Time Series: Implementation of Classification and Regression Algorithm using Machine Intelligence Techniques

Semester – 7

Major Project – I (01CE0716)

A PROJECT REPORT

Submitted by

MASRUK HABIB

92100103165

PUVANENTHIRARAJAH

92100103168

MAY THAZIN

92100103067

BACHELOR OF TECHNOLOGY

in

Computer Engineering

Marwadi University, Rajkot

December, 2024

Major Project-I (01CE0716)

Marwadi University

Faculty of Technology

Department of Computer Engineering

2024-25

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **Time Series: Implementation of Classification and Regression Algorithm using Machine Intelligence Techniques** has been carried out by **Masruk Habib** (92100103165), **Puvanenthirarajah** (92100103168), **May Thazin** (92100103067) under my guidance in partial fulfilment for the degree of Bachelor of Technology in Computer Engineering, 7th Semester of Marwadi University, Rajkot during the academic year 2024-25.

Ravikumar R N	Dr. Krunal Vaghela
Asst. Professor	Associate Professor
Internal Guide	Head of the Department

Marwadi University

Rajkot

DECLARATION

We hereby declare that the **Major Project-I (01CE0716)** report submitted along with the Project entitled **Time Series: Implementation of Classification and Regression Algorithm using Machine Intelligence Techniques** submitted in partial fulfilment for the degree of Bachelor of Technology in Computer Engineering to Marwadi University, Rajkot, is a bonafide record of original project work carried out by us at Marwadi University under the supervision of **Prof. Ravikumar R N** and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

	Name of the Student	Sign of Student
1	Masruk Habib	_____
2	Puvanenthira Rajah	_____
3	May Thazin	_____

Acknowledgement

We would like to express our deepest gratitude to the Almighty for His divine blessings, which enabled us to successfully complete this major project.

We are profoundly thankful to Prof. Ravikumar R. N., Assistant Professor, Marwadi University, Rajkot, Gujarat, for his invaluable guidance and unwavering support throughout the course of this project. His expertise and enthusiasm in the field of Machine Learning have been a constant source of inspiration. His meticulous supervision, insightful feedback, and continuous encouragement played a pivotal role in the successful completion of this project.

We would also like to extend our sincere thanks to Dr. Kurnal Vagela, Head of the Department of Computer Engineering, for his support and assistance. We are equally grateful to the faculty members and staff of the Computer Engineering department at Marwadi University for their cooperation and assistance throughout this endeavour.

Furthermore, we express our appreciation to all individuals who contributed to this project, whether through constructive feedback, encouragement, or practical support. Your collective efforts have been crucial in bringing this project to fruition.

This project would not have been possible without the support, guidance, and cooperation of all involved, for which we are deeply thankful.

Abstract

The emergence of collaborative robots (cobot's) like Universal Robots UR3 has significantly contributed to Industry 4.0 by enabling safe human-robot collaboration and dynamic adaptability in manufacturing settings. The UR3, a compact and versatile robotic arm, performs tasks such as assembly, quality inspection, and material handling with ease, aided by an intuitive programming interface that suits industries requiring rapid production adjustments and mass customization. This study focuses on predictive maintenance for the UR3, leveraging time-series data analysis to develop an advanced fault detection system. By employing machine learning techniques—including regression, classification, and clustering—on a dataset from the UR3 CobotOps, the study explores methods for identifying anomalies that signal potential faults. Models such as Long Short-Term Memory (LSTM), ARIMA, XGBoost, and LightGBM are evaluated using metrics like accuracy, precision, recall, and F1 Score. Hybrid models, combining algorithms like Random Forest and LightGBM, demonstrate enhanced performance with high accuracy and balanced recall and precision scores, underscoring the advantages of integrated approaches. Results show that hybrid models achieve superior fault detection capabilities, thereby improving operational efficiency, safety, and cost-effectiveness in robotic systems. The findings contribute to knowledge in predictive maintenance, with implications for scalable, data-driven optimization in robotics and smart manufacturing environments.

Keywords— *Collaborative robots, UR3 cobot, predictive maintenance, fault detection, machine learning, time-series analysis, Industry 4.0, hybrid models, smart manufacturing, operational efficiency.*

List of Figures

Figure 3.1 Work Flow 17
Figure 3.2 Analysis Current and Temperature Values Over Time..... 23
Figure 3.3 Speed vs Current Scatter Plots 24
Figure 3.4 Temperature Histograms 26
Figure 3.5 Speed Values Over Time 27
Figure 3.6 Steps of ARIMA Model 31
Figure 3.7 Comparison of Original and Differenced Tool Current Time Series..... 32
Figure 3.8 Tool Current vs. Index (Last 500 Records) 33
Figure 3.9 Correlation Matrix 35
Figure 3.10 Time Series Decomposition of Tool Current 36
Figure 3.11 Loss Per Epoch 37
Figure 3.12 Steps of K-mean clusters 38
Figure 3.13 Elbow Method for Determining Optimal Number of Clusters 39
Figure 3.14 Silhouette Score for Determining Optimal Number of Clusters 40
Figure 4.1 Single Model Performance ……..... 43
Figure 4.2 Hybrid Model performance 44
Figure 4.3 Final outcome of Classification 45
Figure 4.4 Final outcome of ARIMA 47
Figure 4.5 The Architecture Summary of a Sequential NN Model with Two Layers 48
Figure 4.6 Final outcome of LSTM..... 49
Figure 4.7 Silhouette Plot for K-Means Clustering (k=2) 50
Figure 4.8 K-Means Clustering Visualization with 3 Clusters 51
Figure 4.9 3D Visualization of Clusters using PCA 52
Figure 4.10 3D Visualization of Clusters using PCA 1 53

List of Tables

Table 1.1 Project Management Table 7
Table 3.2.1 Feature Description 18
Table 4.1 Result Analysis on Classification 42
Table 4.2 ARIMA Model Performance Metrics..... 46

Abbreviations

ML	**Machine Learning**
Cobots	**Collaborative robots**
ET	**Extra Trees**
AB	**AdaBoost**
LOC	**line of code**
Rf	**Random Forest**
XGBoost	**Extreme Gradient Boosting**
LightGBM	**Light Gradient Boosting Machine**
TP	**True positives**
TN	**True negatives**
FP	**False positives**
FN	**False negatives**
ARIMA	**Auto Regressive Integrated Moving Average**
LSTM	**Long Short-Term Memory**
MSE	**Mean Squared Error**
RMSE	**Root Mean Squared Error**
R^2	**R-squared**

Table of Contents

Acknowledgement.....i

Abstract ii

List of Figures iii

List of Tables iv

List of Abbreviations v

Table of Contents vi

Chapter 1 INTRODUCTION 1

1.1 Introduction of UR3 Cobot 1

1.1.1 Use of UR3 Cobot 1

1.1.2 Study's Goals 2

1.1.3 Benefit 2

1.2 Motivation 4

1.3 Expected Outcomes 5

1.4 Project Management and Finance …..... 6

Chapter 2 LITERATURE REVIEW……..... 8

2.1 Related Works..... 8

Chapter 3 RESEARCH METHODOLOGY..... 17

3.1 Methodology..... 17

3.2 Data Analysis..... 18

3.3 Classification Techniques for the UR3 Cobot..... 20

3.3.0 Data preprocess..... 20

3.3.1 Model Training..... 20

3.3.2 Visualization..... 23

3.3.2.1 Analysis of Current and Temperature Values Over Time..... 23

3.3.2.2 Speed vs Current Scatter Plots..... 24

3.3.2.3 Temperature Histograms..... 25

3.3.2.4 Speed Values Over Time..... 27

3.3.3 Model Evalution..... 28

3.4 ARIMA Model for the UR3 Cobot 30

3.4.1 Steps of ARIMA Model 31
3.4.2 Visualization of ARIMA Model..... 32
3.4.2.1 Comparison of Original and Differenced Tool Current Time Series… 32
3.4.2.2 Tool Current vs. Index (Last 500 Records) 33
3.5 LSTM on UR3 Cobot..... 34
3.5.1 Visualization of LSTM 35
3.5.2 Correlation Matrix 35
3.5.3 Time Series Decomposition of Tool Current 36
3.5.4 Loss Per Epoch 37
3.6 K-mean clusters on UR3..... 38
3.6.1 Elbow Method for Determining Optimal Number of Cluster 39
3.6.2 Silhouette Score for Determining Optimal Number of Clusters 40
Chapter 4 RESULT AND DISCUSSION..... 41
4.1 Experimental Results & Analysis 41
4.2 Result Graph 43
4.2.1 Single Model Performance 43
4.2.2 Hybrid Model Performance..... 44
4.2.3 Final outcome of Classification..... 45
4.2.4 Final outcome of ARIMA 46
4.2.5 The Architecture Summary of a Sequential NN Model with Two Layers 48
4.2.5.1 Final outcome of LSTM 49
4.2.6 Silhouette Plot for K-Means Clustering (k=2) 50
4.2.7 K-Means Clustering Visualization with 3 Clusters..... 51
4.2.8 3D Visualization of Clusters using PCA 52
4.2.9 3D Visualization of Clusters using PCA 53
Chapter 5 CONCLUSION AND FUTURE DIRECTION..... 54
5.1 Summary of the Study..... 54
5.2 Conclusion 55
5.3 Implication for Future Study 55
References 56

CHAPTER 1
INTRODUCTION

1.1 Introduction of UR3 Cobot

Collaborative robots (cobots), such as the UR3, have gained popularity in Industry 4.0 due to their ability to operate safely alongside humans and in dynamic industrial contexts. Universal Robots' UR3 is a small and flexible robotic arm that is commonly used in manufacturing for tasks including assembly, pick-and-place, and quality checking. Its intuitive programming interface allows operators to teach the robot tasks easily, making it ideal for industries requiring rapid production changes and mass customization [1]. Unlike traditional automation, cobots can adapt to changes in the workplace while preserving efficiency and safety [2].

In smart factories, predictive maintenance for cobots is crucial for preventing machinery failures. The UR3 cobot's ability to identify defects during programmed tasks via real-time motion residual analysis provides an innovative solution to this problem. This predictive capability assures smooth operation and reduces downtime [3]. As cobot technology advances, the UR3 bridges the gap between research and real industrial applications, facilitating human-robot collaboration in modern production [1][3].

1.1.1 Uses Of Ur3 Cobots

The UR3 cobot is a versatile tool used in various industrial and research applications due to its flexibility, ease of programming, and precision. Some of the primary uses include:

- Assembly
- Material Handling
- Machine Tending
- Polishing and Grinding
- Screwing and Bolting
- Gluing and Dispensing
- Quality Inspection
- Research and Development

With its capacity to collaborate with people and automate repetitive tasks, the UR3 cobot is critical in improving productivity and maintaining consistent quality across a wide range of sectors. [4].

1.1.2 Study's goals

The primary aim of this study is to create an effective fault detection system for robotics and automation based on time series data from the UR3 cobot dataset. By analysing multi-dimensional data, including electrical currents, temperatures, joint speeds, and gripper activity, we aim to identify patterns and anomalies that signal potential faults. The project's goals include creating a strong machine learning model for accurate fault prediction, implementing a real-time monitoring system for proactive fault detection, and optimizing robotic performance and maintenance schedules to improve operational efficiency and safety in industrial applications**.**

1.1.3 Benefit

The study of fault detection in robotics and automation using time series data offers several significant benefits:

- **Improved Operational Efficiency:** By accurately predicting faults, the system minimizes downtime and enhances the overall productivity of robotic operations.
- **Enhanced Safety:** Early detection of potential issues helps prevent accidents, ensuring a safer working environment for human operators collaborating with robots.
- **Cost Reduction:** Implementing a proactive maintenance strategy reduces repair costs and extends the lifespan of robotic systems by addressing issues before they escalate.
- **Data-Driven Insights:** Leveraging time series data from the UR3 cobot enables informed decision-making regarding operational optimizations and maintenance scheduling.

- **Scalability:** The fault detection model can be adapted and scaled to various robotic applications across different industries, enhancing its utility and impact.
- **Contribution to Knowledge:** This research contributes to the existing body of knowledge in robotics and automation, promoting the advancement of smart manufacturing practices.

1.2 Motivation

With the growing integration of collaborative robots into various sectors such as healthcare, logistics, manufacturing, and agriculture, it is essential to ensure their reliable operation to boost productivity and safety. The implementation of these sophisticated robotic systems often puts them to dynamic and unpredictable environments, in which traditional fault detection techniques may prove insufficient. This limitation may result in unexpected downtime, endangered safety, and raised operational expenses, especially in environments where human-robot interaction is common. The rising complexity of cobot applications necessitates the development of innovative approaches that can effectively address these challenges [5][6].

Our study specifically aims to enhance fault detection strategies through the utilization of machine learning techniques applied to time series data collected from the UR3 cobot. By leveraging multi-dimensional data streams—such as joint angles, speeds, force readings, and other operational parameters—we seek to identify and predict potential faults before they escalate into critical failures. Moreover, the insights gained from this research contribute to the broader field of predictive maintenance, providing valuable methodologies that can be applied across various industrial domains to optimize the performance of collaborative robotics [7]. As we explore the integration of these advanced fault detection techniques, we aim to establish a robust framework for identifying and mitigating potential failures in collaborative robots. This proactive approach ensures that cobots can maintain optimal performance and reliability across various operational settings. By refining fault detection methodologies, our research contributes to enhancing system resilience and minimizing operational downtime, ultimately leading to more efficient and cost-effective robotic applications.

1.3 Expected Output

The study provides a comprehensive analysis of fault detection within collaborative robotic systems, specifically highlighting the UR3 CobotOps dataset. This dataset comprises 7,409 instances and 20 features, encompassing multi-dimensional time-series data associated with various operational parameters, including electrical currents, temperatures, and joint speeds. The expected outcomes of the study involve improved fault detection abilities by utilizing machine learning models such as XGBoost, LightGBM, Random Forest, and ExtraTrees. These models are anticipated to deliver high accuracy in identifying anomalies or faults within the robotic system, with performance levels reaching up to 97.5%. Furthermore, the effectiveness of the models will be determined through precision, recall, and F1-scores to ensure a well-rounded detection performance. In the field of predictive maintenance, regression models such as ARIMA are employed to anticipate future operational conditions, facilitating the detection of potential problems before they result in system failures. The preliminary findings from the ARIMA model indicate favorable error metrics (MAE: 0.0393, MSE: 0.0055, RMSE: 0.0741). However, additional model refinement and exploration of sophisticated methods like LSTM and Prophet may lead to enhanced prediction accuracy. The investigation also seeks to improve operational efficiency by examining the interactions between variables, including speed in relation to current or temperature variations across various joints. The findings from these analyses, combined with clustering techniques, will enhance robot efficiency, minimize downtime, and support improved decision-making in system maintenance. The goal of this study is to develop scalable and precise models that can be adapted across different robotic systems, advancing the fields of robotics and automation through the application of machine learning techniques.

1.4 Project Management and Finance

The integration of project management and finance in this study highlights the importance of structured planning and resource allocation for successfully executing a machine learning project focused on fault detection in robotics using time-series data. Well-established project management methodologies ensure efficient operations in each phase, while financial resources support infrastructure development, data collection, and model deployment. This study analyzes the UR3 CobotOps dataset, utilizing machine learning models such as XGBoost, Random Forest, and ExtraTrees to identify key indicators and risk factors related to operational faults.

Sound financial management is critical for optimizing resource use, from acquiring computational infrastructure to refining machine learning models. The synergy between finance and project management emphasizes the need for strategic planning, prudent decision-making, and flexible management techniques. This combination is essential for developing a robust and scalable fault detection model for robotics and automation.

The following table outlines the project timeline and tasks, with each phase aligned to its respective duration and deliverables.

Table 1.1 Project Management Table

Work	**Time**
Project Initiation	Week 1-2
Research and Data Collection	Week 3-6
Data Preprocessing	Week 7-10
Model Development	Week 11-15
Testing and Evaluation	Week 16-19
Reporting and Documentation	Week 20-24
Total	Week 24

A minimum of 24 weeks efforts was required to complete the project but may have varied due to various factor. The cost of the project has not been estimated and dose included a business model but if estimated roughly based on the day of work required and the number of LOC written, the project can be estimated to have cost 150k.

CHAPTER 2

LITERATURE REVIEW

2.1 Related Works

This paper rises the need for predictive maintenance in smart factories, focusing on the unique challenges of cobots executing programmable motions. It proposes an approach to enhance fault diagnosis and detection that is based on motion residual analysis. This method strengthens cobots' diagnostic skills by defining standards for fault analysis, interpreting detected values, and addressing the causes of faults. Although it is tested in a simulated environment, the method shows potential for real-world applications, with future plans to expand data collection and validation in industrial settings [2].

It addresses the challenge of opaque AI models in fault detection, which often obscure decision-making processes and undermine worker trust. By integrating Liang-Kleeman Information Flow analysis with Fuzzy Cognitive Maps (IF-FCMs), the authors enhance the interpretability and reliability of fault diagnosis in industrial robotics. Using the UR3 CobotOps Dataset, the study demonstrates that IF-FCMs outperform traditional FCMs in diagnostic accuracy and interpretability, effectively distinguishing genuine causal links from spurious correlations. The application of SMOTE-ENN and Stratified K-Fold Cross-Validation techniques further improves model robustness. Despite its promising results, the study recommends further validation in real-world industrial settings and suggests integrating other cognitive mapping approaches to expand its applicability [8].

