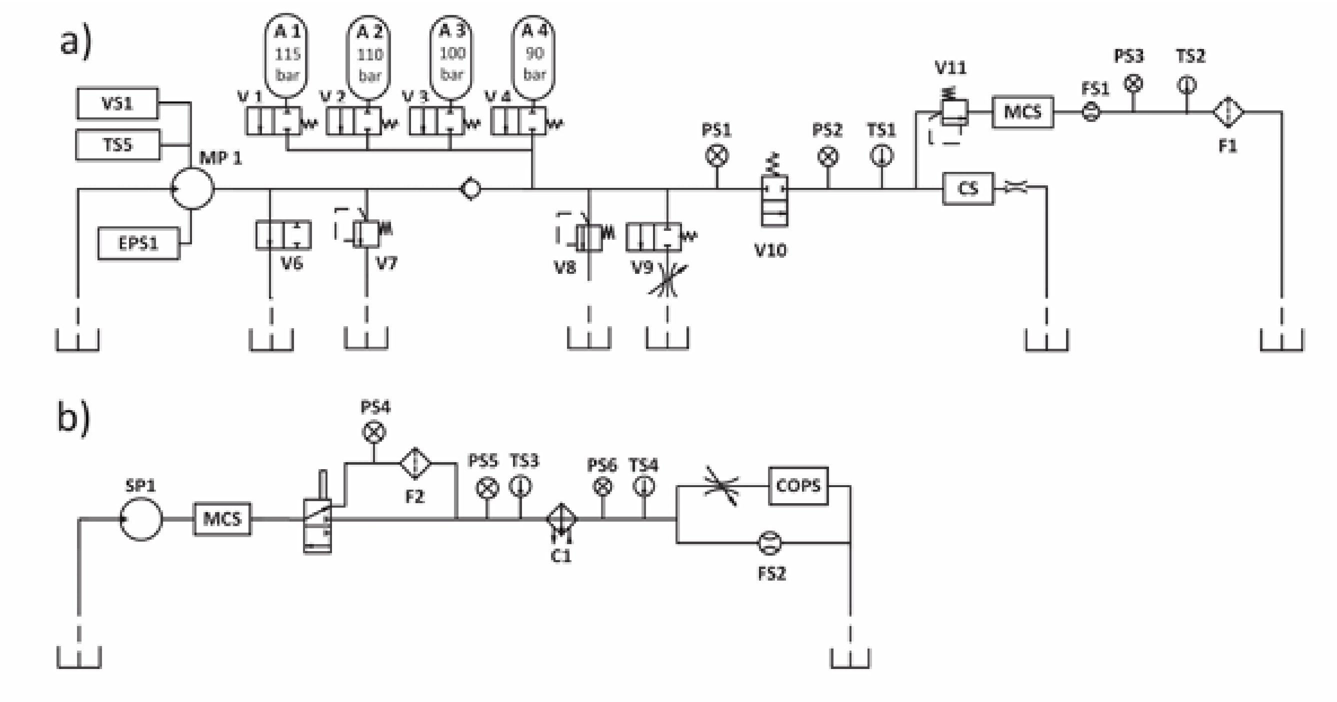


Fig. 2 Circuit Diagram of a Hydraulic Test Rig [15]

TABLE III
CONTROL PARAMETERS (DEPENDENT VARIABLES)

Component	**Condition**
Cooler	Cooling power decrease
Valve	Switching degradation
Pump	Internal leakage
Accumulator	Gas leakage
Stable Flag	Condition stability

B. Data Pre-Processing

Data preprocessing is essential since it aligns the collected data and lays a base for any data-driven technique, allowing an accurate and detailed analysis [12]. For this research paper, the data preprocessing implement was first extracting data from text files into arrays. Calculating the mean for all sensors per cycle. Setting the data frame for the processed data includes data from sensors and condition components. Lastly, explore the data to identify any missing or null values. Data exploration is visualising data by identifying patterns and outliers, data distribution, and its importance. During the exploration, it was found that no data points were missing or outliers. Since the dataset does not include any outliers or missing data, this helps improve the predictions' accuracy.

C. Modelling

Upon completing data pre-processing, training was assigned 80% of the data, while 20% was reserved for testing. This allocation is consistent with research suggesting that the best results are obtained using 20-30% of all available test data sets and 70-80% on training [28].

Predictive maintenance data modelling contains the prediction model development through ML and DL algorithms- RF, Catboost, and Long Short-Term Memory (LSTM). Complex pattern recognition and relationship identification effectiveness guide the choice of these algorithms. This process entails choosing appropriate algorithms as well as training them using information that will help bring out patterns in this regard.

RF is a versatile approach that outperforms regression- and classification-related tasks. It is created by combining several decision trees built on random samples and features. It is fast-fitting, low parameter-sensitive, has built-in error estimation, and is effective even when dimensions are extremely high [29]. Catboost, a well-known algorithm for gradient-boosting trees, utilises a random permutation process and unbiased boosting to mitigate information loss and variance. It considers any possible combination involving categorial to facilitate its performance and generalisation [30]. LSTM architecture contains memory blocks equipped with self-connected memory cells and three gates for ruling the information flow. It is suitable for long-term dependencies clustering in sequential data, including forecasting time series, natural language processing and speech recognition [31].

D. Evaluation

Using Precision, Recall, F1-Score, and Accuracy metrics will be implemented to evaluate the performance of ML and DL models. Table IV shows the formulas of each metric.

