T.C.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

PREDICTIVE MAINTENANCE FOR AUTOMATED PRODUCTION SYSTEMS

Abdurrahman BULUT

SUPERVISOR

Prof. Dr. İbrahim SOĞUKPINAR

JUNE 2023

GEBZE, KOCAELI

T.C.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

PREDICTIVE MAINTENANCE FOR AUTOMATED PRODUCTION SYSTEMS

Abdurrahman BULUT

SUPERVISOR

Prof. Dr. İbrahim SOĞUKPINAR

JUNE 2023

GEBZE, KOCAELI

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 23/06/2023 by the following jury.

JURY

Supervisor Name	Prof. Dr. İbrahim SOĞUKPINAR	
University	Gebze Technical University	
Faculty	Faculty of Engineering Department of Computer Engineering	

Supervisor Name	Dr. Gökhan Kaya	
University	Gebze Technical University	
Faculty	Faculty of Engineering Department of Computer Engineering	

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my advisor, Prof. Dr. İbrahim SOĞUKPINAR, for his guidance, support, and encouragement throughout the course of this research project. His expertise and experience in the field of machine learning have been invaluable in the development of my research. He provided me with invaluable feedback and suggestions which helped me to improve my work. I am grateful for the time he spent discussing my ideas and for the many helpful suggestions he provided. I am also grateful to the Kaggle community who shared the dataset that I used in my project.

Additionally, I would like to acknowledge the related works and literature in the field of predictive maintenance and machine learning, which provided valuable insights and a foundation for this research. Their contributions have laid the foundation for this research and have been a constant source of inspiration.

June 2023 Abdurrahman Bulut

LIST OF SYMBOLS AND ABBREVIATIONS

comp0	:	There is no problem in any of the components
comp1,comp2,comp3,comp4	:	There is a problem with the related component
SVM	:	Support Vector Machine
EDA	:	Exploratory Data Analysis
KNN	:	K-Nearest Neighbors
ANN	:	Artificial Neural Network

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ÖZET

Teknolojinin ilerlemesi, makinelerin yaygın olarak kullanılmasını sağlamış, insan gücüne olan bağımlılığı azaltmayı ve üretimde daha büyük verimlilik elde etmeyi amaçlamıştır. Özellikle Endüstri 4.0'ın ortaya çıkmasıyla birlikte otomatik üretim makinelerinin kullanımı hızla artmıştır. Ancak, bu makineler arızalara yatkın olup, sahipleri için maddi ve zamansal zorluklara yol açabilir. Bu makinelerde yerleştirilen sensörlerin kullanılmasıyla önemli olay kayıtları elde edilebilir. Bu olay kayıtları, geçmiş sorun kayıtlarıyla birleştirilerek değerli ve kapsamlı bir veri havuzu oluşturulur. Bazı büyük şirketler, bu veriyi kullanarak makinelerin arızalarının önlenmesini amaçlayan projelerle ilgilenmektedir.

Bu projede, birbirine bağlı ve özellikler açısından zengin veri setleri kullanılarak sınıflandırmalar yapılmıştır. Bir eğitim seti oluşturulmuş ve en doğru çözümü belirlemek için çeşitli sınıflandırma ve öğrenme algoritmaları araştırılmıştır. Ayrıca, kullanıcıların makine olay kayıtlarını yükleyebilecekleri ve analiz edebilecekleri sezgisel bir arayüz tasarlanmıştır. Bu arayüz aracılığıyla, kullanıcılar sağlanan olay kayıtlarına dayanarak makinelerinde olası hasar durumlarıyla ilgili bilgi edinirler.

Gelişmiş veri analizi teknikleri ve makine öğrenme algoritmalarının kullanılmasıyla bu proje, makine arızalarının tahmin edilmesine ve önlenmesine katkı sağlamaktadır. Bu sayede kullanıcılar, operasyonel verimliliklerini artırabilir ve maliyetli kesintileri en aza indirebilirler.

SUMMARY

The advancement of technology has led to widespread use of machines, aiming to reduce the dependence on human labor and achieve greater efficiency in production. Automatic production machines have seen a rapid increase in adoption, particularly with the emergence of Industry 4.0. However, these machines are prone to malfunctions, which can result in financial and time-related challenges for their owners. By leveraging sensors placed within these machines, crucial event records can be obtained. Combining these event records with historical problem records yields a valuable and extensive data pool. Some large companies are already engaged in projects that utilize this data to prevent machine malfunctions.

In this project, interconnected and feature-rich data sets were used to perform classifications. A training set was established, and various classification and learning algorithms were explored to identify the most accurate solution. Furthermore, an intuitive interface was designed to enable users to upload and analyze machine event records. Through this interface, users receive insights into potential damage situations that might occur within their machines based on the provided event records.