This paper presents a novel method for detecting high-impedance faults (HIFs) using frequency-band energy curves (FBEC). The study, which uses field and simulation data, applies continuous wavelet transform (CWT) and Gaussian smoothing for feature extraction and noise reduction, achieving an accuracy of 86.5% and an F1-score of 0.87. The method also shows resilience to noise with 77.3% accuracy at 20 dB SNR. Future research directions include enhancing noise resilience, exploring new smoothing techniques, and applying a harmonic criterion to reduce detection errors [9].

I uses both simulation and field data to study the detection of voltage droop-induced timing faults caused by hardware Trojans. Utilizing time-series analysis and machine learning algorithms such as Random Forest, SVM, and KNN, the study achieved notable results: for simulation data, an accuracy of 95.4%, TPR of 98%, FPR of 5%, and an F1-score of 0.97; for field data, an accuracy of 86.5%, TPR of 90%, FPR of 7%, and an F1-score of 0.89. Future research should focus on exploring more diverse types of hardware Trojans and various real-world scenarios [10].

This paper focuses on predicting vibrations in robotic arms caused by joint flexibility, utilizing a double inertia elastic system model to derive the transfer function from electromagnetic torque to arm vibration. The study employs experimental data from a single-joint robot testbed and an articulated industrial robot, but it does not specify a particular dataset used. The algorithms used include a direct parametric method for identifying key parameters like equivalent stiffness and damping, as well as a vibration dynamics model to predict the vibration spectrum. The results indicate that the predicted vibration signals closely match actual signals, validating the proposed prediction method. The research gap identified is the need for improved accuracy in vibration prediction for industrial robots, which are often affected by field noise and lack high precision. Future directions suggested include enhancing parameter identification methods and exploring the applicability of the proposed model to more complex robotic systems [11].

It utilized a publicly available dataset that was transformed into a binary format (fault vs. normal operation) and addressed class imbalance through random oversampling and SMOTE methods. The primary algorithm employed was a genetic programming algorithm, specifically a symbolic classifier, which was trained using a 5-fold cross-validation process. The results demonstrated high classification performance, with the best metrics on the SMOTE dataset showing accuracy (ACC) of 0.99, area under the curve (AUC) of 0.99, precision of 0.992, recall of 0.9893, and F1-score of 0.99. The research identified a gap in existing literature regarding the computational intensity of machine learning models for fault detection, which often lack simple mathematical expressions for practical use. Future directions include creating a balanced original dataset and applying synthetic data generation techniques to enhance the dataset for better model training and validation [12].

In the study, researchers developed a method for manipulating a 6-DOF UR3 robot using deep learning algorithms applied to simulated data. The developed models achieved high accuracy in manipulation tasks, significantly improving control precision and efficiency. However, the study highlights the necessity for future research to validate the findings using real-world data, refine the deep learning models, and integrate additional sensors to enhance manipulation accuracy and robustness [13].

In this paper, researchers proposed a method for anomaly detection using matrix completion techniques applied to real-world datasets (D1, D2, and D3). Various algorithms were compared, including JumpStarter, OmniAnomaly, MSCRED, LeSiNN, RRCF, and FMAD. The FMAD algorithm achieved impressive results with precision ranging from 78.6% to 86.3%, recall from 92.2% to 98.3%, and F1 scores from 84.6% to 91.7%. FMAD outperformed the other algorithms across all datasets. The study suggests future research should focus on optimizing the algorithm for larger datasets and exploring its potential for real-time anomaly detection [14].

In the study, the authors developed a method for process monitoring and fault prediction utilizing the Bag-of-Words approach on real-world process monitoring datasets. The algorithms compared included SVM and Random Forest, with the Bag-of-Words model achieving a precision of 90.4%, recall of 89.7%, and an F1 score of 90.0%. The study suggests that future research should focus on integrating more complex feature extraction methods and applying deep learning techniques to improve prediction accuracy and robustness [15].

In the study, researchers applied Convolutional Neural Networks (CNNs) to proprietary industrial process data for fault detection. The model achieved an F1 score of 0.92, with a precision of 0.90 and a recall of 0.94. While the results demonstrate the effectiveness of CNNs in detecting faults within the dataset, the study highlights the need for future research to focus on real-time implementation and testing across diverse industrial scenarios to ensure broader applicability and robustness [16].

This paper explores the segmentation of multivariate time-series data in industrial settings using a dynamic latent variable (DLV) model. The study utilizes publicly available industrial datasets and applies the DLV model in combination with K-means clustering for the segmentation process. The results demonstrate an improvement in segmentation accuracy and interpretability compared to existing methods. The authors suggest further research to explore the generalization of the model to various types of industrial processes, indicating a potential research gap in this area. The key measures used in the study include segmentation accuracy and interpretability [17].

This review paper provides a comprehensive overview of various fault diagnosis methods in robotic systems, focusing on both hardware and software faults. The study utilizes various datasets for robot systems and applies algorithms such as Fuzzy Logic and Artificial Neural Networks (ANN) to classify different types of faults and diagnose methods used in robotics. The results highlight that hardware faults, particularly those related to sensors and actuators, are the most critical and need to be identified and corrected in real-time. The paper suggests a research gap in the integration of different fault diagnosis methods and the application of these methods to non-rigid robots. Key measures used in the study include fault detection accuracy and real-time performance [18].

The paper presents a method for predicting vibrations in the articulated arms of industrial robots, addressing the challenges posed by joint flexibility. It combines the internal transfer function of the drive system with parameter identification under external excitation to model the dynamics of the robot joint system as a double inertia elastic system. The authors establish a transfer relationship from the motor's electromagnetic torque to the arm's vibration, allowing for the identification of key parameters such as stiffness and damping through impact tests. Experimental results validate the effectiveness of the proposed method, demonstrating

its capability to predict vibration spectra accurately during robot motion, thus enhancing the reliability of industrial robotic systems [19].

This paper utilizes a dataset extracted from UE Systems Co., which includes 10 distinct fault classes related to bearings and pipelines, comprising 20,000 samples. The algorithms employed in this study include k-Nearest Neighbor (KNN), Logistic Regression (LR), Decision Tree (DT), Gaussian Naive Bayes (GNB), Support Vector Machine (SVM), and a meta-classifier that combines these models. The results indicate that the proposed meta-classifier achieved an impressive accuracy of 93% through k-fold cross-validation, with a rapid execution time of 11 ms on a 64 MHz Cortex-M4 microcontroller, demonstrating its potential for real-time monitoring applications. The research gap identified includes the method's reliance on extensive historical data and its focus on time-domain features, which may limit its applicability in scenarios with limited data and overlook valuable insights from frequency-domain analysis. Future directions involve enhancing the model's deployment capabilities across various microcontroller units (MCUs) and optimizing PCB designs to improve reliability and performance in diverse industrial environments [20].

The paper addresses the challenges in condition-based predictive fault detection within the automotive sector, proposing a novel multi-sensor, multi-fault resilient architecture. The study utilized a dataset comprising 300,000 data points to test and tune the proposed algorithms, which include Support Vector Machine (SVM) models for fault detection and identification. The results indicate that different sensors exhibit varying accuracies in detecting specific fault types, with the paper highlighting the importance of considering noise in simulations, as it can significantly impact detection accuracy. The research identifies gaps in the current methodologies, particularly in multi-sensor, multi-fault detection and condition-based fault prediction. Future directions suggested include expanding the definition of faults to encompass a broader range of sensor system faults and improving the simulation of data to account for noise, which could enhance the validity of the results [21].

This paper utilizes a dataset derived from a robotic fuse quality test bench, which involved raw multi-sensor fusion data from a robotic manipulator. The study employed various algorithms, including a two-dimensional Convolutional Neural Network (2DCNN), BiLSTM, and 1D-CNN, among others, to enhance diagnostic accuracy. The results demonstrated exceptional performance, with the proposed model achieving an average accuracy of 99.98%

and a testing accuracy of 99.74% for four fault classes, significantly outperforming conventional methods. The research gap identified in the paper highlights the challenges faced by conventional machine learning methods in extracting meaningful features from complex datasets, suggesting a future direction focused on improving deep learning techniques for heterogeneous tabular data and exploring more efficient computational methods to reduce execution times in industrial settings [22].

It reviews various fault detection and diagnosis (FDI) techniques specifically for multi-degree-of-freedom (DOF) robots, highlighting the challenges and advancements in the field. The paper does not specify a particular dataset used, but it emphasizes the importance of large datasets for training deep learning models. It discusses both measurement-based and model-based approaches, although specific algorithms are not detailed in the provided contexts. The results indicate that existing methods primarily identify faults in one or two DOFs, suggesting a need for more comprehensive approaches that can handle the complexities of multi-DOF robots. The research gap identified includes the limited ability of current models to generalize across different operating conditions and the black-box nature of deep learning models, which complicates fault diagnosis. Future directions proposed include the exploration of explainable AI techniques, advanced sensor fusion, and the deployment of FDI systems at the edge of networks to enhance real-time decision-making and operator trust [23].

It provides a comprehensive overview of various datasets used in time series anomaly detection (TSAD), categorizing them into multivariate, univariate, and mixed types, with links to access these datasets. The study discusses numerous deep learning algorithms, including Autoencoders (AEs) and Deep Autoencoding Gaussian Mixture Models (DAGMM), highlighting their strengths and weaknesses in detecting anomalies. The results indicate that specialized deep learning models significantly enhance the detection of anomalous patterns across various domains, such as finance and healthcare. However, the paper identifies research gaps, particularly in the integration of temporal information in models and the need for improved interpretability of results. Future directions suggest a focus on developing models that can better handle the complexities of time series data and address the challenges outlined in the survey [24].

This paper provides a comprehensive overview of various datasets used in time series anomaly detection, categorizing them into multivariate, univariate, and general-purpose

datasets. The survey discusses several deep learning algorithms, including Autoencoders (AEs) and Denoising Autoencoders (DAEs), highlighting their effectiveness in detecting anomalies in time series data. The results indicate that these deep learning models can significantly enhance anomaly detection capabilities compared to traditional methods, with 64 recent models being analyzed and categorized. The paper identifies research gaps, particularly in the challenges of applying deep learning models to time series data, and suggests future directions for research, emphasizing the need for further exploration of model robustness and adaptability across various domains [25].

It utilizes a dataset collected from a UR5e collaborative robot during pick-and-place tasks, which includes data on the robot's joints, force-torque readings, and program execution times. The study evaluates various anomaly detection algorithms, including data-driven methods and autoencoders, to identify exogenous anomalies such as collisions and environmental changes. The results indicate that several methods achieved high detection performance, particularly when leveraging knowledge of the robot's program structure, although performance declined in scenarios involving rarely-visited program branches and stochastic trajectories. The research gap identified is the need for more robust anomaly detection methods that can handle complex real-world scenarios, suggesting future directions could include enhancing algorithms to better adapt to dynamic environments and integrating more comprehensive datasets for training [26].

This paper discusses the advancements in collaborative robot (cobot) programming, focusing on communication, optimization, and learning features that enhance human-robot collaboration in industrial settings. The paper highlights the need for more annotated datasets relevant to industrial collaborative tasks, although specific datasets used in the research are not mentioned in the provided contexts. Various algorithms are discussed, including Learning from Demonstration (LfD) algorithms, which require user input for effective operation. The results indicate that while cobots can learn policies through operator guidance, there is a significant gap in making these technologies accessible to non-expert users. The research gap identified includes the need for user-friendly interfaces and reliable perception solutions tailored for industrial applications, as well as the challenge of intellectual property restrictions on data sharing. Future directions emphasize the importance of developing intuitive UIs, automating

processes in the learning pipeline, and fostering collaboration between researchers and industry to create practical, standardized solutions [27].

This paper by Daniel Silvestre, João Hespanha, and Carlos Silvestre addresses the critical issue of detecting and isolating faults in cyber-physical systems, particularly within smart grids. The study utilizes the IEEE 14 bus testbed dataset to evaluate the performance of fault detection methods. The authors employ Set-Valued Observers (SVOs) as the primary algorithm for both centralized and decentralized fault detection strategies. The results indicate that the SVOs can effectively detect faults, with simulations showing that detection times are significantly influenced by the amount of available information, highlighting a need for further research to optimize performance under limited data conditions. The research gap identified includes the challenge of detecting faults in the presence of disturbances and noise, suggesting future directions that involve enhancing detection algorithms to improve robustness against such uncertainties and exploring more comprehensive datasets for validation [28].

The paper presents a comprehensive concept for condition monitoring of industrial robots, focusing on self-tests and MEMS-based sensor nodes to assess wear conditions. It does not specify the year of publication or the dataset used, but it highlights the application of machine learning algorithms, particularly Support Vector Machine (SVM), Random Forest (RF), and k Nearest Neighbours (kNN) for classifying wear conditions based on vibration signals. The results indicate that while the SVM classifier was validated using synthetic data due to the lack of real data, the approach shows promise for predictive maintenance. A significant research gap identified is the absence of labeled case data in practical applications, which hinders the effectiveness of supervised learning methods. Future research directions include exploring automated labeling of measurement data to enhance the training of machine learning models [29].

It utilizes a dataset derived from simulation scenarios involving drone delivery systems, which are essential for various applications such as COVID-19 sample tests and package deliveries. The algorithms employed in the study include a one-dimensional convolutional neural network (1D-CNN) and an encoder-decoder long short-term memory (LSTM) framework, which are designed to predict air traffic congestion while considering the complexities of unmanned aircraft traffic management (UTM). The results indicate that the proposed model achieves the smallest root mean square error (RMSE) and absolute mean

percentage error (AMPE) compared to existing methods, with an RMSE of 0.4 and AMPE of 3.4, although it incurs higher computation times. The research gap identified is the limited exploration of air traffic patterns for UTM, particularly in dynamic environments, and the future direction suggests enhancing prediction accuracy while addressing computational efficiency [30].

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Methodology

This mind map represents an approach for Fault Detection in UR3 Robotics and Automation. It explores three key techniques: Regression, Classification, and Clustering. Under regression, various models like LSTM, Prophet, and ARIMA are used for time-series prediction, with metrics such as R^2, MSE, RMSE, and MAE for evaluation. Classification involves techniques like AdaBoost, Random Forest, and XGBoost, using metrics like Precision, Recall, and Accuracy for performance evaluation. Lastly, Clustering focuses on the K-Means algorithm for unsupervised fault detection.

Figure 3.1 Work Flow

3.2 Data Analysis

Our dataset, based on the UR3 CobotOps Dataset from the UCI repository, includes 7,409 instances and 20 key features for real-time cobot condition monitoring. These features, such as electrical currents (Current_J0 to Current_J5), temperatures (Temperature_J0 to Temperature_J5), and rotational speeds (Speed_J0 to Speed_J5), are collected via MODBUS and RTDE protocols. Additionally, binary status flags like Robot_ProtectiveStop (1 for stop, 0 for no stop) and Grip_lost (1 for lost grip, 0 for no lost grip) monitor critical events. This dataset is ideal for classification, regression, and clustering tasks, supporting advanced research in fault detection, predictive maintenance, and operational optimization in robotics.

Table 3.2.1 Feature Description

Feature Name	Description
Num	An identifier for each data point, often a simple index.
Timestamp	The specific date and time when the data point was recorded.
Current_J0	The electrical current (in amperes) running through Joint 0 of the cobot.
Temperature_T0	The temperature (in degrees Celsius) at Joint 0 of the cobot.
Current_J1	The electrical current running through Joint 1.
Temperature_J1	The temperature at Joint 1.
Current_J2	The electrical current running through Joint 2.
Temperature_J2	The temperature at Joint 2.
Current_J3	The electrical current running through Joint 3.
Temperature_J3	The temperature at Joint 3.
Current_J4	The electrical current running through Joint 4.
Temperature_J4	The temperature at Joint 4.
Current_J5	The electrical current running through Joint 5.
Temperature_J5	The temperature at Joint 5.
Speed_J0	The rotational speed (in degrees per second) of Joint 0.

Speed_J1	The rotational speed of Joint 1.
Speed_J2	The rotational speed of Joint 2.
Speed_J3	The rotational speed of Joint 3.
Speed_J4	The rotational speed of Joint 4.
Speed_J5	The rotational speed of Joint 5.
Tool_current	The electrical current running through the tool (such as a gripper) attached to the cobot.
Cycle	The operation cycle count, indicating how many cycles the cobot has completed.
Robot_ProtectiveStop	A flag indicating if a protective stop was triggered to prevent damage or ensure safety (1 for stop, 0 for no stop).
Grip_lost	A flag indicating if the cobot's gripper lost grip on the object it was handling (1 for lost grip, 0 for no lost grip).

3.3 Classification Techniques for the UR3 Cobot

3.3.0 Data preprocess

To clean and preprocess the data, the leading and trailing quotes from the 'Timestamp' column were removed, and the column was converted into datetime objects for accurate time-based indexing. The 'Timestamp' column was then set as the Data Frame index using *`df.set_index('Timestamp', inplace=True)`* to facilitate time-series analysis. Missing values in the dataset were handled by applying forward fill using *`df.fillna(method='ffill')`*, which fills missing entries with the last valid observation. Finally, *`df.isnull().sum()`* was used to verify that all missing values were successfully handled.