TABLE IV
EVALUATION METRICS

Metrics	**Formula**
Accuracy	$\frac{TP+TN}{TP+FP+TN+FN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F1-Score	$\frac{2 * Precision * Recall}{Precision + Recall}$

Evaluation metrics are among the most critical tools for estimating model effectiveness and providing thrilling insights into a number of aspects of the functionality of making predictions by the model. Precision can also be referred to as positive predictive value. It provides a measure of correctness for those positive predictions made by a model in measuring the ratio of accurate positive outcomes to all positive forecasts. Recall, sometimes called sensitivity, is a measure of the model's capability to capture all relevant instances from a dataset. It finds the ratio of correct positive predictions divided by the number of actual positive predictions. The F1-Score encapsulates both precision and recall in one. The score is calculated as the harmonic mean of both, giving a good indicator across classes in an imbalanced data set. Finally, accuracy evaluates the overall correctness of the model based on the number of proper outcomes, including both true positives and true negatives, out of the whole number of cases considered.

IV. RESULTS

A comprehensive prediction of the maintenance of hydraulic systems is carried out using ZeMA gGmbH data [27]. After the pre-processing for accuracy and alignment. ML and DL models are applied in predicting the failures in hydraulic systems., including Random Forest, Catboost, and Long Short-Term Memory. In this case, performance-based assessment metrics would be presented together with the model performances, such as Precision, Recall, F1-Score, and Accuracy. The results obtained for each of the five critical components of hydraulic systems, cooler, valve, pump, accumulator, and stable flag, are clearly compared by the algorithms used as shown in Tables V to VII.

A. *Random Forest*

The RF model predicts the component ‘Cooler’ with extraordinary accuracy on all metrics as shown in Table V. The valve component had a consistent performance of 95.92% except for the F1-Score, which is slightly lower at 95.90%. The pump component, however, displayed sorted scores which are near perfect: 99.55% for accuracy and recall, 99.56% for precision, and F1 score. The Accumulator component’s and Stable Flag performance was also high, ranging from 97.01% to 97.60%. RF model was excellent at predicting several conditions, especially Pump and Cooler. Those high marks under all these standards signify that this model can be a useful resource when it comes to predictive maintenance. The slightly lower scores for the Valve, Accumulator and Stable Flag components, while still high, suggest areas where the model’s performance could potentially be improved.

TABLE V
RF COMPARISON OF EVALUATION METRICS

	Accuracy	**Precision**	**Recall**	**F1-Score**
Cooler	100%	100%	100%	100%
Valve	95.92%	95.92%	95.92%	95.90%
Pump	99.55%	99.56%	99.55%	99.55%
Accumulator	97.51%	97.60%	97.51%	97.52%
Stable Flag	97.01%	97.13%	97.05%	97.02%

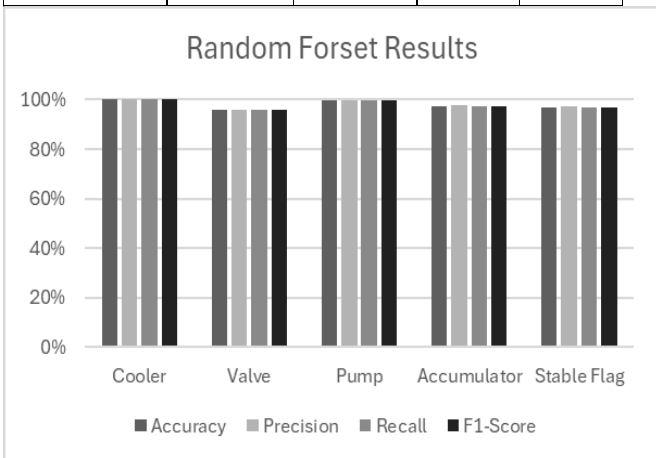


Fig. 3 Random Forest Evaluation Metrics Results

B. *Categorical Boost*

The Catboost model predicts with extraordinary accuracy for component ‘Cooler’, reaching 100% predictor rank in all dimensions as displayed in Table VI. The Valve achieved scores between 89.02% and 89.12%, suggesting a good estimation rate, though not as perfect. The scores for the Pump component ranged from 98.87% to 98.92%, showing prediction accuracy that is close to perfect. High prediction accuracy is indicated as the Accumulator and Stable Flag component scores range from 95.01% to 95.63%. These findings prove how good our model is at predicting the status of different hydraulic system components: it can do so with assurance due to results obtained through its use, indicating the robustness and reliability of the Catboost model. The model may assist in the predictive maintenance and monitoring of hydraulic systems by preventing potential system failures and enhancing operational efficiency. There are corresponding areas where the valve component could be used to enhance the overall model's performance. Therefore, further modifications will target and meet this specific design need.

TABLE VI
CATBOOST COMPARISON OF EVALUATION METRICS

	Accuracy	Precision	Recall	F1-Score
Cooler	100%	100%	100%	100%
Valve	89.12%	89.02%	89.12%	88.77%
Pump	98.87%	98.92%	98.87%	98.87%
Accumulator	95.01%	95.06%	95.01%	95.02%
Stable Flag	95.69%	95.83%	95.69%	95.63%