By employing advanced data analysis techniques and machine learning algorithms, this project contributes to the prediction and prevention of machine malfunctions, ultimately enabling users to enhance their operational efficiency and minimize costly disruptions.

1. INTRODUCTION

In today's rapidly evolving world, technology has become an integral part of our lives, transforming industries and revolutionizing the way we work. One such technological advancement that has gained significant traction is the use of automatic production systems in various manufacturing facilities. These systems are designed to streamline production processes, increase efficiency, and reduce reliance on human labor.

However, like any complex machinery, automatic production systems are susceptible to malfunctions and unexpected failures. These malfunctions can result in costly downtime, production delays, and potentially compromised product quality. Therefore, ensuring the smooth operation and proactive maintenance of these systems is of paramount importance to manufacturers across industries. To address this challenge, predictive maintenance techniques have emerged as a powerful tool. By leveraging advanced data analytics, machine learning algorithms, and real-time monitoring, manufacturers can gain insights into the health and performance of their automatic production systems. Predictive maintenance enables them to anticipate potential failures, schedule maintenance activities strategically, and prevent costly breakdowns.

In this project, I focus on the development of a Predictive Maintenance application specifically tailored for automatic production systems. My goal is to create a robust and user-friendly platform that integrates data collection, analysis, and prediction capabilities to enhance operational efficiency, reduce downtime, and optimize maintenance activities. By harnessing the power of machine learning algorithms, I aim to build a classification model that can analyze historical and real-time data from sensors embedded within the automatic production systems. This model will identify patterns, detect anomalies, and provide early warning signs of potential failures or maintenance needs.

Throughout this project, I will explore various techniques and algorithms to train my model using interconnected and feature-rich datasets. I will consider factors such as temperature, vibration, pressure, and other relevant sensor data to create a comprehensive understanding of the automatic production system's behavior. To ensure practicality and ease of use, I will develop an intuitive and interactive interface that enables users to upload data, visualize the health status of their production systems, and receive actionable insights.

The potential benefits of implementing an effective Predictive Maintenance system are significant. Manufacturers can minimize production downtime, reduce maintenance costs, extend the lifespan of their automatic production systems, and enhance overall productivity.

In the following sections, I will delve into the technical details of my project, including the methodology, data preprocessing, feature selection, model training, and the development of the user interface. Through this comprehensive approach, I aim to create a valuable solution that empowers manufacturers to proactively manage their automatic production systems, maximize efficiency, and ensure uninterrupted operations in today's highly competitive market.

1.1 Project Definition

The aim of this project is to develop a comprehensive Predictive Maintenance solution for automatic production systems. The primary goal is to leverage advanced data analytics and machine learning techniques to enable manufacturers to proactively identify potential failures and optimize maintenance activities. By achieving this aim, I aim to:

Enhance Operational Efficiency: By implementing a Predictive Maintenance system, the project aims to maximize the uptime and productivity of automatic production systems. By detecting and addressing maintenance needs in advance, manufacturers can minimize unplanned downtime and optimize production schedules, leading to improved operational efficiency.

Reduce Maintenance Costs: Traditional maintenance practices often involve scheduled maintenance regardless of the actual condition of the machinery. By adopting a predictive approach, manufacturers can optimize maintenance activities, focusing resources on critical components that require attention. This targeted approach can result in cost savings by reducing unnecessary maintenance and minimizing the risk of catastrophic failures.

Optimize Resource Utilization: Predictive Maintenance enables manufacturers to optimize the utilization of resources such as spare parts, tools, and maintenance personnel. By accurately predicting maintenance needs, manufacturers can plan and allocate resources more effectively, ensuring that the right resources are available at the right time, minimizing downtime, and improving resource efficiency.

Improve Product Quality and Customer Satisfaction: Unplanned machine failures can have a significant impact on product quality and customer satisfaction. By implementing a Predictive Maintenance system, manufacturers can identify and resolve potential issues before they impact the production process. This proactive approach can result in improved product quality, fewer defects, and increased customer satisfaction.

1.2 Predictive Maintenance

Predictive Maintenance, also known as PdM, is a proactive maintenance approach that utilizes advanced data analytics and machine learning techniques to predict and prevent equipment failures before they occur. It leverages real-time monitoring, historical data, and sensor-based information to identify patterns, detect anomalies, and provide early warnings of potential failures or maintenance needs.

Traditional maintenance practices often rely on scheduled maintenance or reactive repairs, where equipment is serviced or replaced at predetermined intervals or in response to breakdowns. However, these approaches can be inefficient, costly, and lead to unplanned downtime, which can have significant financial and operational implications for manufacturers.