3.3.1 Model Training

Extra Trees Classifier (ETC): The Extra Trees Classifier, or Extremely Randomized Trees, is an ensemble learning method used for classification tasks. Unlike traditional decision trees, Extra Trees classifiers introduce randomness in the tree-building process by selecting splits at random points for each feature, rather than choosing the best possible split based on criteria like Gini or entropy. This added randomness often leads to a reduction in overfitting, especially in high-dimensional datasets, and can improve model robustness. While the model excels in computational efficiency and handles both categorical and numerical data well, its random nature might result in less interpretability compared to traditional decision trees. Nonetheless, the Extra Trees Classifier is valued for its speed and ability to capture complex patterns within the data.

Random Forest (RF): A cornerstone in ensemble learning, Random Forest constructs multiple decision trees during training, amalgamating their outputs to determine the class mode in classification tasks or the mean prediction in regression scenarios. Renowned for its robustness against noise and scalability, this method significantly mitigates overfitting by infusing randomness into the tree-generation process. By harnessing the collective wisdom of diverse decision trees, Random Forest not only enhances predictive accuracy but also offers insights into intricate data patterns while maintaining computational efficiency.

AdaBoost (AB): AdaBoost, short for Adaptive Boosting, is a powerful ensemble learning technique that harnesses the collective strength of multiple weak learners to construct a robust and accurate predictive model. It operates by iteratively adjusting the weights of training instances based on their classification performance in preceding iterations, thereby focusing more on difficult-to-classify examples. By adaptively altering the training data distribution, AdaBoost strives to enhance the model's efficacy on the training set, with a primary focus on correctly classifying previously misclassified examples. This iterative refinement process continues until a strong learner, capable of effectively generalizing to unseen data, is achieved. AdaBoost's versatility and ability to combine diverse weak learners make it a popular choice for a wide range of classification tasks, demonstrating impressive performance across various domains.

XGBoost (Extreme Gradient Boosting): XGBoost is a highly efficient and scalable machine learning algorithm based on gradient boosting, designed for both classification and regression tasks. It builds an ensemble of weak learners, typically decision trees, where each new tree corrects the errors made by the previous ones. XGBoost stands out due to its regularization techniques, which help prevent overfitting, and its ability to handle sparse data and missing values efficiently. The model supports parallel processing, making it computationally faster than traditional gradient boosting. While it can be complex to tune due to the number of hyperparameters involved, XGBoost remains one of the most popular choices in competitive machine learning due to its high accuracy, flexibility, and performance on large datasets.

AdaBoost (Adaptive Boosting): AdaBoost is a boosting algorithm that combines multiple weak learners, typically decision stumps, to create a strong classifier. The algorithm works by sequentially fitting weak models to the data, with each model focusing on the mistakes made by the previous ones. Weights are assigned to misclassified instances, increasing their importance in subsequent iterations, allowing the model to progressively correct errors. AdaBoost is particularly effective at improving the performance of weak learners and works well with both classification and regression tasks. However, it is sensitive to noisy data and outliers, as these can be assigned disproportionately high weights. Despite this limitation,

AdaBoost is widely used for its simplicity, ability to reduce bias, and solid performance on a variety of datasets.

LightGBM (Light Gradient Boosting Machine): LightGBM is a gradient boosting framework that uses tree-based learning algorithms, designed for high performance and efficiency. It excels in handling large datasets with high-dimensional features by utilizing techniques such as leaf-wise growth and histogram-based decision tree learning, which make it faster and more memory-efficient compared to other boosting algorithms like XGBoost. LightGBM supports both classification and regression tasks, and it handles categorical features natively without requiring extensive preprocessing. One of its key advantages is the ability to scale to massive datasets with millions of instances while maintaining high accuracy. However, LightGBM can be prone to overfitting if the data is small or if hyperparameters are not carefully tuned. Despite this, it remains a popular choice for its speed, scalability, and performance in competitive machine learning environments.

Hybrid ML Model Combination: These combinations indicate the use of ensemble learning techniques, where two different algorithms are merged to enhance predictive performance. The first label mentions "Hybrid (Extra Trees + LightGBM)", combining the Extra Trees classifier with the LightGBM model. The second is "Hybrid (XGBoost + LightGBM)", a blend of XGBoost and LightGBM, both highly popular gradient boosting frameworks. Finally, "Hybrid (RF + LightGBM)" integrates Random Forest (RF) with LightGBM. These hybrids suggest a focus on optimizing model performance by leveraging the strengths of each component algorithm.

3.3.2 Visualization

The comprehension of algorithms and their composite characteristics is significantly enhanced through the visualization of datasets and the analysis of their outcomes.

3.3.2.1 Analysis of Current and Temperature Values Over Time

The figure, titled Analysis Current and Temperature Values Over Time**,** presents two time series plots. The top plot displays current values for six different channels (J0 to J5) over a day-long period. It shows a period of high variability in current readings at the beginning and end of the time series, with a stable period in between. The bottom plot depicts temperature values for the same six channels. The temperature data shows a gradual increase over time, with different channels stabilizing at distinct temperatures. Both plots highlight the different behaviors and characteristics of current and temperature readings across various channels throughout the observed period.

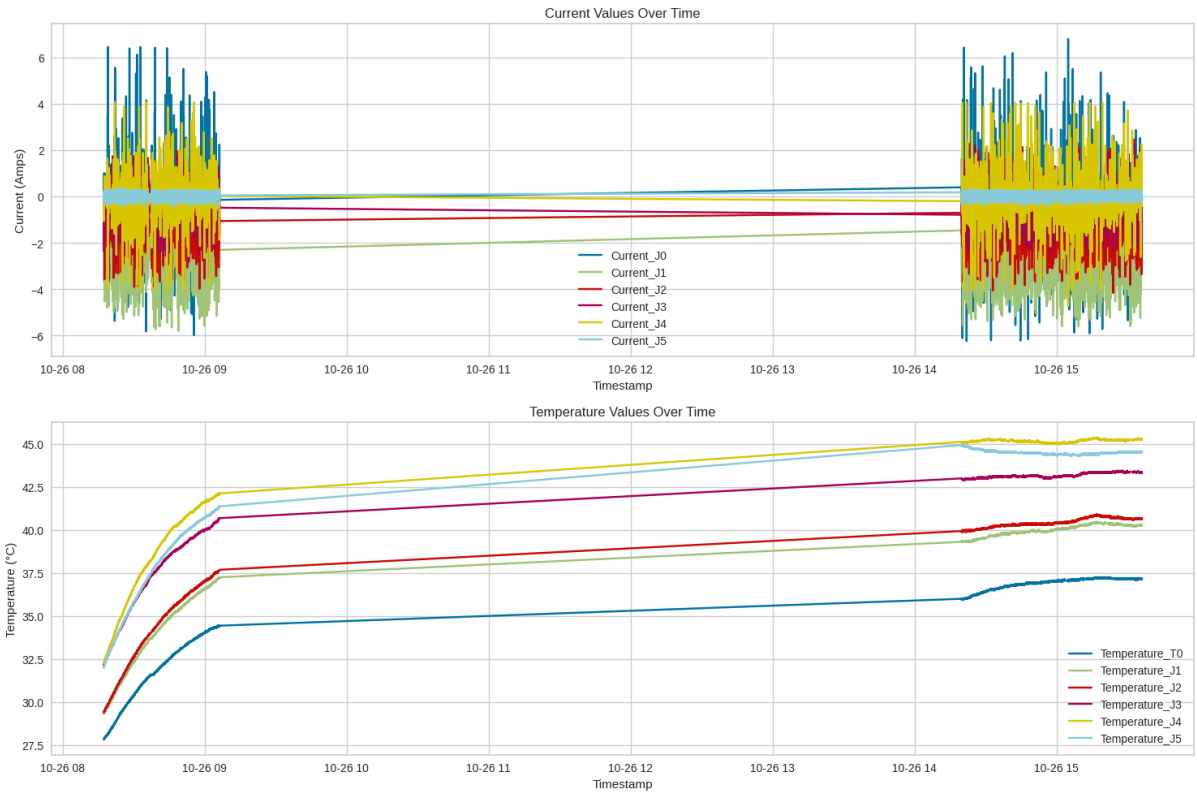

Figure 3.2 Analysis Current and Temperature Values Over Time

3.3.2.2 Speed vs Current Scatter Plots

The figure displays a grid of scatter plots comparing the speed and current values across six different joints (J0 to J5). Each row represents a different speed variable, while each column corresponds to a different current variable. The scatter plots illustrate the relationship between speed and current for each pair, with varying colours enhancing visual distinction. Patterns and correlations, such as clusters or linear relationships, can be observed across different joint pairs, providing insights into their operational characteristics. The diverse colour palette and clear labelling facilitate easy interpretation and comparison of the data relationships.

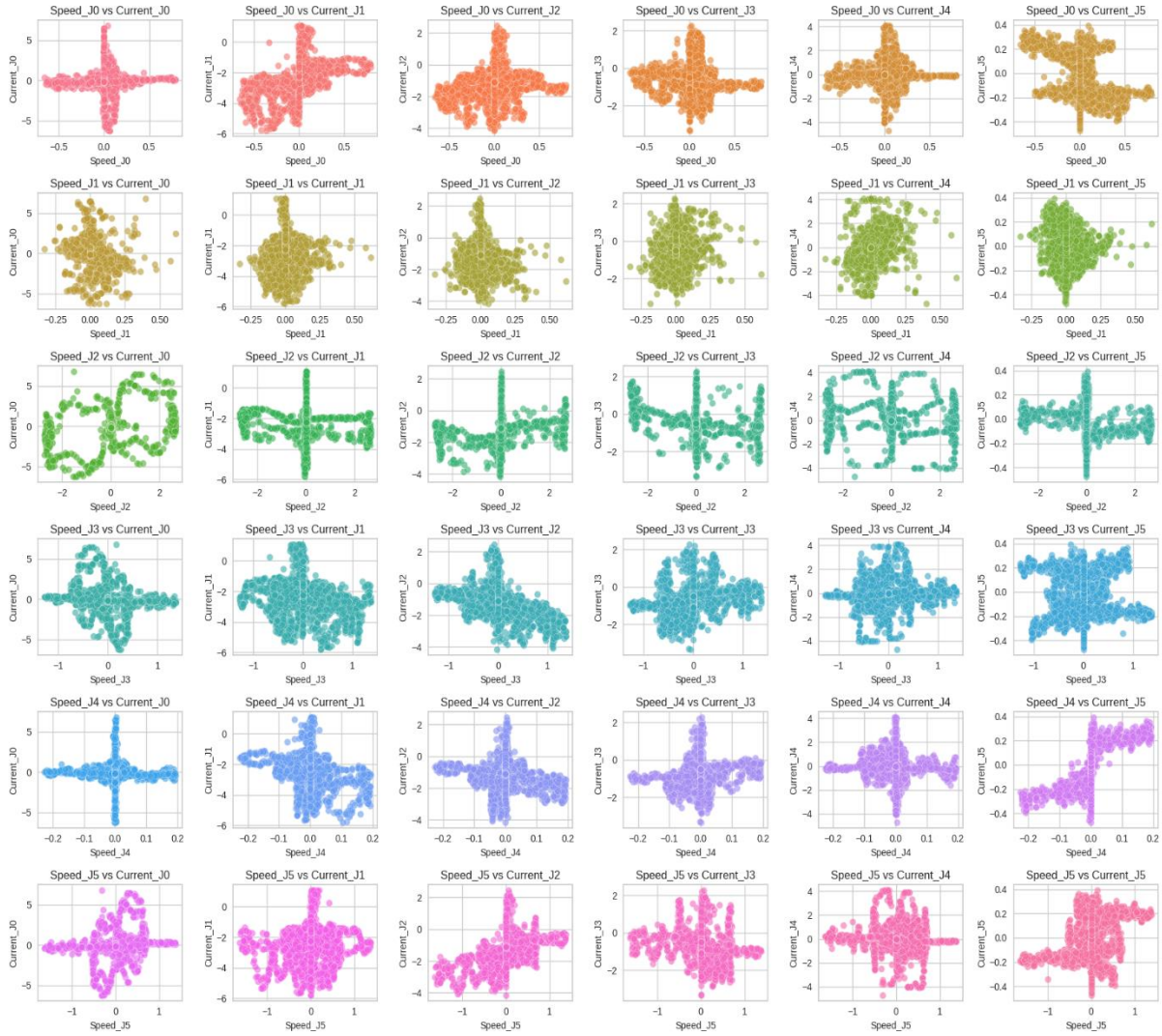

Figure 3.3 Speed vs Current Scatter Plots

3.3.2.3 Temperature Histograms

The figure showcases individual histograms for six temperature channels: Temperature_T0, Temperature_J1, Temperature_J2, Temperature_J3, Temperature_J4, and Temperature_J5. Each subplot represents the frequency distribution of temperature readings within a specific channel, revealing distinct patterns and ranges for each. The histograms are color-coded using a gradient palette, enhancing visual differentiation. Notable peaks in the distributions suggest common operating temperature ranges, with each subplot clearly labeled for easy interpretation. This comprehensive view helps in understanding the temperature characteristics and variations across different channels in the dataset.

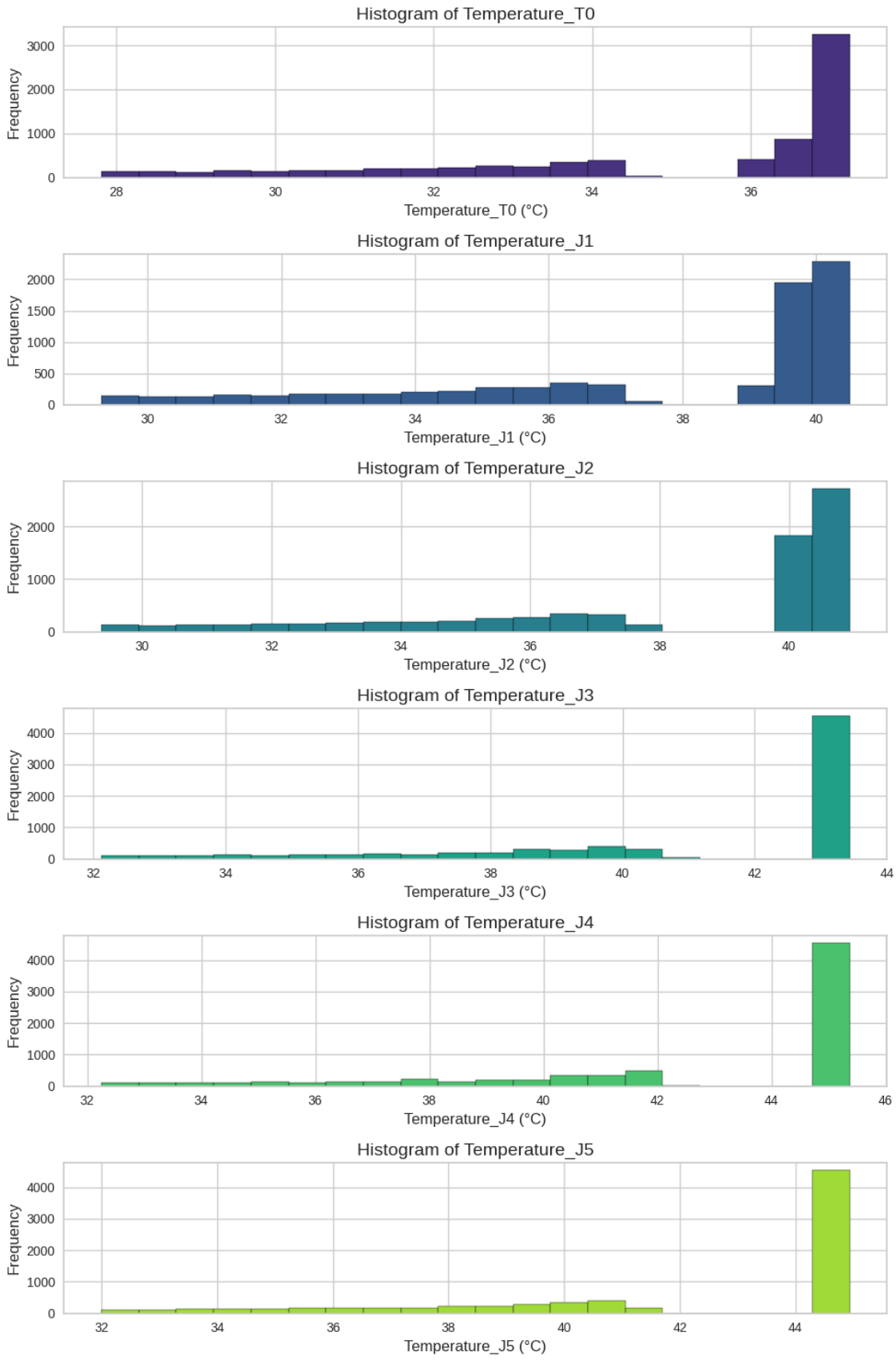

Figure 3.4 Temperature Histograms

3.3.2.4 Speed Values Over Time

The figure illustrates the variation in *speed values across six joints (J0 to J5) over a specified time period*. Each line in the plot represents the speed of a particular joint, color-coded for clarity. The x-axis denotes the timestamps, while the y-axis represents the speed values in units. The plot shows periods of high variability and activity at the beginning and end of the time range, with a notable drop in activity in the middle. This pattern indicates fluctuations in the speed readings, likely corresponding to different phases or states of operation. The legend below the plot provides clear identification of each joint's speed data.