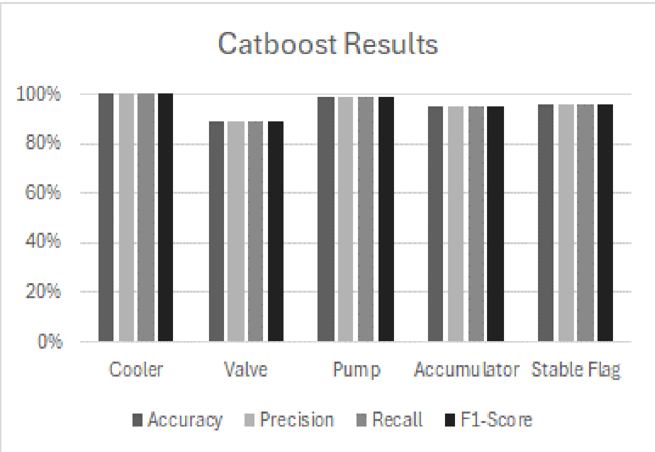


Fig. 4 Catboost Evaluation Metrics Results

C. *Long Short-Term Memory*

The Cooler, Valve and Accumulator components turned to be the best predicted by an LSTM model with this observation pointing out that there exists an outstanding forecasting ability for such components as seen in Table VII. In terms of performance, however, it is evident that the Cooler, Valve and Accumulator yield better scores as compared to the Pump, with 73.92% for accuracy rate, 82.00% for Precision rate and F1 Scores suggesting potential improvements. The lowest performing component is the Stable Flag component with an accuracy rate of about 66% for recall and 67% precision. The corresponding 53% F1-Score, while the related 44% precision. From these findings, it can be deduced that the LSTM model has some strongholds that should be strengthened to increase its ability to predict outcomes of the hydraulic system more accurately. Future research can be focused on optimizing LSTM architecture, for instance, by introducing more methods of feature scaling and selection in order to enhance the prediction of these constituents.

TABLE VII
LSTM COMPARISON OF EVALUATION METRICS

	Accuracy	Precision	Recall	F1-Score
Cooler	100%	100%	100%	100%
Valve	100%	100%	100%	100%
Pump	73.92%	82.23%	73.92%	71.81%
Accumulator	100%	100%	100%	100%
Stable Flag	66.67%	44.44%	66.67%	53.33%

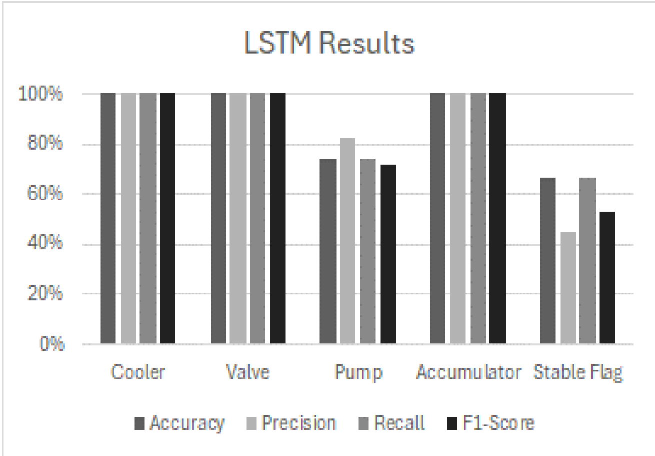


Fig. 5 LSTM Evaluation Metrics Results

Figure 6 illustrates the three ML algorithms—RF, Catboost, and LSTM—across five critical components of hydraulic systems: Cooler, Valve, Pump, Hydraulic Accumulator, and Stable Flag. Notably, the Cooler predictions achieved perfect accuracy (100%) across all models. LSTM outperformed the other algorithms for Valve maintenance with 100% accuracy, while Pump predictions favoured the RF model (99.55%). RF and Catboost performed well for Accumulator maintenance (97.51% and 95.01%, respectively). However, in the case of Stable Flag predictions, reliability was observed with RF and Catboost (97.05% and 95.69%), while LSTM lagged (66.67%).

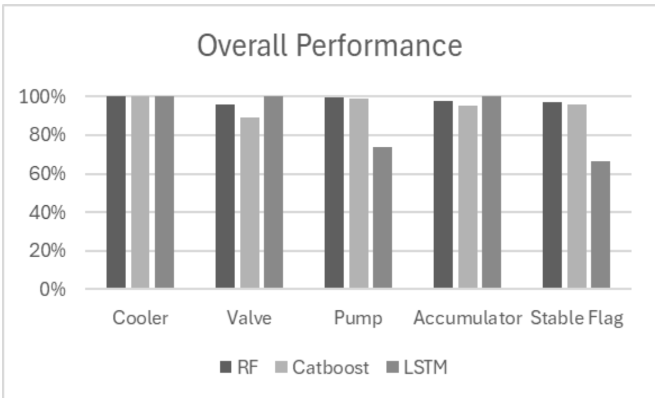


Fig. 6 Overall Accuracy of Classifier Models

RF performed well for the cooler, but the valve, accumulator, and stable flag had lower accuracy. It should be addressed through better hyperparameter tuning and feature selection. Catboost showed high accuracy in the cooler component but lower performance on the valve, indicating the need for feature selection analysis. LSTM had excellent accuracy for valve, cooler, and accumulator prediction results; however, lower accuracy for the pump and stable flag components suggests a need for more hyperparameter tuning or incorporating additional context into the model.

V. Conclusions

In conclusion, the aim was to explore the potential of various types of classifier models in predicting hydraulic equipment failure. This was done using ML and DL models. In particular, RF, Catboost and LSTM algorithms were used in the research. The results show that valve predictions are made by LSTM with excellent performance levels whereas pumps have the best outcome in terms of RF. Besides, accumulators, as well as stable flags, could be predicted accurately by both RF and Catboost. The recommendation for future research is to continue implementing feature scaling, selection, and extraction methods to improve the accuracy of fault prediction models for hydraulic systems.

References

[1] F. Raza, AI for Predictive Maintenance in Industrial Systems, 10.13140/RG.2.2.27313.35688 . 2023.

[2] Quatrini, E., Costantino, F., Pocci, C., & Tronci, M. (2020). Predictive model for the degradation state of a hydraulic system with dimensionality reduction. Procedia Manufacturing, 42, 516–523. https://doi.org/10.1016/j.promfg.2020.02.039

[3] R. Beebe, Predictive Maintenance of Pumps Using Condition Monitoring,https://openlibrary.org/books/OL3301501M/Predictive_maintenance_of_pumps_using_condition_monitoring , 2004

[4] R. K. Mobley, An Introduction to Predictive Maintenance, Elsevier, . https://doi.org/10.1016/b978-0-7506-7531-4.x5000-32002.