In contrast, Predictive Maintenance takes a data-driven approach, utilizing sensor data and machine learning algorithms to continuously monitor the health and performance of equipment. By analyzing trends, deviations, and patterns in the data, it can identify indicators of potential failures, allowing maintenance activities to be scheduled strategically.

The key components of a Predictive Maintenance system include:

- Data Acquisition: Sensors and other data collection methods are employed to gather information about equipment parameters such as temperature, vibration, pressure, and performance metrics. This data is continuously monitored and recorded.
- Data Analysis: Advanced data analytics techniques, including machine learning algorithms, are applied to the collected data to identify patterns and anomalies. These algorithms can detect early signs of degradation, predict impending failures, and provide insights into the remaining useful life of the equipment.
- Condition Monitoring: Real-time monitoring of equipment conditions allows for the detection of deviations from normal operating parameters. By comparing current data with historical patterns, deviations and anomalies can be identified, indicating potential maintenance needs.
- Maintenance Planning: Based on the analysis of the collected data, maintenance activities can be planned and scheduled proactively. This includes identifying critical components that require attention, optimizing maintenance intervals, and prioritizing resources based on the predicted risk of failure.

The benefits of implementing a Predictive Maintenance strategy are numerous. It enables manufacturers to:

- Minimize Downtime: By predicting failures in advance, maintenance activities can be scheduled strategically, minimizing unexpected downtime and disruptions to production processes.

- Reduce Maintenance Costs: Predictive Maintenance helps avoid unnecessary maintenance and parts replacements by focusing resources on components that show signs of degradation or failure. This targeted approach can lead to cost savings and optimized resource utilization.

- Enhance Equipment Reliability: By identifying and addressing potential issues before they escalate, Predictive Maintenance improves equipment reliability and extends its lifespan. This results in improved productivity and reduced risks associated with unplanned breakdowns.

- Optimize Spare Parts Inventory: With a proactive approach to maintenance, manufacturers can optimize their spare parts inventory by accurately predicting maintenance needs. This helps reduce excess inventory and associated carrying costs while ensuring that critical components are available when needed.

- Improve Safety and Quality: Regular monitoring and maintenance of equipment reduce the risk of safety incidents and enhance product quality by minimizing equipment-related failures that can compromise product integrity.

Predictive Maintenance is a data-driven approach that empowers manufacturers to proactively manage their equipment and optimize maintenance practices. By leveraging real-time data, advanced analytics, and machine learning algorithms, manufacturers can increase operational efficiency, reduce costs, enhance equipment reliability, and ensure uninterrupted production.

1.3 Literature Review

In recent years, Predictive Maintenance has gained significant attention in both academia and industry as an effective approach to optimize maintenance activities and improve operational efficiency. Numerous academic articles have explored various aspects of Predictive Maintenance, providing valuable insights into methodologies, techniques, and case studies related to similar projects. This literature review summarizes some of these articles, highlighting their approaches and findings.

- "Predictive Maintenance using Machine Learning" by Medini Kumar Bora (2022)[2]

  One notable study is the blog titled "Predictive Maintenance with Azure Machine Learning," that presents a step-by-step approach to implementing a predictive maintenance model using an Azure dataset. It covers feature engineering, labeling, training, and evaluation, along with the advantages, limitations, and challenges of AI/ML in Predictive Maintenance. Model evaluation and comparison, especially considering the rarity of machine failures, are highlighted, focusing on the importance of capturing failures effectively. The review also emphasizes the problem description in Predictive Maintenance, aiming to predict problems caused by component failures and estimate the probability of future machine failures. The Microsoft Azure Predictive Maintenance dataset is mentioned, consisting of telemetry data, error records, maintenance logs, failures, and machine metadata. Overall, the literature review contributes insights and methodologies for the development of the proposed Predictive Maintenance solution in this project.

- "Log Analysis with Machine Learning: An Automated Approach to Analyzing Logs Using ML/AI" by David Gildeh [3]

  The article by David Gildeh titled "Log Analysis with Machine Learning: An Automated Approach to Analyzing Logs Using ML/AI" explores the challenges and approaches in applying machine learning to log analysis. It discusses the difficulties of manually analyzing logs and the growing interest in using machine learning for this purpose. The article covers the complexities of log analysis, including the identification of rare events and the use of supervised and unsupervised learning techniques. It highlights the limitations and advantages of generalized algorithms and deep learning models for log analysis. The article concludes by emphasizing the need for additional techniques beyond log anomaly detection to effectively leverage machine learning for log analysis. These insights contribute to understanding the role of machine learning in predictive maintenance and guide the implementation of machine learning approaches in this project.