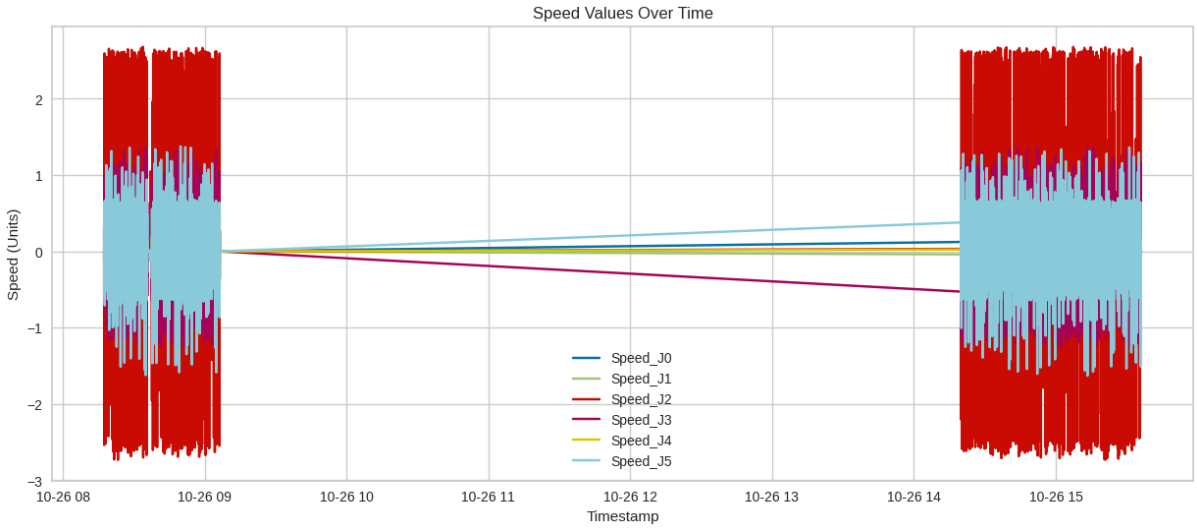

Figure 3.5 Speed Values Over Time

3.3.3 Model Evolution

Throughout the meticulous assessment stage of the algorithm, a comprehensive set of essential performance metrics, including accuracy, precision, recall, and the F1 score, were conscientiously utilized. The metrics mentioned are essential instruments for evaluating the efficacy and predictive ability of each constructed model, offering a nuanced comprehension of their capabilities.

Accuracy:

Accuracy is a crucial metric that quantifies the ratio of correctly predicted instances to the total number of cases assessed. From a mathematical perspective, accuracy can be defined as the quotient of true positives (TP) and true negatives (TN) divided by the total number of instances, encompassing both accurate and inaccurate predictions. A high accuracy score demonstrates the model's proficiency in accurately predicting outcomes across all categories, thus highlighting its overall effectiveness.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Precision:

Precision quantifies the degree of correctness in positive predictions, emphasizing the ratio of correctly identified positive instances (TP) to the total number of instances predicted as positive. The calculation involves dividing the true positive rate (TP) by the sum of TP and false positives (FP). A high precision score indicates that the model effectively reduces the occurrence of false positives, rendering it especially advantageous in situations where the misclassification of negative instances as positive has substantial implications.

$$precision = \frac{TP}{TP+FP} \quad (2)$$

Recall:

Recall, also referred to as sensitivity, measures the model's capacity to accurately detect positive instances among the total number of positive instances in the dataset. The calculation involves dividing the number of true positives (TP) by the number of true negatives (FN). A

high recall score indicates that the model demonstrates exceptional performance in accurately identifying all pertinent instances of a specific class, thereby reducing the occurrence of false negatives. This metric is particularly vital in situations where the absence of positive instances can have substantial consequences.

$$precision = \frac{TP}{TP+FN} \quad (3)$$

F1 score:

The F1 score is a metric that effectively balances precision and recall, rendering it highly valuable, particularly in situations where there are imbalances between classes. The F1 score provides a comprehensive assessment of a model's performance by calculating the harmonic mean of precision and recall. It is especially beneficial in scenarios where both incorrect positive and negative results have significant implications, as it encompasses the balance between these two measures. A higher F1 score is indicative of a more robust model, as it signifies a better balance between precision and recall.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

3.4 ARIMA Model for the UR3 Cobot

The Autoregressive Integrated Moving Average (ARIMA) model is a widely used and versatile approach for time series forecasting, as it explains a given time series based on its past values and past forecast errors. This approach makes ARIMA especially suitable for predicting future values by capturing patterns within the data itself, independent of external predictors. To prepare the dataset for ARIMA modeling, the 'Timestamp' column was first cleaned and converted to a `datetime` format to ensure proper time ordering. Missing values were handled by forward-filling, which propagates the last known value to fill any gaps, maintaining data continuity without introducing bias.

A key assumption for ARIMA modeling is that the time series is stationary, meaning that its statistical properties, such as mean and variance, remain constant over time. To confirm stationarity, the Augmented Dickey-Fuller (ADF) test was conducted, yielding a test statistic of -14.45 and a p-value of 7.08e-27. These results, significantly below the 0.05 significance level, reject the null hypothesis of a unit root, indicating that the time series is indeed stationary. As a result, no further differencing is needed, simplifying the modeling process by allowing ARIMA to proceed without additional transformations. Although the ADF test confirms stationarity, an autocorrelation plot can provide further insights by showing the correlation of the series with its lagged values, which aids in identifying patterns like seasonality or trends and in selecting the appropriate ARIMA parameters.

An ARIMA model is characterized by three parameters: the autoregressive (AR) term, which captures the influence of past values; the integrated (I) term, which reflects the degree of differencing needed to achieve stationarity; and the moving average (MA) term, which accounts for the dependency between observations and residual errors. In this case, given that the ADF test confirms stationarity, the integrated component (I) will be zero, and the appropriate values for the AR and MA terms can be determined by examining the autocorrelation patterns within the data.

3.4.1 Steps of ARIMA Model:

The image presents a flowchart illustrating the process of time series forecasting. It begins with "Load the Dataset," followed by "Data preprocessing," which prepares the dataset for analysis. The next step is to "Check Stationarity," where the time series is evaluated for constant statistical properties over time. If the series is not stationary, it moves to "Make it stationary" to stabilize the data. Afterward, the process involves determining the "p, d, q Values," which are the parameters for an ARIMA model. The final step is "Forecasting," where future values of the time series are predicted based on the model.

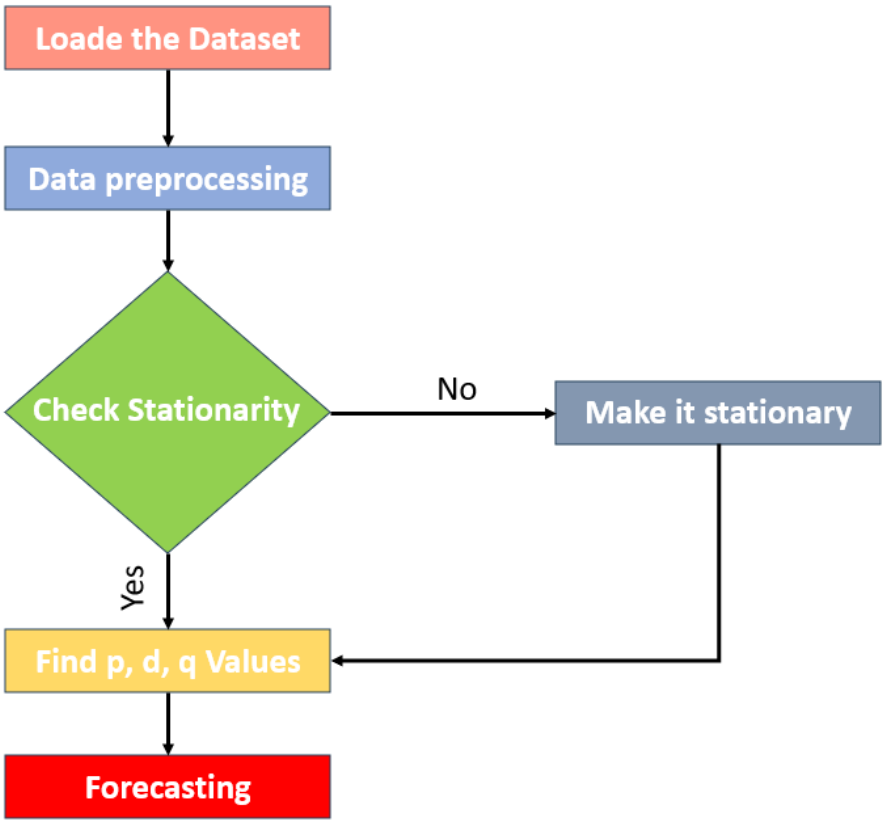

Figure 3.6 Steps of ARIMA Model

3.4.2 Visualization of ARIMA Model

The visualization of datasets and the analysis of their results greatly improves our understanding of algorithms and their composite properties.

3.4.2.1 Comparison of Original and Differenced Tool Current Time Series

This figure presents two time series plots of tool current readings. The top plot shows the original tool current data, while the bottom plot displays the differenced tool current data. Differencing is used to remove trends and make the data stationary, as evidenced by the stabilized fluctuations in the differenced plot compared to the original. Both plots illustrate tool current values over time, highlighting periods of activity and inactivity. Differencing appears to reduce variance and make the data more suitable for time series modeling.

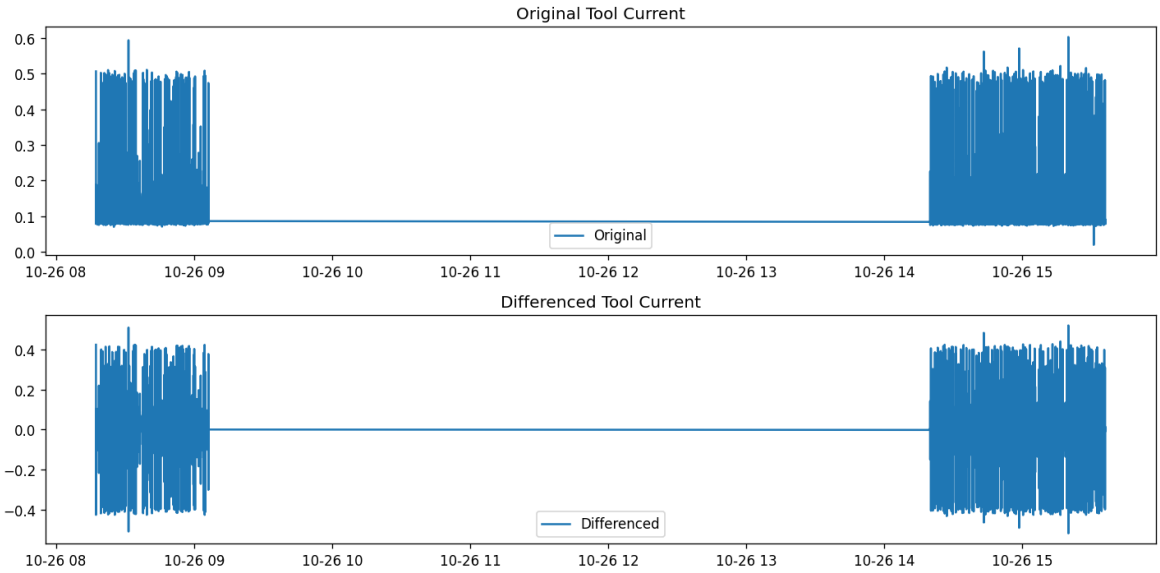

Figure 3.7 Comparison of Original and Differenced Tool Current Time Series

3.4.2.2 Tool Current vs. Index (Last 500 Records)

This line plot shows the tool current measurements over the last 500 recorded instances, plotted against their corresponding index values. The x-axis represents the time index, while the y-axis represents the tool current. The plot reveals periods of low tool current interspersed with sharp spikes, indicating fluctuations in tool usage or load. This pattern of recurring peaks suggests variability in the tool's operation, which could be indicative of changes in workload or different operational phases. The plot provides insights into the short-term dynamics of tool current over the selected time window.

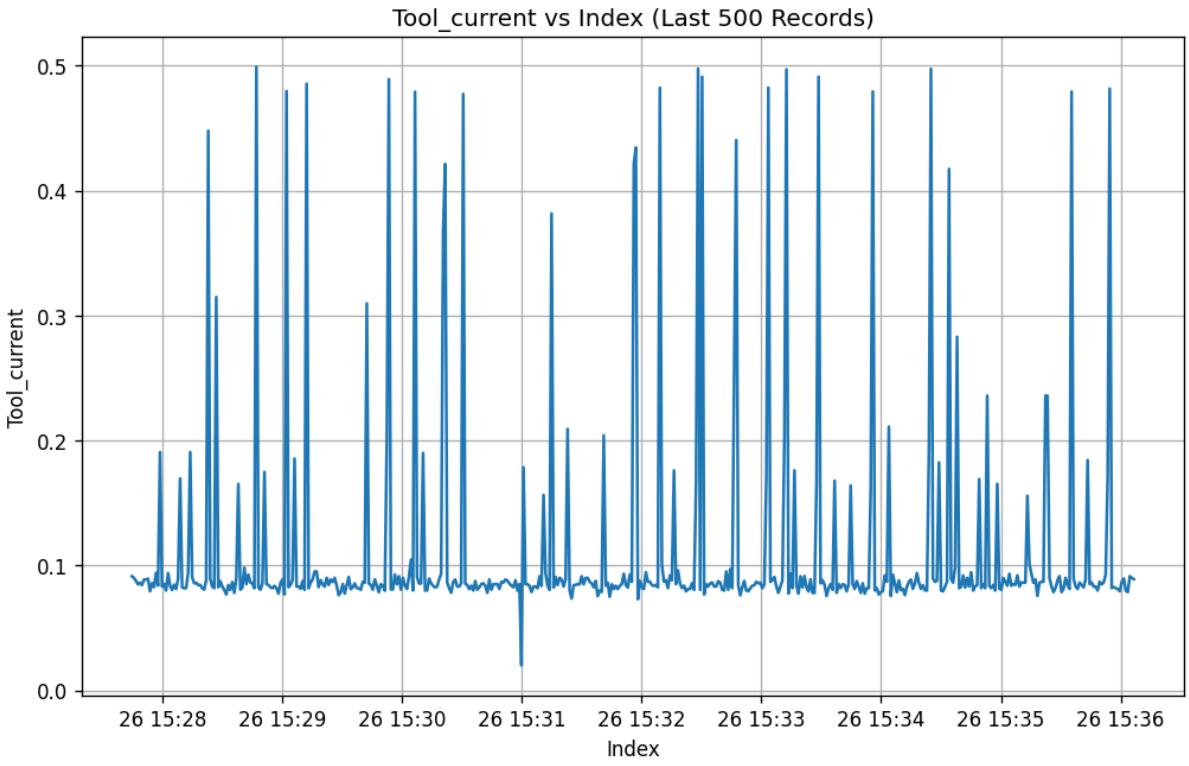

Figure 3.8 Tool Current vs. Index (Last 500 Records)

3.5 LSTM on UR3 Cobot

Long Short-Term Memory (LSTM) networks are a specialized type of Recurrent Neural Network (RNN) that excel in handling sequential data by maintaining long-term dependencies. Traditional RNNs struggle with the vanishing gradient problem, which makes it difficult for them to learn patterns in long sequences, as the gradients diminish during backpropagation. LSTMs overcome this limitation by introducing memory cells and three key gates—the input gate, forget gate, and output gate—that work together to regulate the flow of information.

The input gate controls how much new information is allowed into the memory cell, the forget gate determines how much of the existing information should be retained or discarded, and the output gate decides what part of the memory will be passed to the next step in the sequence. This architecture enables LSTMs to retain relevant information over extended periods, making them particularly well-suited for tasks that involve temporal dependencies or require the model to remember past inputs, even if they occurred many time steps earlier.

LSTMs are highly effective for a wide range of applications, including time series forecasting, natural language processing (NLP), speech recognition, and video analysis, where understanding the order and context of input data is crucial. They are also capable of handling variable-length sequences, which is useful in real-world scenarios like sentence analysis in NLP, where the length of inputs can vary significantly. Moreover, LSTMs can be configured in bidirectional networks, allowing them to capture context from both past and future inputs, further enhancing their ability to understand complex sequences.

Training LSTM networks involves a method called Backpropagation Through Time (BPTT), which is a version of backpropagation tailored for sequence data. This technique ensures that the weights are updated properly across time steps, allowing the network to learn from both short-term and long-term patterns effectively. While LSTMs are computationally more intensive than simpler models, their ability to model long-range dependencies and capture intricate temporal relationships makes them a powerful tool for sequential data processing in machine learning.