[5] Yugapriya, M., Judeson, A. K. J., & Jayanthy, S. (2022). Predictive Maintenance of Hydraulic Systems using Machine Learning Algorithms. 2022 International Conference on Electronics and Renewable Systems (ICEARS). https://doi.org/10.1109/icears53579.2022.9751840

[6] Wang, J., Zhang, T., & Wang, W. (2020). Health classification of hydraulic system based on k-means. 2020 International Conference on Urban Engineering and Management Science (ICUEMS). https://doi.org/10.1109/icuems50872.2020.00121

[7] Chen, C., Chan, Y., & Yu, S. (2023). Hydraulic System Failure Prediction Method with Limited Failure Data. 2023 Sixth International Symposium on Computer, Consumer and Control (IS3C). https://doi.org/10.1109/is3c57901.2023.00053

[8] Peng, Z., Zhang, K., & Chai, Y. (2020). Multiple fault diagnosis for hydraulic systems using Nearest-centroid-with-DBA and Random-Forest-based-time-series-classification. Proceedings of the 39th Chinese Control Conference. https://doi.org/10.23919/ccc50068.2020.9189401

[9] Askari, B., Cavone, G., Carli, R., Grall, A., & Dotoli, M. (2023). A Semi-Supervised Learning Approach for Fault Detection and Diagnosis in Complex Mechanical Systems. 023 IEEE 19th International Conference on Automation Science and Engineering (CASE). https://doi.org/10.1109/case56687.2023.10260469

[10] Buabeng, A., Simons, A. J., Frempong, N. K., & Ziggah, Y. Y. (2023). Hybrid intelligent predictive maintenance model for multiclass fault classification. Soft Computing. https://doi.org/10.1007/s00500-023-08993-1

[11] Mallak, A., & Fathi, M. (2021). Sensor and component fault detection and diagnosis for hydraulic machinery integrating LSTM autoencoder detector and diagnostic classifiers. Sensors, 21(2), 433. https://doi.org/10.3390/s21020433

[12] Chekkala, V. L. (2020). Predictive Maintenance for Fault Diagnosis and Failure Prognosis in Hydraulic System. National College of Ireland. https://norma.ncirl.ie/4389/1/vijitlaxmanchekkala.pdf

[13] Guo, P., Wu, J., Xu, X., Cheng, Y., & Wang, Y. (2019). Health condition monitoring of hydraulic system based on ensemble support vector machine. 2019 Prognostics & System Health Management Conference—Qingdao (PHM-2019 Qingdao). https://doi.org/10.1109/phm-qingdao46334.2019.8942981

[14] Gaurkar, S., Kotalwar, A., & Gabale, S. (2020). Predictive Maintenance of Industrial Machines using Machine Learning. International Research Journal of Engineering and Technology (IRJET), 8(11). https://www.academia.edu/63429811/IRJET_Predictive_Maintenance_of_Industrial_Machines_using_Machine_Learning

[15] Helwig, N., Pignanelli, E., & Schütze, A. (2015). Condition monitoring of a complex hydraulic system using multivariate statistics. 2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings. https://doi.org/10.1109/i2mtc.2015.7151267

[16] Berghout, T., Benbouzid, M., Muyeen, S. M., Bentrcia, T., & Mouss, L. H. (2021). Auto-NAHL: a neural network approach for Condition-Based maintenance of complex industrial systems. IEEE Access, 9, 152829–152840. https://doi.org/10.1109/access.2021.3127084

[17] Soh, J., & Kim, D. (2021). Condition Monitoring with Time Series Data Based on Probabilistic Model. 2021 24th International Conference on Electrical Machines and Systems (ICEMS). https://doi.org/10.23919/icems52562.2021.9634481

[18] Wu, J., Guo, P., Cheng, Y., Zhu, H., Wang, X., & Shao, X. (2020). Ensemble Generalized Multiclass Support-Vector-Machine-Based health evaluation of complex degradation systems. IEEE/ASME Transactions on Mechatronics, 25(5), 2230–2240. https://doi.org/10.1109/tmech.2020.3009449

[19] Xu, Z., Yu, H., Gui, Y., & Liu, C. (2021). Compound fault diagnosis in hydraulic system with multi-output SVM. CSAA/IET International Conference on Aircraft Utility Systems (AUS 2020). https://doi.org/10.1049/icp.2021.0470

[20] Liu, J., Xu, H., Peng, X., Wang, J., & He, C. (2023). Reliable composite fault diagnosis of hydraulic systems based on linear discriminant analysis and multi-output hybrid kernel extreme learning machine. Reliability Engineering & Systems Safety, 234, 109178. https://doi.org/10.1016/j.ress.2023.109178

[21] Kim, D., & Heo, T. (2022). Anomaly Detection with Feature Extraction Based on Machine Learning Using Hydraulic System IoT Sensor Data. Sensors, 22(7), 2479. https://doi.org/10.3390/s22072479

[22] Ma, X., Wang, P., Zhang, B., & Sun, M. (2021). A multi-rate sensor information fusion strategy for multitasking fault diagnosis based on convolutional neural network. Journal of Sensors, 2021, 1–17. https://doi.org/10.1155/2021/9952450

[23] Keleko, A. T., Kamsu-Foguem, B., Ngouna, R. H., & Tongne, A. (2023). Health condition monitoring of a complex hydraulic system using Deep Neural Network and DeepSHAP explainable XAI. Advances in Engineering Software, 175, 103339. https://doi.org/10.1016/j.advengsoft.2022.103339

[24] Zhang, P., Hu, W., Cao, W., Chen, L., & Wu, M. (2022). Multi-Fault diagnosis of hydraulic systems based on fully convolutional networks. 2022 13th Asian Control Conference (ASCC). https://doi.org/10.23919/ascc56756.2022.9828182

[25] Kim, K., & Jeong, J. (2022). Multi-layer Stacking Ensemble for Fault Detection Classification in Hydraulic System. 2022 26th International Conference on Circuits, Systems, Communications and Computers (CSCC). https://doi.org/10.1109/cscc55931.2022.00066

[26] Khan, H. A., Bhatti, U. A., Kamal, K., Alkahtani, M., Abidi, M. H., & Mathavan, S. (2023). Fault classification for cooling system of hydraulic machinery using AI. Sensors, 23(16), 7152. https://doi.org/10.3390/s23167152

[27] Schneider, T., Klein, S., & Bastuck, M., "Condition monitoring of hydraulic systems Data Set at ZeMA [Data set]," Zenodo, 2018, https://doi.org/10.5281/zenodo.1323611.