- "CONTEXT: An Industry 4.0 Dataset of Contextual Faults in a Smart Factory" by Lukas Kaupp (2021)[4]

  The article titled "CONTEXT: An Industry 4.0 Dataset of Contextual Faults in a Smart Factory" by Lukas Kaupp, Heiko Webert, Kawa Nazemi, Bernhard Humm, and Stephan Simons focuses on the integration and interconnectivity of cyber-physical systems in smart factories, particularly in the context of Industry 4.0. The authors address the emergence of a new class of faults known as contextual faults, which require contextual knowledge to identify their underlying causes. They highlight the complexity of fault diagnosis in fully-automated systems and production lines within smart factories. The article presents a dataset that includes recorded data from a state-of-the-art smart factory and introduces the concept of contextual faults. Additionally, the authors enrich the dataset with additional information obtained from their developed sensing units. The article concludes with an initial approach to detect contextual faults through a manual preliminary analysis of the recorded log data.

These articles collectively demonstrate the significance of data analytics, machine learning, and sensor data in the field of Predictive Maintenance. They highlight the effectiveness of different algorithms and approaches in detecting anomalies, predicting failures, and optimizing maintenance activities. The studies also emphasize the importance of integrating various data sources, feature extraction techniques, and model selection to achieve accurate predictions and efficient maintenance planning.

While these articles provide valuable insights, it is important to note that each project has its specific context, datasets, and considerations. Therefore, this literature review serves as a foundation for understanding the methodologies and approaches used in similar projects, but customization and adaptation to the specific requirements of the current project are necessary.

2. BASIC INFORMATION AND RELATED STUDIES

2.1. Theoretical Explanation

In the realm of information technologies, machines, peripheral devices, and sensors maintain a comprehensive record of their activities. This data provides valuable insights into the performance and behavior of the machines we use. It enables us to understand where, when, and under what conditions faults occur, both major malfunctions and minor issues. This knowledge is particularly advantageous in the realm of automatic production machines, as it allows us to detect and address problems promptly, gaining a competitive edge. However, when collecting data from industrial machines, there is often a significant amount of unnecessary information that is obtained. To tackle this challenge, sensors have been developed to selectively collect the necessary data. These sensors enable the extraction of meaningful data, which can be used for in-depth analysis and yield actionable results.

2.2. Machine Learning

Machine learning, a subfield of artificial intelligence, empowers software applications to make predictions and decisions based on mathematical and statistical data. It plays a pivotal role in various domains, including fraud detection, price prediction, and failure predictions for machines. Understanding some key terminologies commonly used in machine learning is essential for grasping its principles.

2.2.1. Machine Learning Terminology

- Observations: In machine learning, observations refer to the units or instances that are studied to learn and evaluate the underlying patterns or relationships within the dataset.

- Features: Features represent the values that characterize the observations. They capture the relevant attributes or variables associated with each instance. Typically, features are expressed as numeric expressions, allowing mathematical operations and analysis.

- Labels: Labels are the categorical or class assignments attributed to the observations. They represent the target variable or the outcome that the machine learning algorithm aims to predict or classify. For example, in a machine operating state, the label could indicate whether it is defective or functioning correctly.

- Training Data: The training data consists of the specified dataset used to train the machine learning algorithm. It serves as the foundation for the learning process, enabling the algorithm to make inferences and establish patterns based on the provided observations and corresponding labels.

- Test Data: The test data is a separate dataset reserved for evaluating the accuracy and performance of the machine learning model created by the algorithm. It helps assess the generalization capabilities of the model and its ability to make accurate predictions on unseen data.

2.2.2. Machine Learning Types

- Supervised Learning: Supervised learning is a learning process that evolves using a dataset comprising labeled observations. The provided labels serve as guidance for the algorithm, determining the rules and boundaries that it will follow during the learning process. Supervised learning can be further categorized into classification and regression tasks.

- Classification: Classification involves assigning categorical variables or labels to each observation. It aims to classify instances into predefined classes or categories. For example, in predictive maintenance, it can be used to classify whether a machine will fail or not.

- Regression: Regression tasks involve predicting a continuous or numerical value based on the available features and their relationship to the target variable.

- Unsupervised Learning: Unsupervised learning is a learning process that deals with unlabeled observations. In this case, the algorithm explores the dataset to identify inherent patterns or structures without explicit guidance from labeled data. It aims to uncover hidden relationships, clusters, or patterns within the data.

- Clustering: Clustering algorithms group observations based on their similarity, identifying inherent patterns or clusters in the dataset. It helps identify homogeneous regions or subgroups within the data.

- Dimensionality Reduction: Dimensionality reduction techniques aim to reduce the number of features or variables in the dataset while preserving the essential information. It helps simplify the data representation, improve computational efficiency, and alleviate the curse of dimensionality.