3.5.1 Visualization of LSTM

The comprehension of algorithms and their composite characteristics is significantly enhanced through the visualization of datasets and the analysis of their outcomes.

3.5.2 Correlation Matrix

This correlation matrix visualizes the relationships between various parameters, including currents, temperatures, and tool current measurements. Strong correlations are highlighted in deep red, while weak or negative correlations are shown in blue. Notably, there's a high positive correlation between certain current and temperature pairs, such as `Current_J4` and `Temperature_J4`, and `Current_J5` and `Temperature_J5`, as indicated by values close to 1. Conversely, there are negative correlations, such as between `Current_J4` and `Current_J0`, indicating an inverse relationship. This matrix helps in identifying which parameters are closely related, which could be useful for further analysis or predictive modelling.

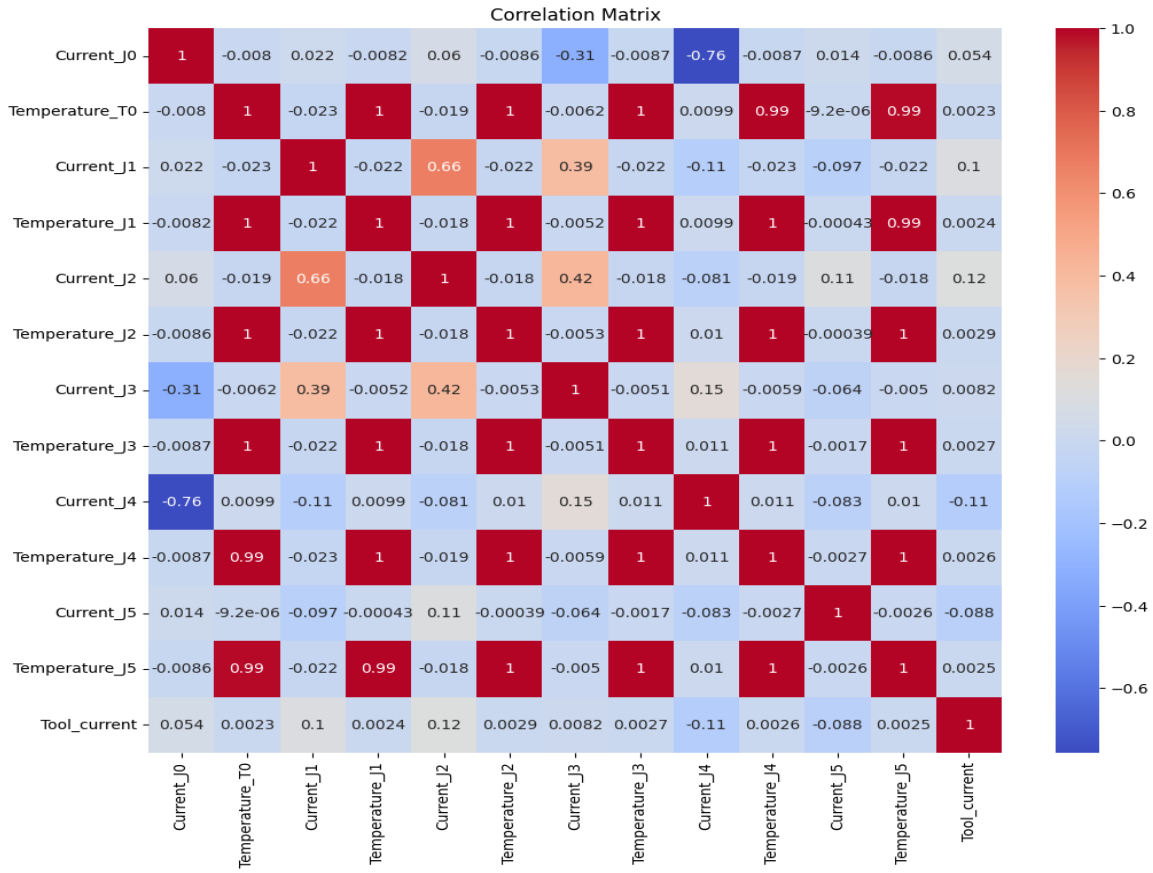

Figure 3.9 Correlation Matrix

3.5.3 Time Series Decomposition of Tool Current

This image displays a time series decomposition of the `Tool_current` parameter, broken down into four components: the observed data, trend, seasonal, and residual. The first plot shows the overall `Tool_current` time series, revealing fluctuations over time. The second plot, the trend component, indicates a slow-moving trend capturing the general direction of `Tool_current` values, which shows some peaks and troughs. The third plot illustrates the seasonal component, highlighting repeating patterns or cycles within the data. The final plot shows the residuals, which are the remaining variations after removing the trend and seasonal components, capturing any irregular or random fluctuations in the data. This decomposition aids in understanding the underlying structure and patterns within the `Tool_current` data.

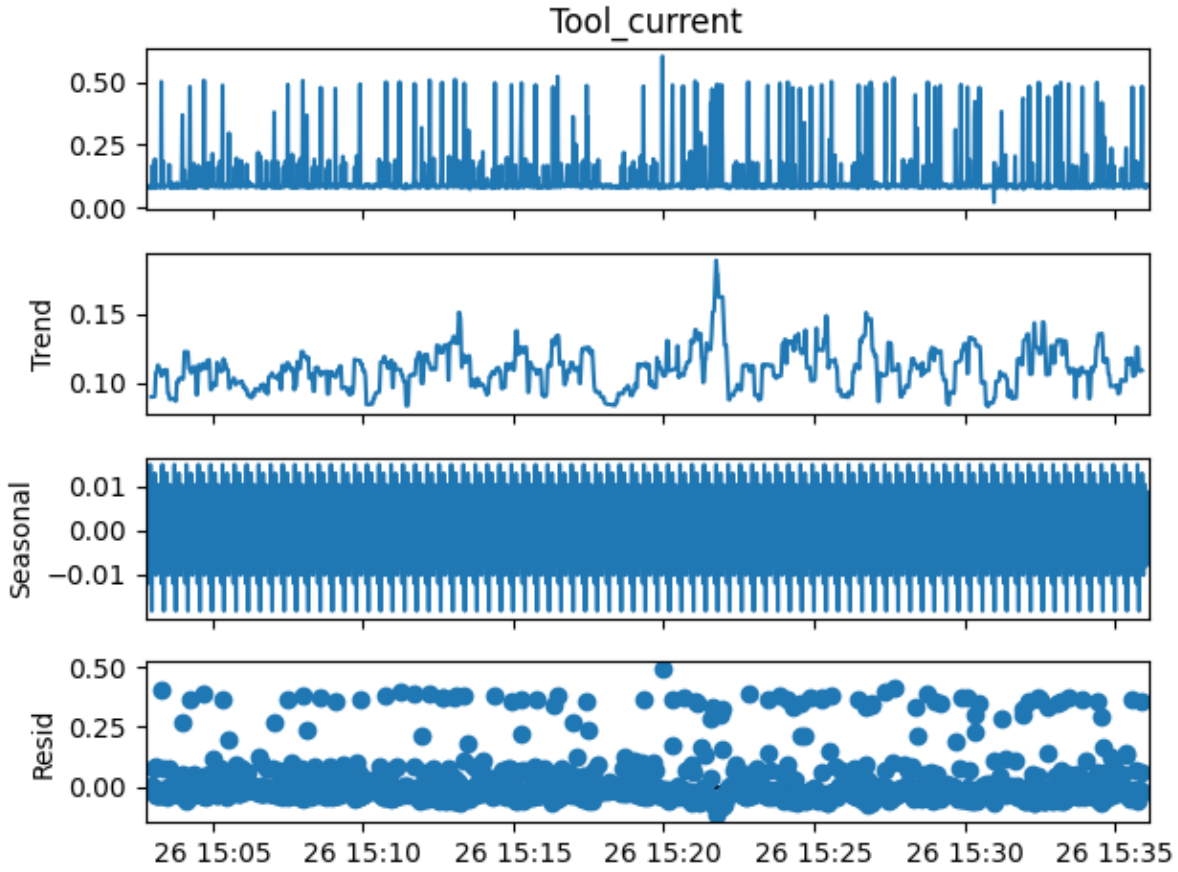

Figure 3.10 Time Series Decomposition of Tool Current

3.5.4 Loss Per Epoch

The provided plot illustrates the loss values over 50 training epochs for a model. Initially, the loss starts at approximately 0.0245 and decreases consistently, showing a general downward trend with minor fluctuations. By the end of the 50 epochs, the loss reaches around 0.0210. This gradual reduction in loss indicates that the model is learning and improving its performance over time. The slight fluctuations suggest occasional adjustments during training, but the overall trend demonstrates convergence towards lower loss values, implying successful training.

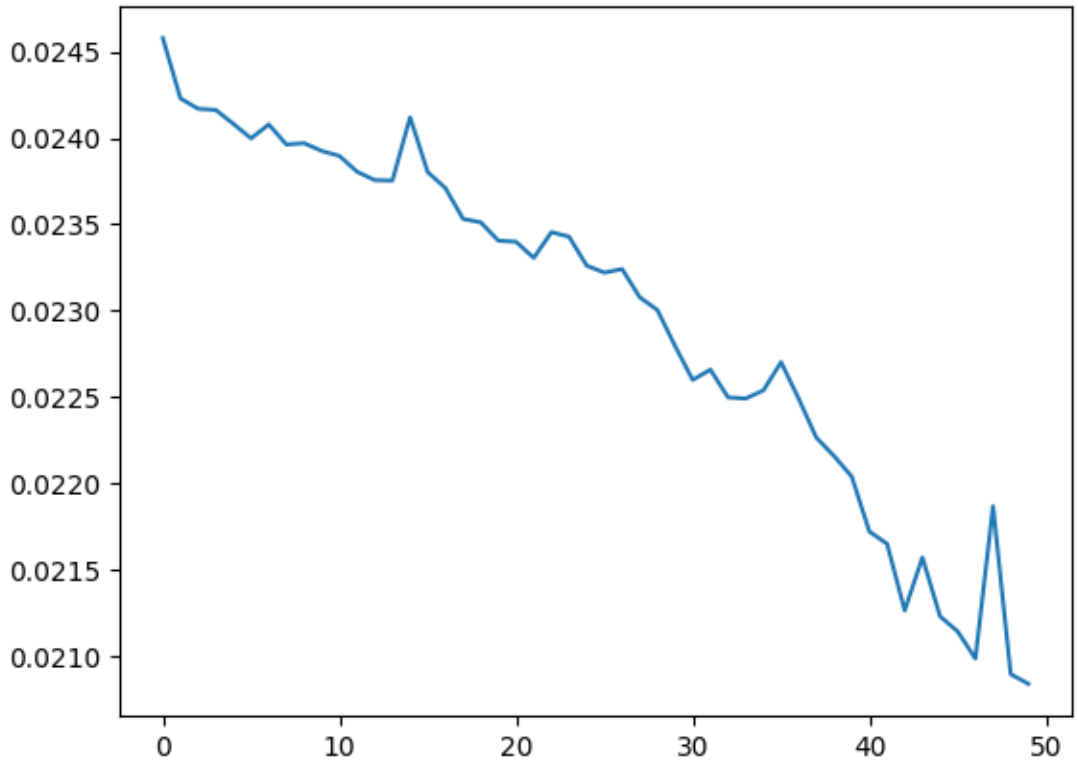

Figure 3.11 Loss Per Epoch

3.6 K-mean clusters on UR3

K-Means Clustering is a popular unsupervised learning algorithm that groups data points into clusters based on their similarities. It works by selecting a set number of clusters (K), assigning each point to the nearest centroid, and updating the centroids based on the average position of the points in each cluster. This process repeats until the centroids stabilize. The goal is to minimize the distance between points and their centroids. While effective, it requires pre-selecting the number of clusters. K-Means is widely used in tasks like customer segmentation and image compression.

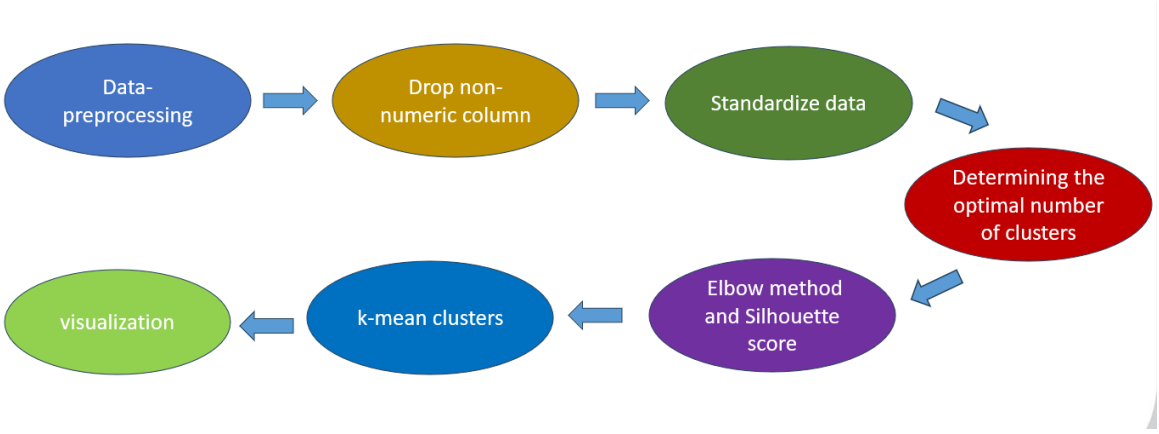

Figure 3.12 Steps of K-mean clusters

3.6.1 Elbow Method for Determining Optimal Number of Clusters

This plot shows the relationship between the number of clusters (k) and the inertia (sum of squared distances) for a clustering algorithm. The "elbow" in the graph, where the rate of decrease slows significantly, suggests the optimal number of clusters. Here, the elbow appears around k=4, indicating that 4 clusters may be optimal for minimizing inertia while preventing overfitting.

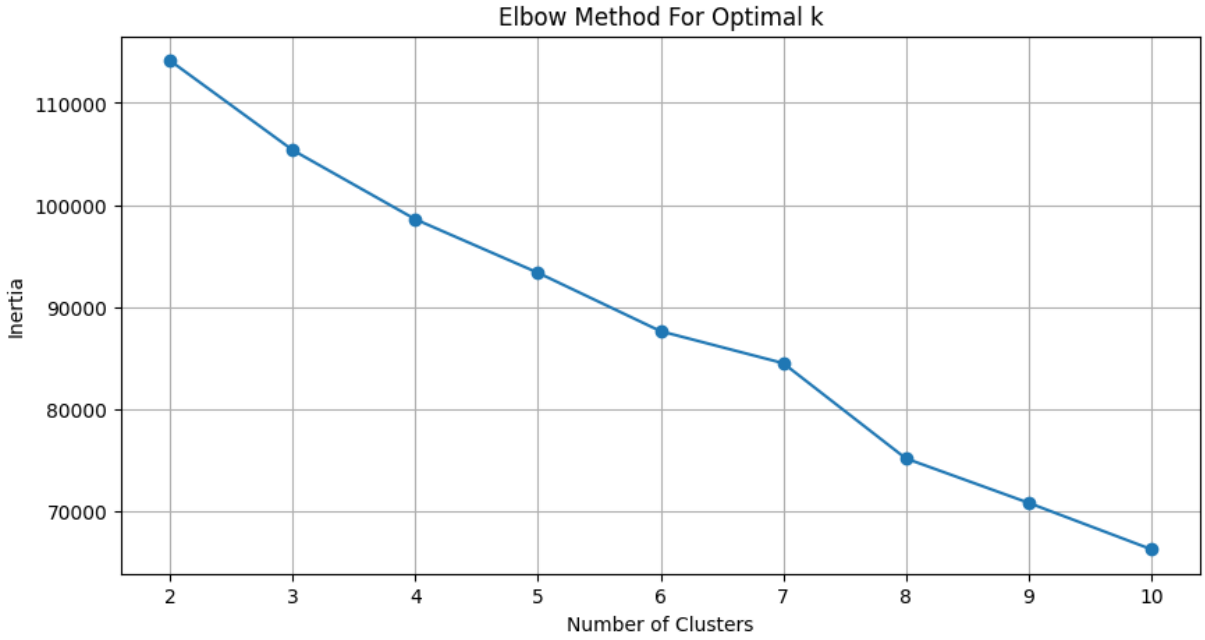

Figure 3.13 Elbow Method for Determining Optimal Number of Clusters

3.6.2 Silhouette Score for Determining Optimal Number of Clusters

This plot illustrates the relationship between the number of clusters (k) and the silhouette score, which measures how well each data point fits within its assigned cluster. Higher silhouette scores indicate better-defined clusters. In this graph, the peak silhouette score is at k=6, suggesting that 6 clusters provide the best balance between cohesion and separation of clusters.