[28] Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation. Departmental Technical Reports (CS). https://digitalcommons.utep.edu/cgi/viewcontent.cgi?article=2202&context=cs_techrep

[29] C. Zhang and Y. Ma, Ensemble Machine Learning: Methods and Applications, 1st ed., Springer Science+Business Media, LLC, Ed.

New York, NY: Springer, 2012. DOI: https://doi.org/10.1007/978-1-4419-9326-7
[30] CRC Press, "Artificial Intelligence in Highway Safety," 1st ed., DOI: 10.1201/9781003005599, ISBN: 978-0-367-43670-4, Sep. 2022.
[31] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, 1st ed., Studies in Computational Intelligence, vol. 385, Springer-Verlag GmbH Berlin Heidelberg, 2012. DOI: https://doi.org/10.1007/978-3-642-24797-2

Dear Dr. Naveen Palanichamy:

Congratulations - your paper #1571030735 ('Evaluating Machine Learning and Deep Learning Algorithms for Predictive Maintenance of Hydraulic Systems') for CITIC 2024 has been **accepted** for presentation in **Track 1 - Artificial Intelligence & Machine Learning**.

The reviewer's comments are given at the bottom of this email or can be found at 1571030735. The similarity score of your paper is **4**. In case the score is above 15, please revise your paper to reduce the similarity score to below 15.

Please revise your manuscript based on reviewers' feedback and upload the camera-ready paper (i.e., Final Manuscript) with complete details of all the authors and affiliations to EDAS before **15 June 2024**. Nevertheless, you are encouraged to upload your camera-ready paper to EDAS earliest possible so that we will be able to forward it to respective journals for future processing.

In addition, here is the link for the registration form: https://forms.office.com/r/uYBQciU9Gm

1. Kindly complete this registration form for each paper to be presented (1 registration per paper).
2. The conference committee will assess the papers for quality, taking into consideration the reviewers' reports and the scope of the work, inorder to determine their suitability for recommendation to collaborating journals or proceedings.
3. The acceptance of papers by collaborating journals is contingent upon meeting the individual requirements of each journal. a) The final decision rests with the respective journals. b) Papers accepted for publication by Cogent Social Science (Q2-Scopus) are subject to an additional fee of USD 1000 for the Article Processing Charge (APC). c) Authors will receive notification regarding the final proposed publication arrangement and have the option to either accept or decline the proposal.

4. However, in the event that an author declines the proposed arrangement, the fees paid are non-refundable.

Thank you and we look forward to seeing you in CITIC 2024.

Regards, The conference chair

TPT3101 Final Year Project (FYP1) Meeting Log
Trimester 1, 2023/24 (Trimester ID:2310)

Meeting Date: 4th April 2024	**Meeting No.: 1**
Meeting Mode: Face to Face	
Project ID: 2718	**Project Type: Research-based**
Project Title : Predictive Equipment Maintenance Of Hydraulics System	
Student ID : 1191202335	**Student Name: Ayat Abdulaziz Gaber Al-Khulaqi**
Student Programme and Specialisation: Data Science	
Supervisor Name: Dr. Naveen	**Co-Supervisor Name: (if applicable)**

1. WORK DONE
[Please write the details of the work done after the last meeting]

Tasks: Journal Paper
I have started writing the abstract, introduction, related works, and methodology sections.
- I looked for more research papers and books to provide accurate information.
- Create comparative tables to compare previous research works.
- Created a flow chart for the methodology to explain how the research is implemented.

2. WORK TO BE DONE
[Please write the details of the work to be done before the next meeting]

Tasks: Journal Paper
- I have to adjust the paper to follow the citic template.
- I have to start coding, testing, and tuning hyper-tuning code.
- After hyper-tuning, I need to complete the experimental setup and results sections.

Tasks: FYP 2
- Start implementing the code.
- To start learning how to create a dashboard in Power BI.

3. PROBLEMS ENCOUNTERED AND SOLUTIONS
[Please write the details of the problems encountered after the last meeting and provide the solutions/plan for the solutions]

4. COMMENTS (Supervisor / Co-Supervisor / Company Supervisor

Dr. NAVEEN PALANICHAMY
BE(Distn.), ME, MBA, PhD
Lecturer
Faculty of Computing and Informatics
Multimedia University
Persiaran Multimedia, 63100 Cyberjaya
Selangor Darul Ehsan

…………………………… ……………………………..

Supervisor's Signature Student's Signature

IMPORTANT NOTES TO STUDENTS:

1. Items 1 – 3 are to be completed by the students prior to the meeting. Item 4 is to be completed by the supervisor / co-supervisor / company supervisor.

2. Student has to upload the soft copies of the meeting logs in Google Classroom and also attach them along with interim (FYP1) report.
   Minimum requirement is SIX Meeting Logs (Period: Week 3 to Week 12). Students can
   have fortnightly meetings with the supervisor.