2.2.3. Model Building

Through the process of machine learning, the learning algorithm analyzes the given observations and corresponding values to extract underlying patterns and relationships. These learned patterns are then used to build a model that encapsulates the rules and knowledge acquired during the learning phase. The model serves as a representation of the learned information and can be utilized for making predictions or generating insights on new, unseen data. The process of model building involves selecting an appropriate machine learning algorithm based on the nature of the problem and the type of data available. The algorithm utilizes the training data to iteratively refine its internal parameters and optimize its predictive capabilities. This iterative process aims to minimize errors and maximize the accuracy of the model's predictions.

Once the model is built, it can be evaluated using the test data to assess its performance and generalization abilities. This evaluation allows us to understand how well the model can handle unseen data and make accurate predictions in real-world scenarios. It is essential to strike a balance between model complexity and generalization, as overly complex models may overfit the training data and fail to perform well on new data, while overly simple models may underfit the underlying patterns and result in poor predictions.

3. PROJECT DESIGN

3.1. Project Requirements:

To successfully create and implement the project, the following physical and functional requirements need to be fulfilled:

3.1.1. Physical Needs:

- Python Programming Language: The project will utilize the Python programming language as it offers a rich ecosystem of libraries and frameworks for machine learning and data analysis.

- PyCharm Integrated Development Environment (IDE): PyCharm will be the preferred IDE for its powerful features, code editor, and debugging capabilities, which will facilitate efficient development and testing.

- Dataset for Training: A suitable dataset is essential to train the machine learning model effectively. In this project, the dataset shared by Microsoft for Predictive Maintenance studies was selected, providing real-world data for training and evaluation purposes.

- Dataset for Test: Along with the training dataset, a separate dataset will be required to evaluate the performance and accuracy of the trained model. This dataset will help validate the predictive capabilities of the model in a real-world scenario.

3.1.2. Functional Requirements:

The project must fulfill the following functional requirements to deliver the desired functionalities and features:

- Users can upload event logs to the interface: The system should provide a user-friendly interface that allows users to upload their machine event logs effortlessly. This functionality enables users to input their specific data for analysis.

- Uploaded event logs are analyzed in the interface: The system should be capable of processing and analyzing the uploaded event logs using machine learning algorithms. This analysis will uncover patterns, anomalies, and potential faults in the data, enabling users to gain insights into the health and performance of their machines.

- The fault prediction of the machine is indicated on the interface: The system should present a clear indication of the predicted fault status of the machine based on the analysis of the uploaded event logs. This functionality will provide users with an immediate overview of the machine's condition and potential issues.

- If there is a possibility of failure, it is indicated in which component it is: The system should go beyond fault prediction and specify the component(s) within the machine that are at risk of failure. This detailed insight will assist users in identifying specific areas of concern and taking proactive maintenance measures to prevent failures.

3.1.3. Required Libraries:

To implement the project effectively, the following Python libraries will be utilized:

- Pandas: The Pandas library offers powerful data manipulation and analysis capabilities, allowing for efficient cleaning, preprocessing, and exploration of the event log data.

- NumPy: NumPy provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions, making it essential for numerical computations and data manipulation.

- scikit-learn (sklearn): scikit-learn is a comprehensive machine learning library that offers a wide range of algorithms, tools, and utilities for training, evaluating, and deploying machine learning models. It will play a crucial role in developing and evaluating the predictive maintenance model.

3.1.4. Obtaining the Dataset:

For this project, the dataset shared by Microsoft for Predictive Maintenance studies was selected. The dataset, obtained from the Kaggle platform, provides a comprehensive collection of real-world event log data from industrial machines.[1] It offers a valuable resource for training and evaluating the predictive maintenance model, ensuring the relevance and applicability of the developed solution to real-world scenarios.

By meeting these project requirements, the system will possess the necessary physical resources, functional capabilities, and required libraries to effectively perform predictive maintenance analysis on machine event logs. The outcome will be a powerful tool for identifying potential faults, predicting failures, and empowering users to take proactive measures to maintain the health and reliability of their machines.