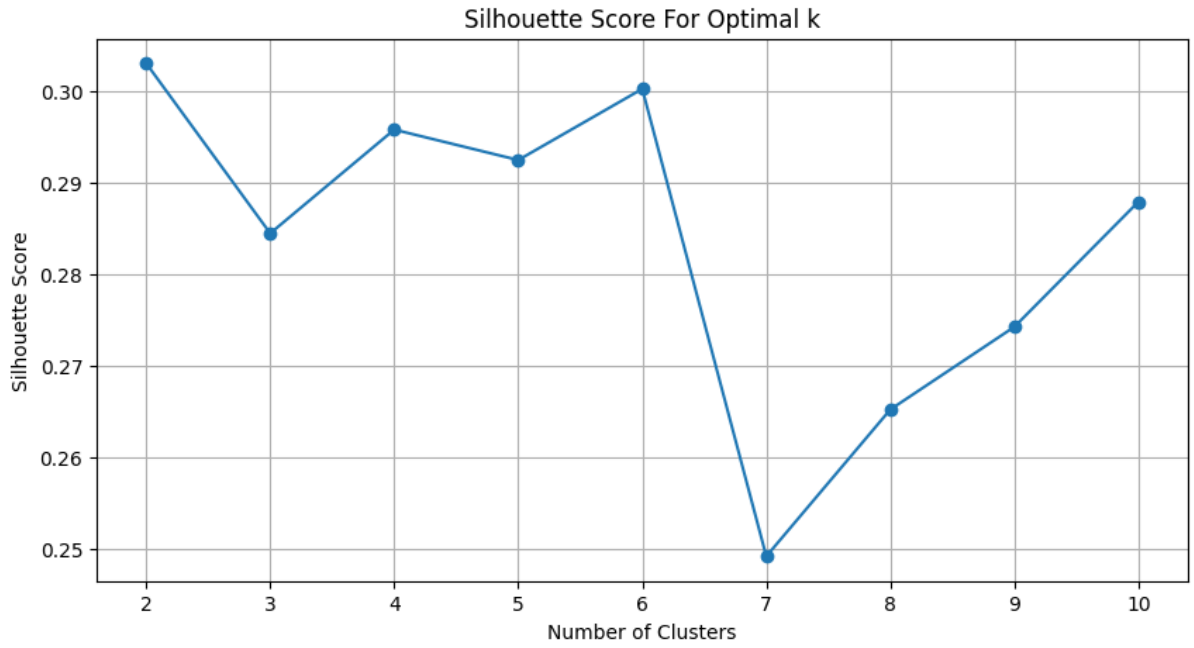

Figure 3.14 Silhouette Score for Determining Optimal Number of Clusters

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Experimental Results & Analysis

In a recent classification assignment, the performance metrics across a range of machine learning models highlighted the substantial advantages of employing hybrid approaches. Notably, the combination of XGBoost and LightGBM emerged as a top performer, achieving an impressive F1-score of 0.613 alongside a high accuracy of 0.977. This outcome underscores the efficacy of integrating two powerful boosting algorithms, which effectively leverage the strengths of each to enhance overall model performance. Similarly, a hybrid model combining Random Forest with LightGBM delivered robust results, with an F1-score of 0.611 and the same high accuracy, demonstrating the strength of blending bagging and boosting techniques.

In contrast, standalone models like Random Forest, although they showcased a strong precision of 0.850, revealed a notable trade-off with comparatively lower recall rates. This imbalance points to the challenges of achieving optimized performance metrics within isolated models, as they often excel in specific areas while underperforming in others. For instance, XGBoost, while maintaining a competitive accuracy of 0.974, demonstrated a lower recall, which may affect its ability to capture all relevant instances in more complex tasks. Simpler algorithms such as AdaBoost and Extra Trees, on the other hand, yielded more modest results in both precision and recall metrics. However, introducing LightGBM into these models resulted in noticeable performance improvements, highlighting the adaptive benefits of hybridization.

These findings collectively emphasize that hybrid models—leveraging multiple algorithms and blending their unique capabilities—are well-suited for achieving balanced and robust performance, especially in complex classification tasks. By integrating diverse methods, these hybrids are able to address the limitations of individual models, ultimately creating more versatile and accurate classifiers that can handle a range of scenarios with greater efficacy.

Table 4.1 Result Analysis on Classification

Models	**Precision**	**Recall**	**F1-Score**	**Accuracy**
XGboost	0.675	0.51923	0.586	0.974
LightGBM	0.727	0.461	0.564	0.975
Extra Trees Classifier	0.681	0.288	0.405	0.970
Random Forest Classifier	0.850	0.326	0.472	0.974
AdaBoost Classifier	0.571429	0.384615	0.459770	0.968
Hybrid (Random Forest + LightGBM)	0.787	0.500	0.611	0.974
Hybrid (XGBoost + LightGBM)	0.750	0.519	0.613	0.976
Hybrid (Extra Trees + LightGBM)	**0.758**	**0.423**	**0.543**	**0.977**

4.2 Result Graph

4.2.1 Single Model Performance

The bar chart displays performance metrics for various classifiers, including XGBoost, LightGBM, Extra Trees, AdaBoost, and Random Forest. The metrics shown are Accuracy, Recall, F1 Score, and Precision. All classifiers achieve high accuracy, with LightGBM slightly leading at 0.975. However, precision varies significantly, with Random Forest showing the highest precision at 0.850. F1 Scores range from 0.327 for Random Forest to 0.675 for LightGBM, while recall values are relatively lower across the board, especially for Extra Trees and AdaBoost. The chart highlights the trade-offs between precision and recall among these classifiers.

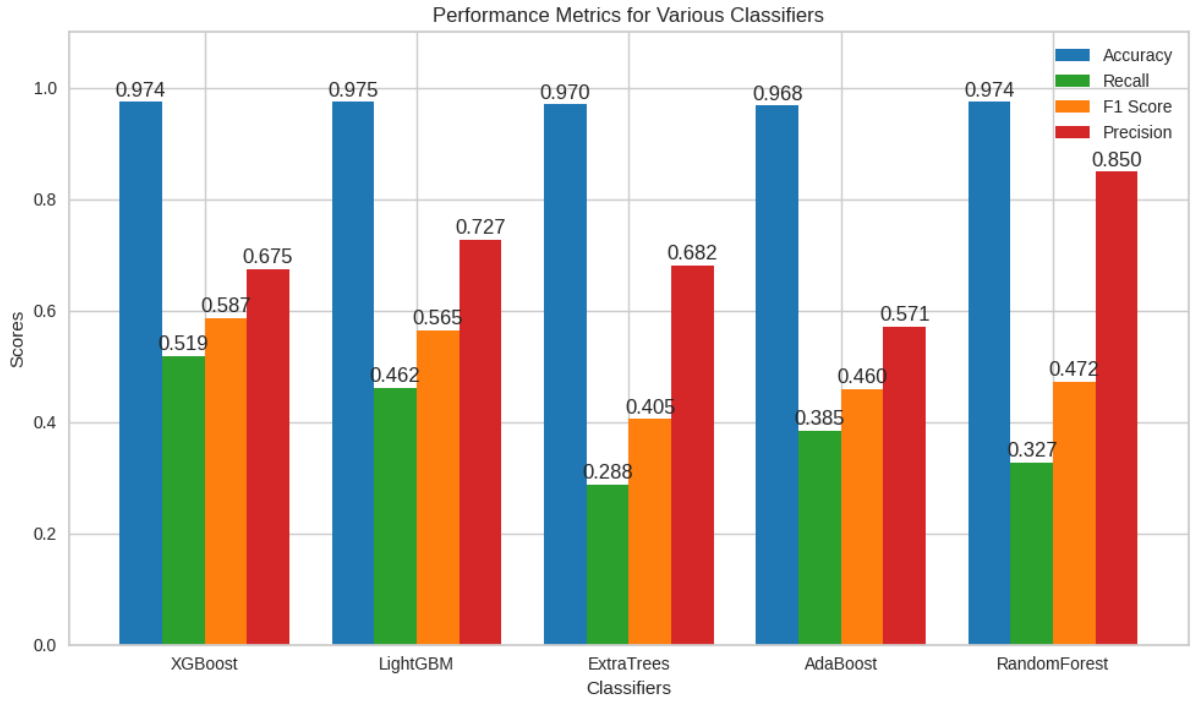

Figure 4.1 Single Model Performance

4.2.2 Hybrid Model Performance

The extended bar chart illustrates performance metrics for a variety of classifiers, including single models like XGBoost, LightGBM, Extra Trees, AdaBoost, and Random Forest, as well as hybrid combinations such as Random Forest + LightGBM, XGBoost + LightGBM, and Extra Trees + LightGBM. Each classifier's performance is assessed based on Accuracy, Recall, F1 Score, and Precision. The hybrid models generally show enhanced performance, with the Random Forest + LightGBM hybrid achieving the highest accuracy at 0.978 and an F1 Score of 0.612, indicating a well-balanced result. The XGBoost + LightGBM combination follows closely with an accuracy of 0.977 and a slightly higher F1 Score of 0.614, showcasing the benefits of combining boosting algorithms. While standalone models like LightGBM and Random Forest demonstrate strong precision scores of 0.675 and 0.850 respectively, they reveal trade-offs in recall and F1 scores. Overall, the chart highlights how hybrid approaches often yield better-balanced results across multiple metrics, making them suitable for complex classification tasks that require both precision and recall.

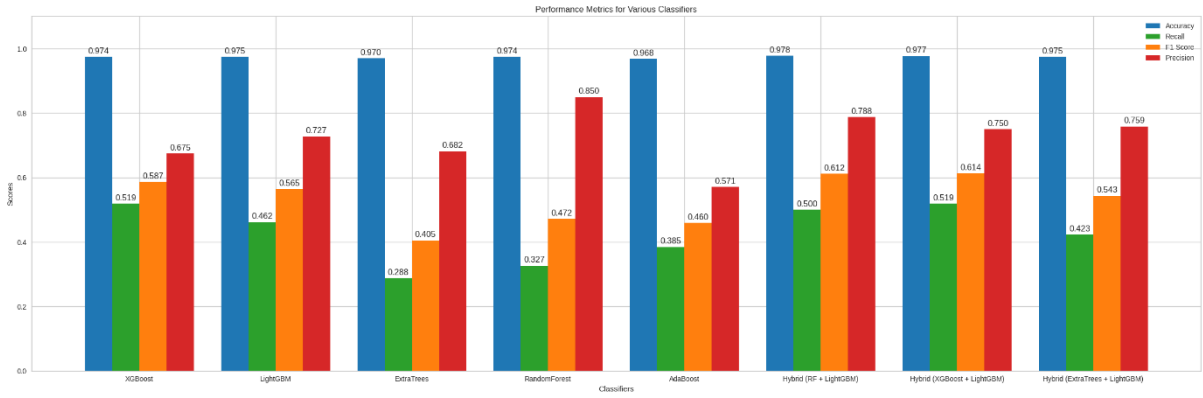

Figure 4.2 Hybrid Model performance

4.2.3 Final outcome of Classification

This bar chart ranks the accuracy of various classifiers in ascending order. The models compared include AdaBoost, Extra Trees, XGBoost, Random Forest, LightGBM, and three hybrid combinations: Extra Trees + LightGBM, XGBoost + LightGBM, and Random Forest + LightGBM.

AdaBoost has the lowest accuracy at 0.9683, while the hybrid model of Random Forest and LightGBM achieves the highest accuracy at 0.9777. Notably, hybrid models generally outperform the individual classifiers, with all hybrid combinations achieving accuracy levels of 0.9750 or higher. This visualization highlights how blending algorithms, particularly those combining bagging and boosting techniques, can enhance accuracy in classification tasks.

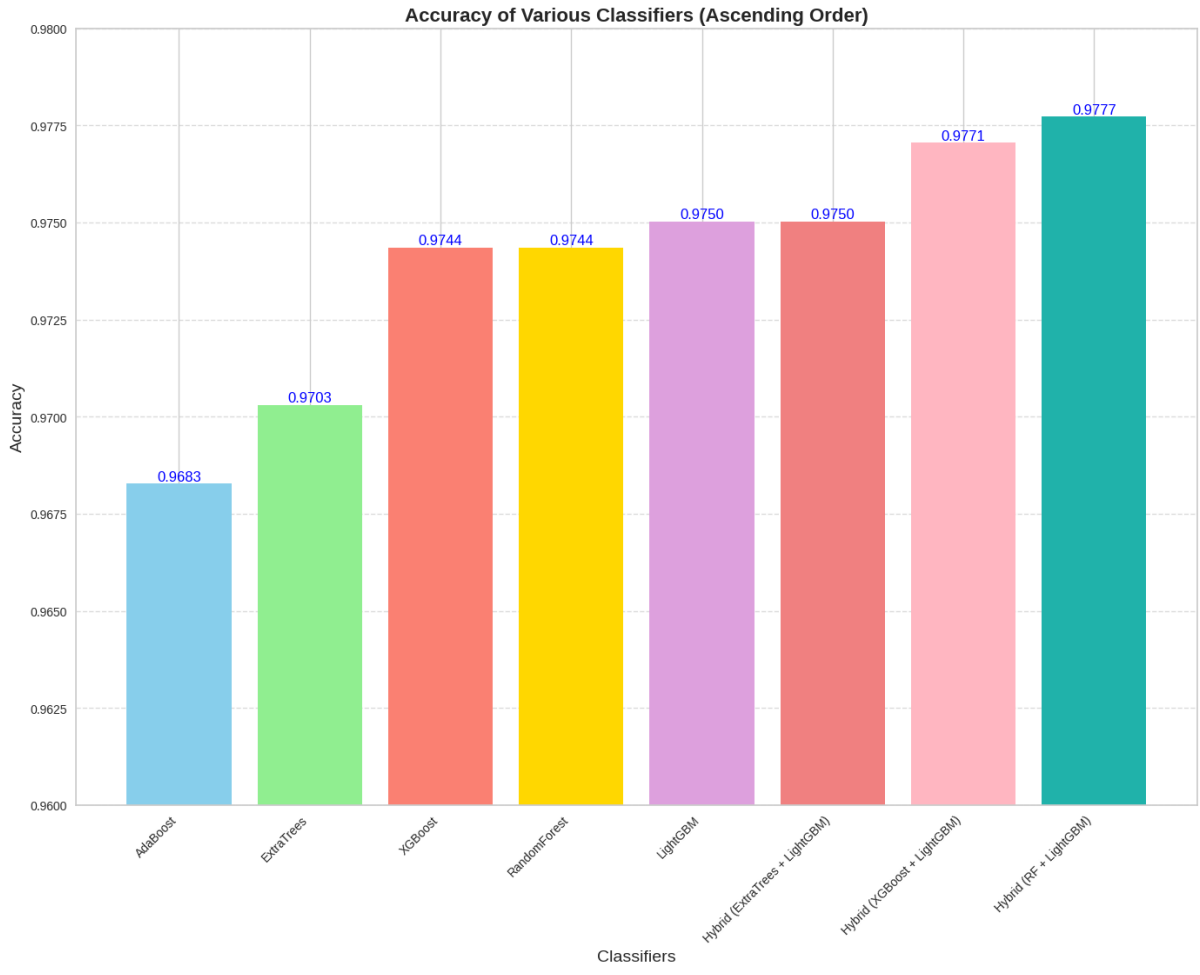

Figure 4.3 Final outcome of Classification

4.2.4 Final outcome of ARIMA

The Augmented Dickey-Fuller (ADF) test confirms that the time series is stationary, with an *ADF statistic of -14.4327* and a *p-value of 7.5851e-27*, suggesting stability in its statistical properties over time. The optimal ARIMA model parameters are identified as *(2, 1, 2)*, where $p = 2$ indicates two lagged autoregressive terms, $d = 1$ represents first-order differencing, and $q = 2$ involves two lagged terms in the moving average. Performance metrics for this ARIMA model include a *Mean Absolute Error (MAE) of 0.0393*, a *Mean Squared Error (MSE) of 0.0055*, and a *Root Mean Squared Error (RMSE) of 0.0741*, highlighting relatively low prediction errors. However, the R^2 value of -0.0049 indicates that the model explains almost none of the variance in the data, suggesting potential for further refinement or alternative modelling approaches.