3. Log sheets provide the basis for evaluating the General Effort (Project Management, Attitude, and Technical Competency) of the student, by the supervisor and also for checking the attendance requirement of the student, by the FYP Committee.

   This also provide the student with feedback from the supervisor / co-supervisor / company supervisor on the tasks done and provide the plan for the upcoming tasks. This can provide the motivation for the student to give consistent and efficient effort throughout the period of FYP.

4. Student who fails to meet the minimum requirement (six nos.) of log sheets will not be allowed to submit FYP report.

FACULTY OF COMPUTING & INFORMATICS

MULTIMEDIA UNIVERSITY

TPT3101 Final Year Project (FYP1) Meeting Log
Trimester 1, 2023/24 (Trimester ID:2310)

Meeting Date: 25th April 2024	**Meeting No.: 2**
Meeting Mode: Face to Face	
Project ID: 2718	**Project Type: Research-based**
Project Title : Predictive Equipment Maintenance Of Hydraulics System	
Student ID : 1191202335	**Student Name: Ayat Abdulaziz Gaber Al-Khulaqi**
Student Programme and Specialisation: Data Science	
Supervisor Name: Dr. Naveen	**Co-Supervisor Name: (if applicable)**

1. WORK DONE

[Please write the details of the work done after the last meeting]

Tasks: Journal Paper

- I have adjusted the paper to follow the citic template.
- I have completed coding, testing, and tuning hyper-tuning code.
- I have completed the experimental setup and results sections.

Tasks: FYP 2

- I have started implementing and testing the code.
- I have started learning how to create a dashboard in Power BI.

2. WORK TO BE DONE

[Please write the details of the work to be done before the next meeting]

Tasks: Journal Paper

- I have to mention the results of the research in the abstract.
- I have to update the introduction, and in the related works section to update the table to fit the citic format.
- I have to update the flow chart and include the data souring in the chart.
-

Tasks: FYP 2

- To continue with implementing and testing the code.
- To start creating draft examples of dashboards.

3. PROBLEMS ENCOUNTERED AND SOLUTIONS

[Please write the details of the problems encountered after the last meeting and provide the solutions/plan for the solutions]

4. COMMENTS (Supervisor / Co-Supervisor / Company Supervisor

Dr. NAVEEN PALANICHAMY
BE(Distn.), ME, MBA, PhD
Lecturer
Faculty of Computing and Informatics
Multimedia University
Persiaran Multimedia, 63100 Cyberjaya
Selangor Darul Ehsan

……………………………	…………………………...
Supervisor’s Signature	Student’s Signature

IMPORTANT NOTES TO STUDENTS:

1. Items 1 – 3 are to be completed by the students prior to the meeting. Item 4 is to be completed by the supervisor / co-supervisor / company supervisor.

2. Student has to upload the soft copies of the meeting logs in Google Classroom and also attach them along with interim (FYP1) report.
   Minimum requirement is SIX Meeting Logs (Period: Week 3 to Week 12). Students can
   have fortnightly meetings with the supervisor.

3. Log sheets provide the basis for evaluating the General Effort (Project Management, Attitude, and Technical Competency) of the student, by the supervisor and also for checking the attendance requirement of the student, by the FYP Committee.

   This also provide the student with feedback from the supervisor / co-supervisor / company supervisor on the tasks done and provide the plan for the upcoming tasks. This can provide the motivation for the student to give consistent and efficient effort throughout the period of FYP.

4. Student who fails to meet the minimum requirement (six nos.) of log sheets will not be allowed to submit FYP report.

FACULTY OF COMPUTING & INFORMATICS

MULTIMEDIA UNIVERSITY

TPT3101 Final Year Project (FYP1) Meeting Log
Trimester 1, 2023/24 (Trimester ID:2310)

Meeting Date: 2nd May 2024	**Meeting No.: 3**
Meeting Mode: Face to Face	
Project ID: 2718	**Project Type: Research-based**
Project Title : Predictive Equipment Maintenance Of Hydraulics System	
Student ID : 1191202335	**Student Name: Ayat Abdulaziz Gaber Al-Khulaqi**
Student Programme and Specialisation: Data Science	
Supervisor Name: Dr. Naveen	**Co-Supervisor Name: (if applicable)**

1. WORK DONE

[Please write the details of the work done after the last meeting]

Tasks: Journal Paper

- The results of the research were mentioned in the abstract.
- The introduction was updated, and the table in the related works section was updated to fit the citic format.
- The flow chart was updated, and the data sourcing was included in the chart.

Tasks: FYP 2

- Continued with implementing and testing the code.
- Started creating draft examples of dashboards.

2. WORK TO BE DONE

[Please write the details of the work to be done before the next meeting]

Tasks: Journal Paper

- Check MMU Press for more journal papers.
- Abstract: mention which of the machine or deep learning models performed the best.
- Introduction: Mention the study gap and include a paragraph explaining about the sub-sections.
- Related Work:
  - Formalise the first paragraph.
  - Combine all three tables into one for better comparison.
  - Apply the correct number formatting for number paper referencing in the tables.
- Methodology:
  - Include a backup reference for data splitting.
  - Resize the methodology flow chart.

Tasks: FYP 2

- To continue with implementing and testing the code.

3. PROBLEMS ENCOUNTERED AND SOLUTIONS

[Please write the details of the problems encountered after the last meeting and provide the solutions/plan for the solutions]

4. COMMENTS (Supervisor / Co-Supervisor / Company Supervisor

Dr. NAVEEN PALANICHAMY
BE(Distn.), ME, MBA, PhD
Lecturer
Faculty of Computing and Informatics
Multimedia University
Persiaran Multimedia, 63100 Cyberjaya
Selangor Darul Ehsan

……………………………
Supervisor's Signature

…………………………...
Student's Signature

IMPORTANT NOTES TO STUDENTS:

1. Items 1 – 3 are to be completed by the students prior to the meeting. Item 4 is to be completed by the supervisor / co-supervisor / company supervisor.