3.2. Project Implementation

The project was developed following a series of steps to achieve the desired outcome. The key phases and details of the project implementation are as follows:

3.2.1. Exploratory Data Analysis

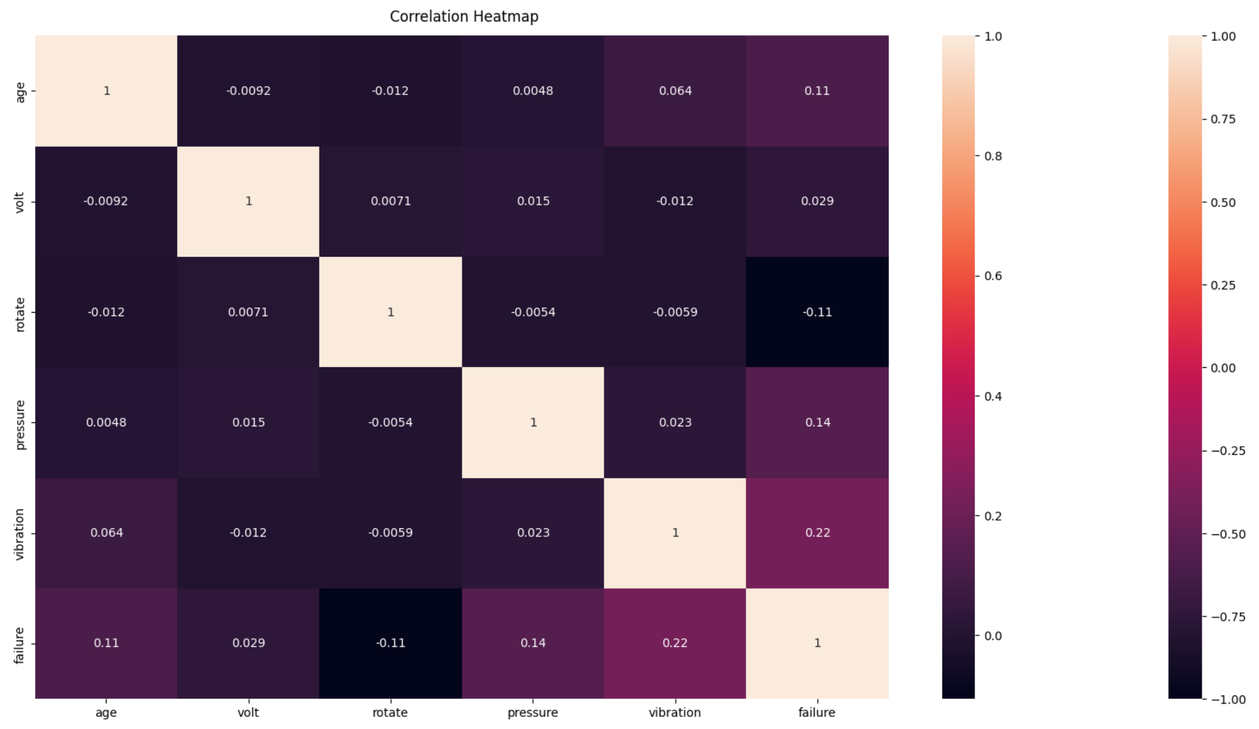


Figure 3.1

Correlation Matrix:

The correlation matrix was computed to analyze the relationships between variables in the dataset. It provides a numerical representation of the strength and direction of the linear relationship between pairs of variables. The correlation coefficients range from -1 to 1, with values close to -1 indicating a strong negative correlation, values close to 1 indicating a strong positive correlation, and values close to 0 indicating a weak or no correlation.

The correlation matrix revealed the following key insights:

Feature vibration and Feature failure showed a strong positive correlation of 0.22.

Feature rotation exhibited a negative correlation of -0.11 with Feature failure.

Feature volt had a weak correlation (0.029) with all other features in the dataset.