Table 4.2 ARIMA Model Performance Metrics

Metric	Value
ADF Statistic	-14.4327
p-value (ADF Test)	7.5851e-27
MAE (Mean Absolute Error)	0.0393
MSE (Mean Squared Error)	0.0055
RMSE (Root Mean Squared Error)	0.0741
R^2 (Coefficient of Determination)	0.0049

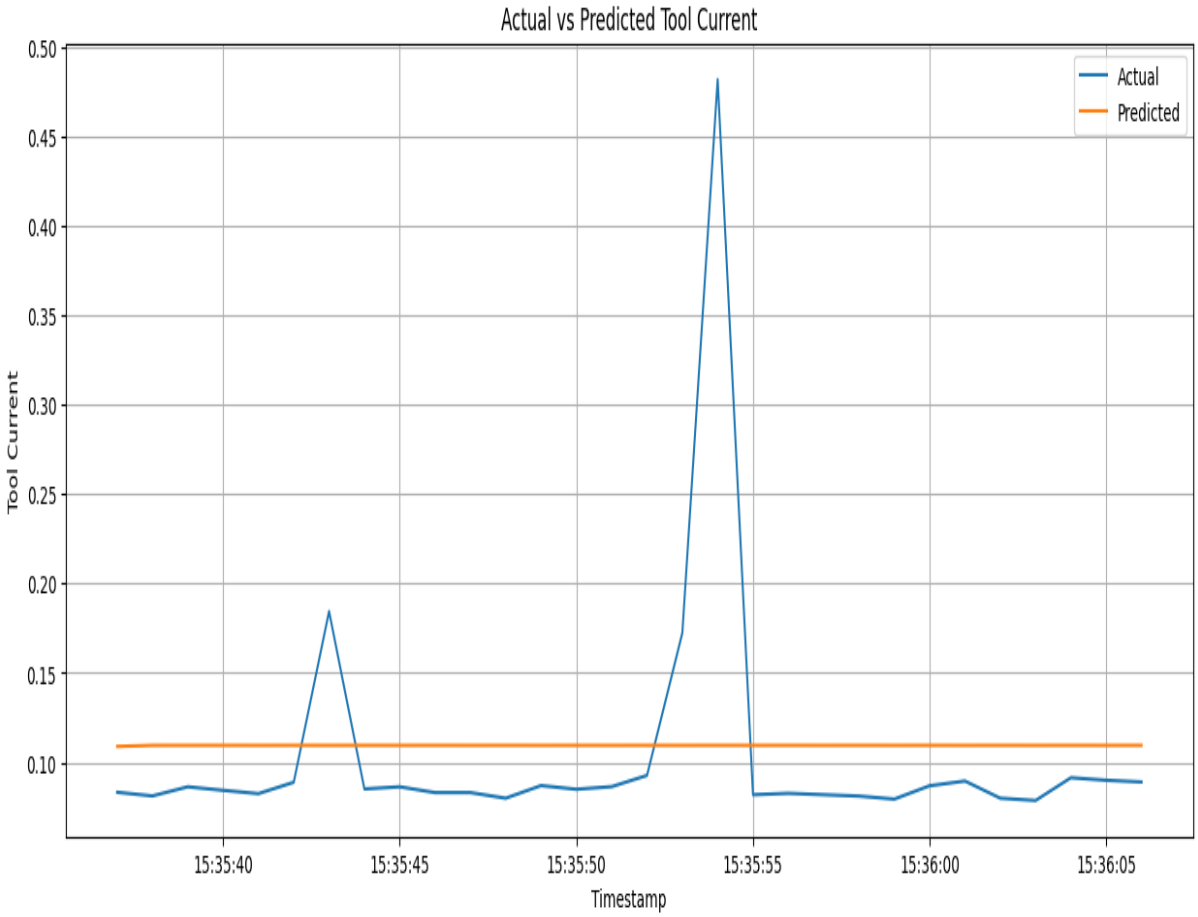

Figure 4.4 Final outcome of ARIMA

4.2.5 The Architecture Summary of a Sequential Neural Network Model with Two Layers

The model architecture shown in the image consists of a sequential neural network with two layers. The first layer is an LSTM (Long Short-Term Memory) layer, which outputs 100 units and has 40,800 trainable parameters, including weights and biases. This is followed by a dense (fully connected) layer that produces a single output, with 101 trainable parameters. The total number of parameters in the model is 40,901, all of which are trainable, as there are no non-trainable parameters. This structure suggests the model is designed for sequence-based tasks, such as time series prediction or classification.

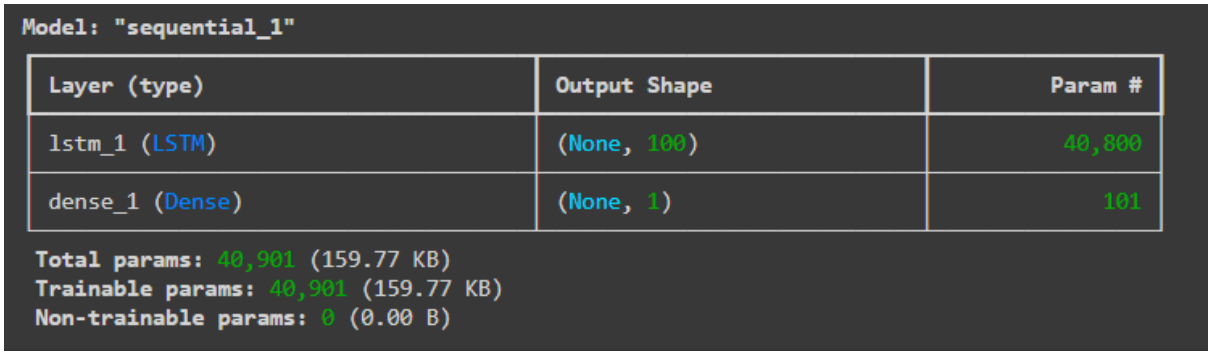

Figure 4.5 The Architecture Summary of a Sequential NN Model with Two Layers

4.2.5.1 Final outcome of LSTM

This plot compares the actual tool current values (blue line) with the LSTM model's predictions (orange line) over a specific time period. While the general trend of the tool current is captured by the model, the prediction line smooths out the higher spikes observed in the actual values, indicating a potential limitation in capturing sudden fluctuations. This suggests that the model may benefit from adjustments to improve accuracy during periods of sharp changes in tool current. *rmse=0.08992910208161961*

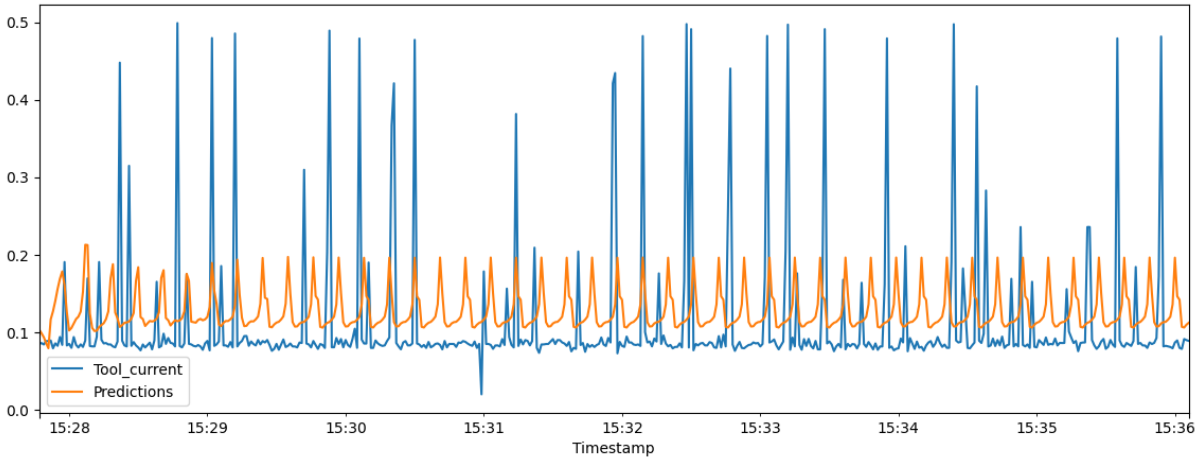

Figure 4.6 Final outcome of LSTM

4.2.6 Silhouette Plot for K-Means Clustering (k=2)

This silhouette plot illustrates the quality of clustering for a K-means algorithm with 2 clusters. Each cluster's silhouette coefficients are represented, with cluster 1 in blue and cluster 2 in orange. The red dashed line indicates the average silhouette score across all clusters. The silhouette coefficient measures how well each data point is assigned to its cluster, with values closer to 1 indicating better fit. This plot helps evaluate the cohesion and separation of the clusters, with higher scores reflecting more distinct and well-defined clusters.

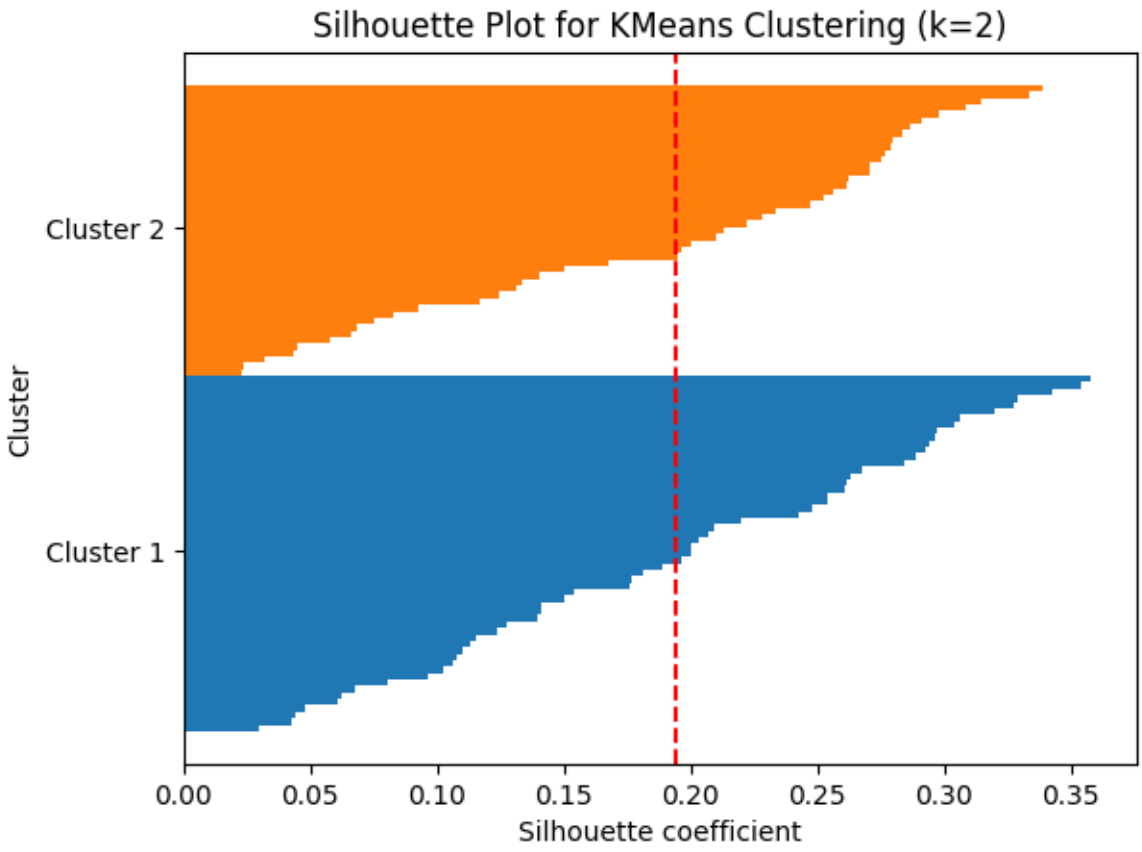

Figure 4.7 Silhouette Plot for K-Means Clustering (k=2)

4.2.7 K-Means Clustering Visualization with 3 Clusters

This scatter plot visualizes the results of K-means clustering with 3 clusters. The data points are projected onto two principal components, allowing for a two-dimensional representation. Each colour represents a different cluster, as indicated by the colour bar to the right. The clusters appear well-separated, indicating that the K-means algorithm has successfully identified distinct groupings within the data based on the specified number of clusters.

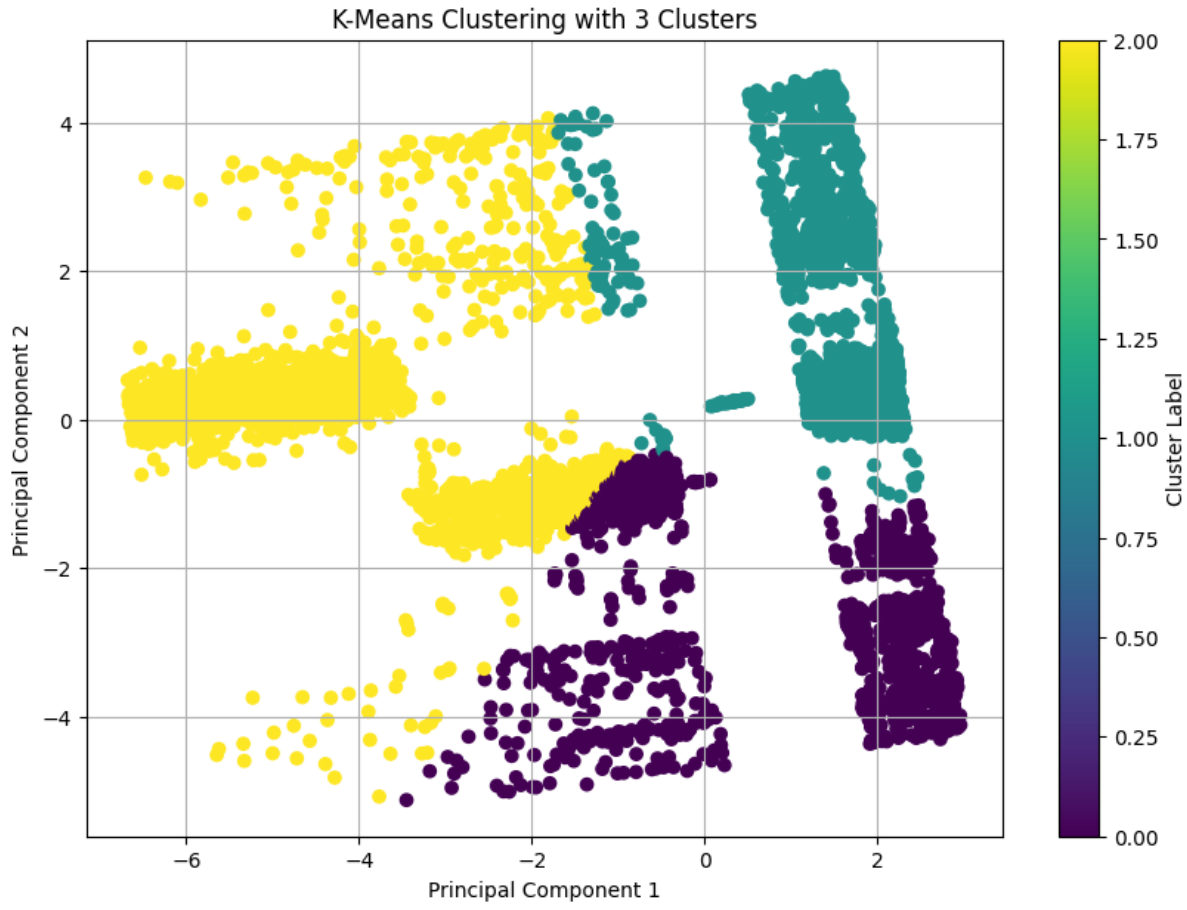

Figure 4.8 K-Means Clustering Visualization with 3 Clusters

4.2.8 3D Visualization of Clusters using PCA

This 3D scatter plot presents a visualization of clusters using Principal Component Analysis (PCA). Data points are plotted across three principal components, revealing the structure of clusters in a three-dimensional space. Different colours correspond to distinct clusters, as shown by the colour gradient on the right. This visualization aids in understanding the spatial distribution and separation of clusters, providing insights into the dimensionality and variance within the dataset. The clusters appear well-separated, suggesting that the data exhibits distinguishable groupings in this reduced-dimensional space.

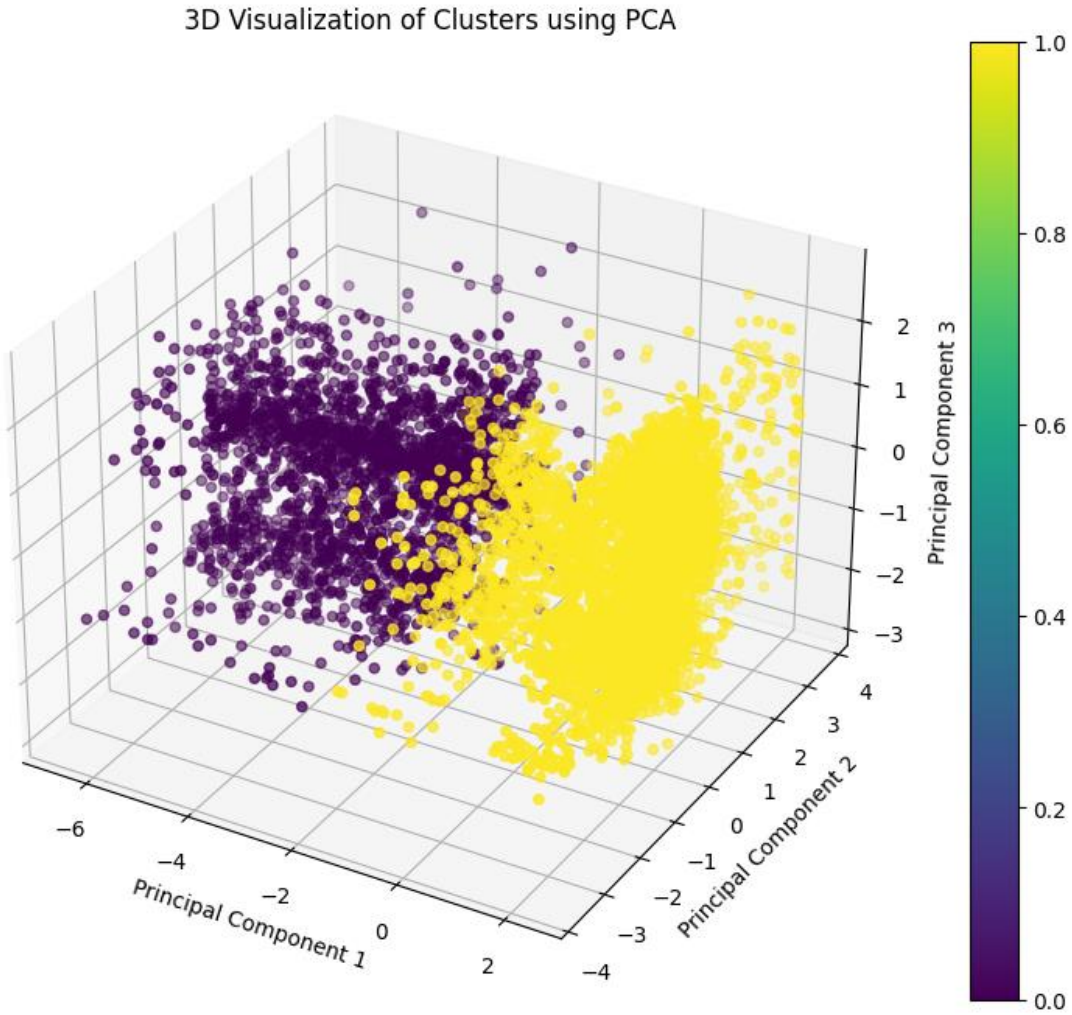

Figure 4.9 3D Visualization of Clusters using PCA

4.2.9 3D Visualization of Clusters using PCA 1

The image shows a 3D scatter plot visualizing clusters formed using Principal Component Analysis (PCA). The points are distributed across three principal components (PC1, PC2, and PC3), with colours representing the first principal component (PC1). Blue tones indicate lower values, transitioning to red for higher values of PC1. The clusters are well-separated, suggesting that PCA effectively reduces the dimensionality of the data while preserving distinct groupings. The use of colour adds an additional layer of interpretability, highlighting the variance explained by PC1 across the data points.