2. Student has to upload the soft copies of the meeting logs in Google Classroom and also attach them along with interim (FYP1) report.
   Minimum requirement is SIX Meeting Logs (Period: Week 3 to Week 12). Students can have fortnightly meetings with the supervisor.

3. Log sheets provide the basis for evaluating the General Effort (Project Management, Attitude, and Technical Competency) of the student, by the supervisor and also for checking the attendance requirement of the student, by the FYP Committee.

   This also provide the student with feedback from the supervisor / co-supervisor / company supervisor on the tasks done and provide the plan for the upcoming tasks. This can provide the motivation for the student to give consistent and efficient effort throughout the period of FYP.

4. Student who fails to meet the minimum requirement (six nos.) of log sheets will not be allowed to submit FYP report.

FACULTY OF COMPUTING & INFORMATICS

TPT3101 Final Year Project (FYP1) Meeting Log
Trimester 1, 2023/24 (Trimester ID:2310)

Meeting Date: 8th May 2024	**Meeting No.: 4**
Meeting Mode: Face to Face	
Project ID: 2718	**Project Type: Research-based**
Project Title : Predictive Equipment Maintenance Of Hydraulics System	
Student ID : 1191202335	**Student Name: Ayat Abdulaziz Gaber Al-Khulaqi**
Student Programme and Specialisation: Data Science	
Supervisor Name: Dr. Naveen	**Co-Supervisor Name:** **(if applicable)**

1. WORK DONE

[Please write the details of the work done after the last meeting]

Tasks: Journal Paper

- I checked MMU Press for more journal papers.
- In the abstract, I mentioned which of the machine or deep learning models performed the best.
- In the introduction, I mentioned the study gap and included a paragraph explaining the sub-sections.
- **In the Related Work section:**
    - I formalized the first paragraph.
    - I combined all three tables into one for better comparison.
    - I applied the correct number formatting for number paper referencing in the tables.
- **In the Methodology section:**
    - I included a backup reference for data splitting.
    - I resized the methodology flow chart.

Tasks: FYP 2

- I continued with implementing and testing the code.

2. WORK TO BE DONE

[Please write the details of the work to be done before the next meeting]

Tasks: Journal Paper

- Condition Components and Fault Analysis
    - Write a short paragraph explaining the importance of the condition components.
    - Link the conditions with the fault analysis.
- Flow Structure and Terminology
    - Update the document to a more flowy structure.
    - Use short terms, e.g., CM, PdM.
- Hydraulic System Section
    - Add a sentence stating that the section discusses the hydraulic system.

3. PROBLEMS ENCOUNTERED AND SOLUTIONS

[Please write the details of the problems encountered after the last meeting and provide the solutions/plan for the solutions]

4. COMMENTS (Supervisor / Co-Supervisor / Company Supervisor

Dr. NAVEEN PALANICHAMY
BE(Distn.), ME, MBA, PhD
Lecturer
Faculty of Computing and Informatics
Multimedia University
Persiaran Multimedia, 63100 Cyberjaya
Selangor Darul Ehsan

…………………………… ……………………………...

Supervisor's Signature Student's Signature

IMPORTANT NOTES TO STUDENTS:

1. Items 1 – 3 are to be completed by the students prior to the meeting. Item 4 is to be completed by the supervisor / co-supervisor / company supervisor.

2. Student has to upload the soft copies of the meeting logs in Google Classroom and also attach them along with interim (FYP1) report.
   Minimum requirement is SIX Meeting Logs (Period: Week 3 to Week 12). Students can
   have fortnightly meetings with the supervisor.

3. Log sheets provide the basis for evaluating the General Effort (Project Management, Attitude, and Technical Competency) of the student, by the supervisor and also for checking the attendance requirement of the student, by the FYP Committee.

   This also provide the student with feedback from the supervisor / co-supervisor / company supervisor on the tasks done and provide the plan for the upcoming tasks. This can provide the motivation for the student to give consistent and efficient effort throughout the period of FYP.

4. Student who fails to meet the minimum requirement (six nos.) of log sheets will not be allowed to submit FYP report.

MULTIMEDIA UNIVERSITY | FACULTY OF COMPUTING & INFORMATICS

TPT3101 Final Year Project (FYP1) Meeting Log
Trimester 1, 2023/24 (Trimester ID:2310)

Meeting Date: 25th May 2024	**Meeting No.: 5**
Meeting Mode: Online	
Project ID: 2718	**Project Type: Research-based**
Project Title : Predictive Equipment Maintenance Of Hydraulics System	
Student ID : 1191202335	**Student Name: Ayat Abdulaziz Gaber Al-Khulaqi**
Student Programme and Specialisation: Data Science	
Supervisor Name: Dr. Naveen	**Co-Supervisor Name: (if applicable)**

1. WORK DONE

[Please write the details of the work done after the last meeting]

Tasks: Journal Paper

- Condition Components and Fault Analysis
    - I wrote a short paragraph explaining the importance of the condition components.
    - I linked the conditions with the fault analysis.
- **Flow Structure and Terminology**
    - I updated the document to a more flowy structure.
    - I used short terms, e.g., CM, PdM.
- **Hydraulic System Section**
    - I added a sentence stating that the section discusses the hydraulic system.

s

2. WORK TO BE DONE

[Please write the details of the work to be done before the next meeting]

Tasks: Journal Paper

- ZeMA gGmbH Dataset Information
    - Provide more information about the ZeMA gGmbH dataset, including the size, types of features, and any preprocessing steps taken before training the models.
- Comparison with Existing Methods
    - Include a brief comparison with existing predictive maintenance approaches. Highlight how the proposed method improves upon or differs from these methods.
- Model Limitations and Challenges
    - Discuss any limitations or challenges encountered with each model, particularly in predicting certain components.
- Clarification of Reliability Statement
    - Clarify the statement "overall reliability observed for Accumulator and Stable Flag components" by providing more context on what "Stable Flag components" refers to.
- Rephrase "predict the maintenance of hydraulic system equipment" for clarity, e.g., "predict the maintenance needs of hydraulic system equipment."
- Specify which model is referred to as "the classifier model, which has proven to be the most efficient and successful."
- Rephrase "revealed their varying performance" for clarity, e.g., "revealed varying performance across different components."