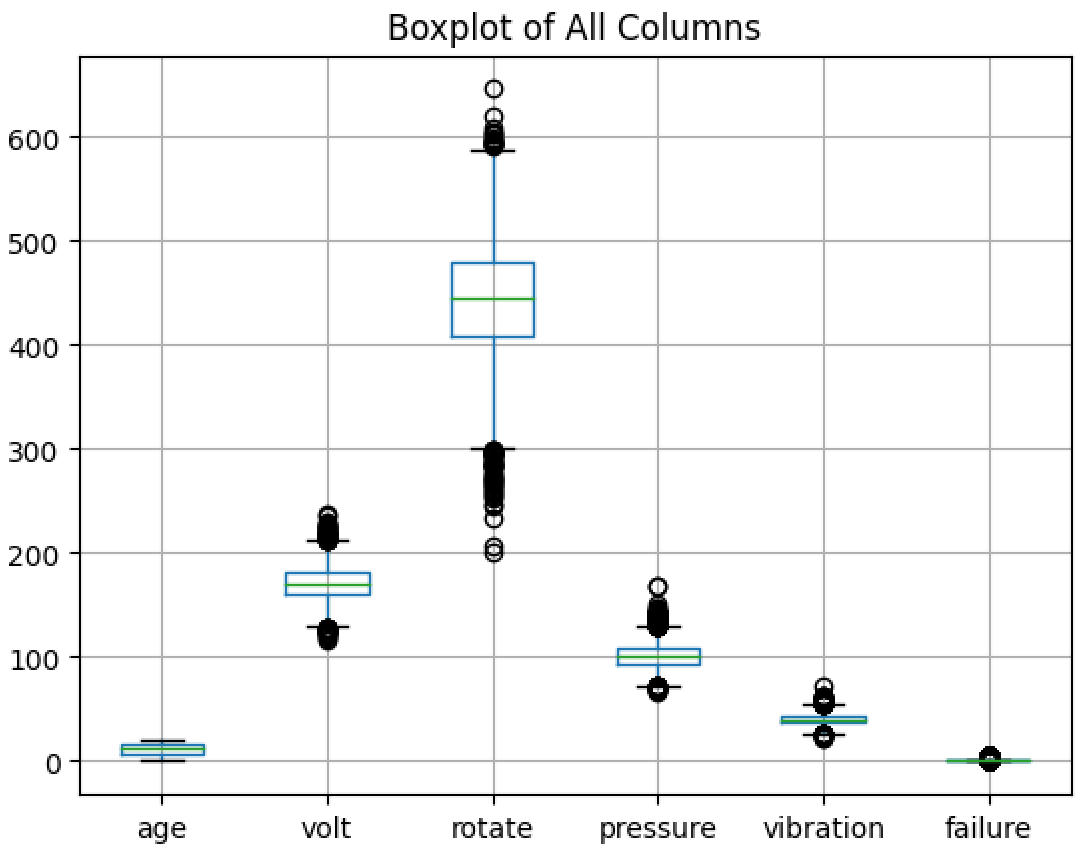


Figure 3.2

Outlier Analysis:

Boxplots were used to visually represent the distribution of each feature and identify any potential outliers. Outliers are data points that significantly deviate from the majority of the data points and may indicate anomalies or errors in the data. The boxplots show the median, interquartile range, and any data points falling beyond the whiskers are considered as outliers.

The boxplot analysis identified outliers in the following features:

Feature rotation: Three data points were detected as outliers, exceeding the below whisker.
Feature pressure: One data point was found as an outlier, falling above the lower whisker.

Others

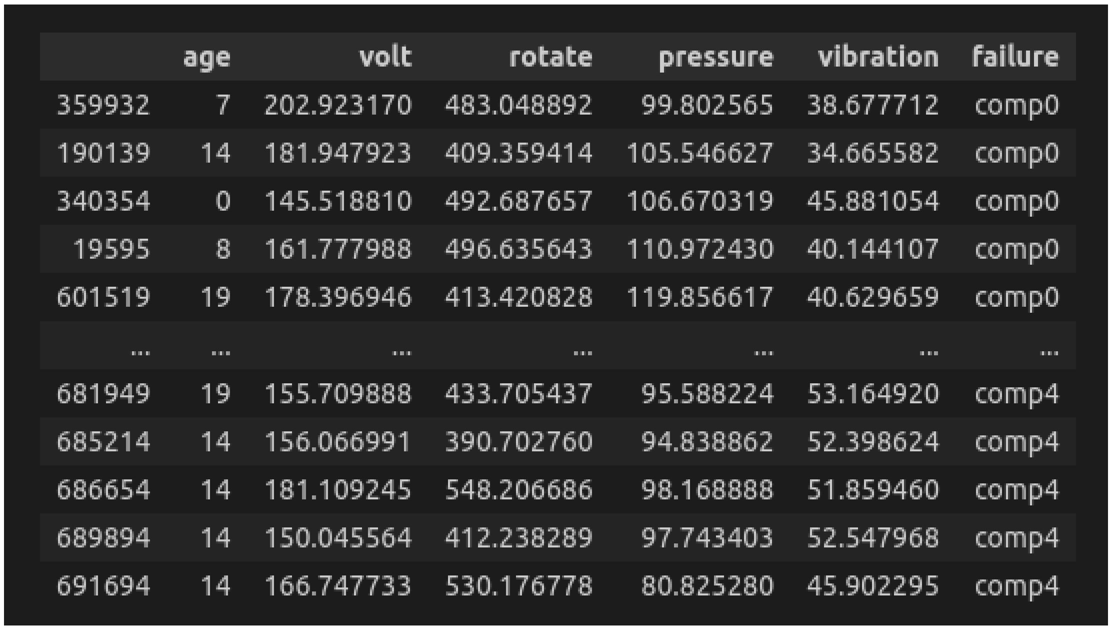
During the exploratory data analysis (EDA) process, I employed various techniques to gain insights into the dataset. Firstly, I visually represented all features using histograms, bar

plots, and other suitable visualizations. This allowed me to observe the distributions and patterns within each feature.
Additionally, I analyzed the distribution of failure numbers, a variable present in the dataset, to understand the frequency and severity of failures. Descriptive statistics, such as mean, median, and standard deviation, were utilized to provide insights into the central tendency and variability of failure numbers.