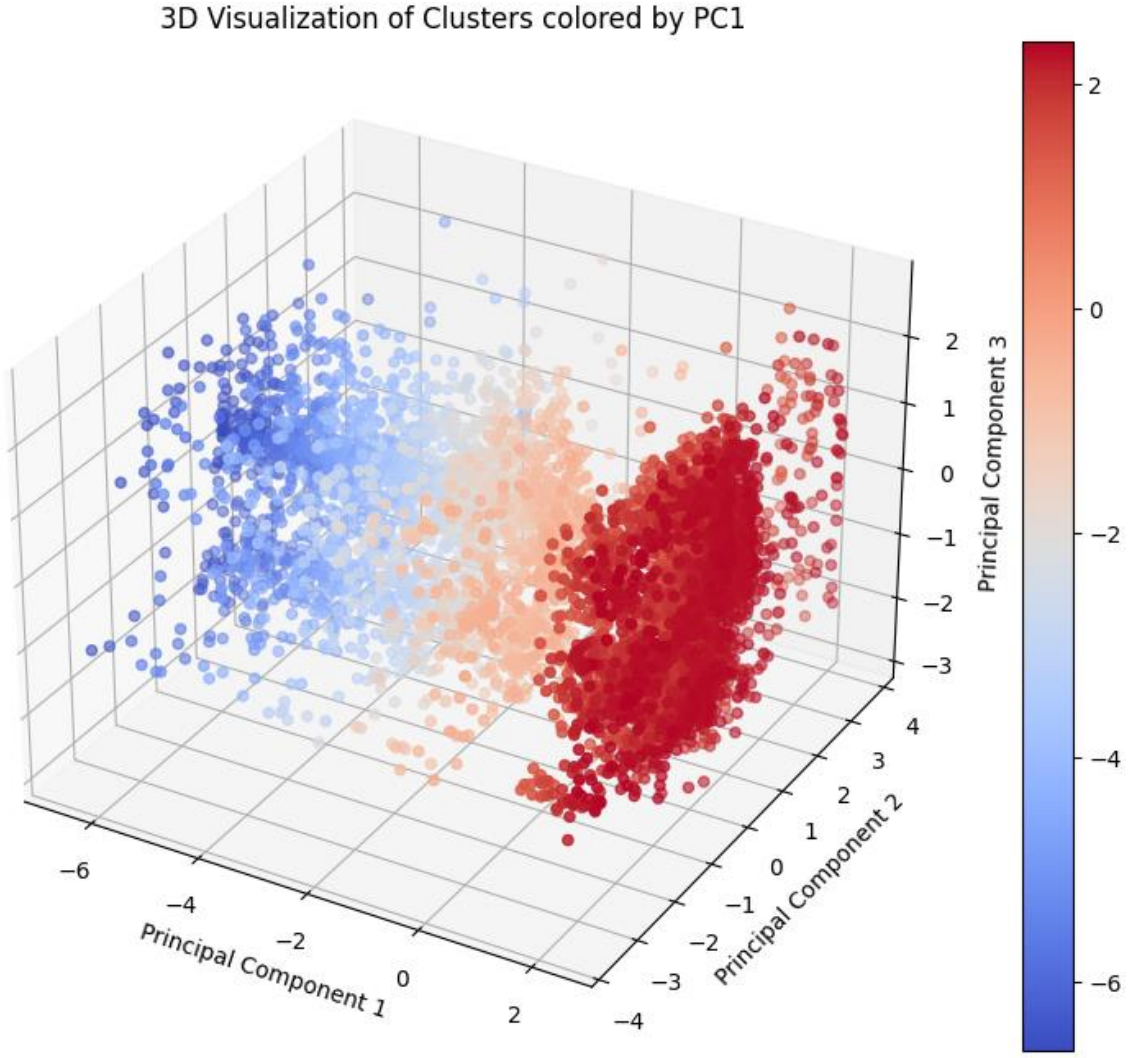

Figure 4.10 3D Visualization of Clusters using PCA 1

CHAPTER 5

CONCLUSION AND FUTURE DIRECTION

5.1 Summary of the Study

This project focused on evaluating the performance of various machine learning algorithms, both as standalone models and hybrid combinations, for classification tasks related to fault detection and prediction on a URC cobot. The primary objective was to assess how hybrid models, which blend different algorithms, compare to individual models in terms of performance metrics such as Accuracy, Precision, Recall, and F1-Score. Additionally, time-series forecasting was explored using ARIMA and LSTM models to analyze their efficacy and suitability for predicting tool current values over time. In the classification task, a diverse range of models, including XGBoost, LightGBM, Extra Trees, AdaBoost, and Random Forest, as well as several hybrid combinations, were evaluated. The hybrid models combining XGBoost with LightGBM and Random Forest with LightGBM emerged as top performers, achieving high F1-Scores of 0.613 and 0.611, respectively, alongside impressive accuracies of 0.977. These hybrids leveraged the complementary strengths of boosting and bagging techniques to achieve robust, well-balanced performance across metrics. In contrast, standalone models like Random Forest and LightGBM exhibited high precision but struggled with recall, indicating limitations in capturing all relevant instances, while simpler algorithms like AdaBoost and Extra Trees showed modest F1-Scores and recall rates. However, adding LightGBM to these models noticeably improved their performance, demonstrating the adaptive benefits of hybridization. For time-series analysis, ARIMA and LSTM models were employed to predict tool current values, with the ARIMA model showing low error metrics but a negative R^2 value, suggesting limited explanatory power. The LSTM model captured the general trend effectively but struggled with sudden fluctuations, indicating potential for improvement in capturing sharp changes. These findings collectively underscore the advantages of hybrid models for complex classification tasks on URC cobots, where the integration of diverse algorithms can create more versatile and accurate classifiers capable of addressing specific limitations of individual models.

5.2 Conclusion

The remarkable potential of machine learning (ML) techniques in revolutionizing the area of hypothyroidism diagnosis is highlighted in the study's conclusion. We have achieved an impressive 99.6% diagnostic accuracy rate by using an all-inclusive ensemble method that combines various algorithms, such as RF, AB, NBC and DT. This achievement emphasizes the importance of combining different approaches rather than relying solely on one and the crucial part hyperparameter optimization performs in optimizing model performance. The findings of our study pave the way for the use of state-of-the-art data-driven methods in endocrinology, which may significantly increase the precision of hypothyroidism diagnosis. Furthermore, the study advocates for the application of precision medicine techniques and the continued exploration of hybrid modelling approaches with the goal of improving patient results in the treatment of hypothyroidism. Our study offers valuable insights and lays a solid foundation for future research aiming at harnessing the potential of machine learning in medical diagnostics, while the world healthcare landscape increasingly shifts to more scalable and effective diagnostic models.

5.3 Implication for Further Study

The study highlights the potential of hybrid models for improving fault detection and prediction on URC cobots, suggesting several avenues for further research. Future studies could explore additional algorithm combinations, advanced time-series forecasting techniques like GRUs or ARIMA-LSTM hybrids, and feature engineering to enhance accuracy. Real-time fault detection with adaptive learning could also improve model robustness, while transfer learning could test the generalizability of these models across different cobots. Additionally, incorporating environmental and operational data may enable the models to handle diverse conditions more effectively. These areas of inquiry could lead to more accurate and adaptable fault prediction systems for industrial robotics.

References

[1] S. El Zaatari, M. Marei, W. Li, and Z. Usman, “Cobot Programming for Collaborative Industrial Tasks: An Overview,” *Robotics and Autonomous Systems*, vol. 116, pp. 162-180, 2019, DOI: 10.1016/j.robot.2019.03.003.

[2] Y. S. Park, J. W. Lee, and D. Y. Yoo, “Programmable Motion-Fault Detection for a Collaborative Robot,” *IEEE Access*, vol. 8, pp. 73812-73822, 2020, DOI: 10.1109/ACCESS.2020.2990276.

[3] Y. S. Park, J. W. Lee, and D. Y. Yoo, “Predictive Maintenance Techniques for Collaborative Robots,” *IEEE Access*, vol. 8, pp. 73812-73822, 2020, DOI: 10.1109/ACCESS.2020.2990276.

[4] Universal Robots. (2023). UR3e Ultra-lightweight, compact cobot. Retrieved from https://www.universal-robots.com/products/ur3-robot/

[5] J. Saenz et al., “An Online Toolkit for Applications Featuring Collaborative Robots Across Different Domains,” Journal of Robotics, vol. 53, no.4, pp. 657-667, 2024.

[6] S. G. Graabæk, E. V. Ancker, A. R. Fugl, and A. L. Christensen, “An Experimental Comparison of Anomaly Detection Methods for Collaborative Robot Manipulators,” IEEE Transactions on Industrial Electronics, vol. 67, no. 9, pp. 7929-7939, Sept. 2020, doi: 10.1109/ACCESS.2023.3289068.

[7] A. H. Sabry, F. H. Nordin, A. H. Sabry, and M. Z. A. Abidin, “Fault Detection and Diagnosis of Industrial Robot Based on Power Consumption Modeling,” IEEE Transactions on Industrial Electronics, vol. 67, no. 9, pp. 7929-7940, Jun. 2023, doi: 10.1109/ACCESS.2023.3289068.

[8] M. Tyrovolas, C. Stylios, K. Aliev, and D. Antonelli, "Leveraging information flow-based fuzzy cognitive maps for interpretable fault diagnosis in industrial robotics," in Technological Innovation for Human-Centric Systems: DoCEIS 2024, L. M. Camarinha-Matos and F. Ferrada, Eds., IFIP Advances in Information and Communication Technology, vol. 716, Cham: Springer, 2024, pp. 66-76. doi: 10.1007/978-3-031-63851-0_6.

[9] H. Bai, J.-H. Gao, W. Li, K. Wang, and M.-F. Guo, "Detection of High-Impedance Fault in Distribution Networks Using Frequency-Band Energy Curve," *IEEE Sensors Journal*, 2023

[10] J. Talukdar, A. Vyas, and K. Chakrabarty, "Detection of Voltage Droop-induced Timing Fault Attacks due to Hardware Trojans," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 2024.

[11] J. Li, D. Wang, X. Wu, K. Xu, and X. Liu, "Vibration prediction of the robotic arm based on elastic joint dynamics modeling," Sensors, vol. 22, no. 16, p. 6170, Aug. 2022. doi: 10.3390/s22166170.

[12] N. Anđelić, S. Baressi Šegota, M. Glučina, and I. Lorencin, "Classification of faults operation of a robotic manipulator using symbolic classifier," Applied Sciences, vol. 13, no. 3, p. 1962, 2023. doi: 10.3390/app13031962.

[13] Y. Cong, R. Chen, M. Bingtao, H. Liu, D. Hou, and C. Yang, "A comprehensive study of 3-D vision-based robot manipulation," IEEE Transactions on Cybernetics, vol. PP, no. 99, pp. 1-17, Sep. 2021. doi: 10.1109/TCYB.2021.3108165.

[14] Y. Cong, R. Chen, B. Ma, H. Liu, D. Hou, and C. Yang, "A comprehensive study of 3-D vision-based robot manipulation," IEEE Transactions on Cybernetics, vol. 53, no. 3, pp. 1682-1698, Mar. 2023. doi: 10.1109/TCYB.2021.3108165.

[15] S. Guo and W. Guo, "Process monitoring and fault prediction in multivariate time series using bag-of-words," IEEE Transactions on Automation Science and Engineering, vol. PP, no. 99, pp. 1-13, Oct. 2020. doi: 10.1109/TASE.2020.3026065.

[16] G. Chadha and A. Schwung, "Time series based fault detection in industrial processes using convolutional neural networks," in Proceedings of the IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society, Oct. 2019. doi: 10.1109/IECON.2019.8926924.

[17] S. Lu and S. Huang, "Segmentation of multivariate industrial time series data based on dynamic latent variable predictability," IEEE Access, vol. PP, no. 1, pp. 1-1, Jun. 2020. doi: 10.1109/ACCESS.2020.3002257.

[18] M. A. Alobaidy, A. Alobaidy, J. Abdul-Jabbar, and S. Saeed, "Faults diagnosis in robot systems: A review," Al-Rafidain Engineering Journal (AREJ), vol. 25, pp. 164-175, Dec. 2020. doi: 10.33899/rengj.2020.127782.1051.

[19] J. Li, D. Wang, X. Wu, K. Xu, and X. Liu, "Vibration Prediction of the Robotic Arm Based on Elastic Joint Dynamics Modeling," 2022. [Online]. Available: https://doi.org/10.3390/s22166170.

[20] A. Moshrefi, H. H. Tawfik, M. Y. Elsayed, and F. Nabki, "Industrial Fault Detection Employing Meta Ensemble Model Based on Contact Sensor Ultrasonic Signal," 2024. [Online]. Available: https://doi.org/10.3390/s24072297

[21] L. Biddle and S. Fallah, "A Novel Fault Detection, Identification and Prediction Approach for Autonomous Vehicle Controllers Using SVM," *Automotive Innovation*, 2023.

[22] H. Eltayeb, A. Adam, J. K. Kimotho, and J. G. Njiri, "Multiple faults diagnosis for an industrial robot fuse quality test bench using deep-learning," 2023. [Online]. Available: https://doi.org/10.1016/j.rineng.2023.101007.

[23] A. H. Sabry, A. Bte, and U. Amirulddin, "A review on fault detection and diagnosis of industrial robots and multi-axis machines," 2024. [Online]. Available: https://doi.org/10.1016/j.rineng.2024.102397.

[24] Z. Zamanzadeh, G. Darban, G. I. Webb, and C. C. Aggarwal, "Deep Learning for Time Series Anomaly Detection: A Survey," arXiv:2211.05244v3 [cs.LG], May 2024. Available: https://arxiv.org/abs/2211.05244v3

[25] Z. Zamanzadeh, G. Darban, G. I. Webb, and C. C. Aggarwal, "Deep Learning for Time Series Anomaly Detection: A Survey," arXiv:2211.05244v3 [cs.LG], May 2024. Available: https://arxiv.org/abs/2211.05244v3

[26] S. G. Graabaek, E. V. Ancker, A. R. Fugl, and A. L. Christensen, "An Experimental Comparison of Anomaly Detection Methods for Collaborative Robot Manipulators," IEEE Access, 2023. [Online]. Available: https://doi.org/10.1109/ACCESS.2023.3289068

[27] Author(s), "Cobot Programming for Collaborative Industrial Tasks: An Overview," Robotics and Autonomous Systems, vol. XX, no. YY, pp. ZZ-ZZ, Month Year. DOI: 10.1016/j.robot.2019.03.003.

[28] D. Silvestre, C. Silvestre, and J. P. Hespanha, "Fault Detection and Isolation in Cyber-Physical Systems: A Set-Valued Observer Approach," in *Proceedings of the IEEE Conference on Cyber-Physical Systems*, Macau, China, 2023, pp. 1-6

[29] E. Uhlmann, J. Polte, and C. Geisert, "Condition Monitoring Concept for Industrial Robots," 2023

[30] G. Georgiev, F.-W. Bauer, R. Sindelar, D. González-Aguilera, P. Rodríguez-Gonzálvez, A. Alharbi, I. Petrunin, and D. Panagiotakopoulos, "Deep Learning Architecture for UAV Traffic-Density Prediction," Drones, vol. 7, no. 78, 2023.