3. PROBLEMS ENCOUNTERED AND SOLUTIONS

[Please write the details of the problems encountered after the last meeting and provide the solutions/plan for the solutions]

4. COMMENTS (Supervisor / Co-Supervisor / Company Supervisor

Dr. NAVEEN PALANICHAMY
BE(Distn.), ME, MBA, PhD
Lecturer
Faculty of Computing and Informatics
Multimedia University
Persiaran Multimedia, 63100 Cyberjaya
Selangor Darul Ehsan

…………………………… ……………………………...

Supervisor's Signature Student's Signature

IMPORTANT NOTES TO STUDENTS:

1. Items 1 – 3 are to be completed by the students prior to the meeting. Item 4 is to be completed by the supervisor / co-supervisor / company supervisor.

2. Student has to upload the soft copies of the meeting logs in Google Classroom and also attach them along with interim (FYP1) report.
   Minimum requirement is SIX Meeting Logs (Period: Week 3 to Week 12). Students can
   have fortnightly meetings with the supervisor.

3. Log sheets provide the basis for evaluating the General Effort (Project Management, Attitude, and Technical Competency) of the student, by the supervisor and also for checking the attendance requirement of the student, by the FYP Committee.

   This also provide the student with feedback from the supervisor / co-supervisor / company supervisor on the tasks done and provide the plan for the upcoming tasks. This can provide the motivation for the student to give consistent and efficient effort throughout the period of FYP.

4. Student who fails to meet the minimum requirement (six nos.) of log sheets will not be allowed to submit FYP report.

TPT3101 Final Year Project (FYP1) Meeting Log
Trimester 1, 2023/24 (Trimester ID:2310)

Meeting Date: 15th June 2024	**Meeting No.: 6**
Meeting Mode: Online	
Project ID: 2718	**Project Type: Research-based**
Project Title : Predictive Equipment Maintenance Of Hydraulics System	
Student ID : 1191202335	**Student Name: Ayat Abdulaziz Gaber Al-Khulaqi**
Student Programme and Specialisation: Data Science	
Supervisor Name: Dr. Naveen	**Co-Supervisor Name: (if applicable)**

1. WORK DONE

[Please write the details of the work done after the last meeting]

Tasks: Journal Paper

- ZeMA gGmbH Dataset Information
  - I provided more information about the ZeMA gGmbH dataset, including the size, types of features, and any preprocessing steps taken before training the models.
- Comparison with Existing Methods
  - I included a brief comparison with existing predictive maintenance approaches. I highlighted how the proposed method improved upon or differed from these methods.
- Model Limitations and Challenges
  - I discussed any limitations or challenges encountered with each model, particularly in predicting certain components.
- Clarification of Reliability Statement
  - I clarified the statement "overall reliability observed for Accumulator and Stable Flag components" by providing more context on what "Stable Flag components" refers to.
- I rephrased "predict the maintenance of hydraulic system equipment" for clarity, e.g., "predict the maintenance needs of hydraulic system equipment."
- I specified which model was called "the classifier model, which has proven to be the most efficient and successful."
- I rephrased "revealed their varying performance" for clarity, e.g., "revealed varying performance across different components."

2. WORK TO BE DONE

Tasks: Journal Paper

- To accurately reformat the references according to the required correct format, ensuring that all citations and bibliographical entries adhere to the specified guidelines.
- To comprehensively revise the report to decrease the percentage of AI-generated content.

Tasks: FYP 2

- To complete the dashboard using Power BI.
- To finalize the comprehensive report, make sure all sections are complete, and subsequently submit it for review to receive feedback and approval.

3. PROBLEMS ENCOUNTERED AND SOLUTIONS

[Please write the details of the problems encountered after the last meeting and provide the solutions/plan for the solutions]

4. COMMENTS (Supervisor / Co-Supervisor / Company Supervisor

Dr. NAVEEN PALANICHAMY
BE(Distn.), ME, MBA, PhD
Lecturer
Faculty of Computing and Informatics
Multimedia University
Persiaran Multimedia, 63100 Cyberjaya
Selangor Darul Ehsan

……………………………
Supervisor's Signature

…………………………...
Student's Signature

IMPORTANT NOTES TO STUDENTS:

1. Items 1 – 3 are to be completed by the students prior to the meeting. Item 4 is to be completed by the supervisor / co-supervisor / company supervisor.

2. Student has to upload the soft copies of the meeting logs in Google Classroom and also attach them along with interim (FYP1) report.
   Minimum requirement is SIX Meeting Logs (Period: Week 3 to Week 12). Students can
   have fortnightly meetings with the supervisor.

3. Log sheets provide the basis for evaluating the General Effort (Project Management, Attitude, and Technical Competency) of the student, by the supervisor and also for checking the attendance requirement of the student, by the FYP Committee.

   This also provide the student with feedback from the supervisor / co-supervisor / company supervisor on the tasks done and provide the plan for the upcoming tasks. This can provide the motivation for the student to give consistent and efficient effort throughout the period of FYP.

4. Student who fails to meet the minimum requirement (six nos.) of log sheets will not be allowed to submit FYP report.

FYP2 Report

ORIGINALITY REPORT

17%	14%	13%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

Coding Drive Link

Link