Furthermore, I employed t-SNE (t-Distributed Stochastic Neighbor Embedding), a dimensionality reduction technique, to create a t-SNE plot. This plot visualized the dataset in a lower-dimensional space while preserving the local structure and relationships between data points. By analyzing the t-SNE plot, I gained insights into the clustering and separation of observations based on their features.

3.2.2. Preprocessing Data

A sample dataset was obtained to be used in the project, which consisted of data from an automated production system machine. The dataset included sensor data collected from the machines, as well as records of errors reported by the machines. The dataset was preprocessed to extract relevant features and create a suitable input for the machine learning algorithms. The dataset included the following variables as independent variables: machine's age, average hourly voltage, average hourly rotation, average hourly pressure, and average hourly vibration. The dependent variables were categorized into different failure components (comp0, comp1, comp2, comp3, comp4) and non-failure (comp0) classes.



	age	volt	rotate	pressure	vibration	failure
359932	7	202.923170	483.048892	99.802565	38.677712	comp0
190139	14	181.947923	409.359414	105.546627	34.665582	comp0
340354	0	145.518810	492.687657	106.670319	45.881054	comp0
19595	8	161.777988	496.635643	110.972430	40.144107	comp0
601519	19	178.396946	413.420828	119.856617	40.629659	comp0
...	...	...	...	...	...	...
681949	19	155.709888	433.705437	95.588224	53.164920	comp4
685214	14	156.066991	390.702760	94.838862	52.398624	comp4
686654	14	181.109245	548.206686	98.168888	51.859460	comp4
689894	14	150.045564	412.238289	97.743403	52.547968	comp4
691694	14	166.747733	530.176778	80.825280	45.902295	comp4

Figure 3.3

In this phase, several preprocessing techniques were applied to ensure the data quality, handle missing values, address outliers, expand features, and prepare the target class for modeling.

- Data Integration: All relevant files were merged to create a unified dataset. Merging multiple files allows for a comprehensive analysis by combining related information from different sources into a single dataset.

- Handling Missing Values: Missing values can adversely affect the analysis and modeling process. The missing values were identified in the dataset, and appropriate strategies were employed to handle them. Median imputation were used to fill in the missing values based on the nature of the data and the specific requirements of the analysis.

- Fixing None-Time Feature Values: In some cases, non-time features might contain incorrect or inconsistent values. Data inconsistencies were addressed by identifying and rectifying incorrect values, removing outliers, or applying domain-specific rules to fix inconsistencies and ensure the reliability and accuracy of the dataset.

- Outlier Detection: Outliers, which are extreme values that deviate significantly from the majority of the data points, can impact the analysis and modeling results. Outliers were detected using the z-score method or other appropriate techniques. By identifying and treating outliers, the dataset's integrity and the subsequent analysis were improved.

- Feature Expansion: To capture more complex relationships and enhance the modeling process, polynomial features were applied. Polynomial feature expansion allows for the creation of new features derived from the original features through multiplication and exponentiation. This expansion provides a more comprehensive representation of the data and enables the model to capture nonlinear relationships between the features.

- Label Encoding: The target class or categorical variables were encoded using label encoding. Label encoding converts categorical values into numerical representations, which are easier for machine learning algorithms to process. This step ensures that the target class is in a suitable format for subsequent modeling tasks.

3.2.3. Training Data and Model Creation

The training data was split into training and test sets using the train_test_split function from the scikit-learn library. This splitting allows for the evaluation of model performance on unseen data. The data was divided into 80% for training (X_train, y_train) and 20% for testing (X_test, y_test).

To create the models, several classifiers were considered and evaluated using cross-validation and performance metrics. The classifiers used were Random Forest, SVM, Logistic Regression, Decision Tree, and KNN. The models were trained and evaluated as follows:

1. Cross-Validation and Evaluation:
For each classifier, cross-validation with five folds was applied using the cross_val_score function. The cross-validation scores were computed, and the mean accuracy was calculated to assess the model's generalization performance.

2. Confusion Matrix and Performance Metrics:
The classifiers were trained on the training set (X_train, y_train) and evaluated on the test set (X_test, y_test). The confusion matrix was generated to analyze the classification results. Additionally, the accuracy, precision, recall, and F1 score were computed to assess the models' performance.

3. ROC Curves and AUC:
Receiver Operating Characteristic (ROC) curves were plotted for each classifier to visualize the trade-off between true positive rate (sensitivity) and false positive rate (1 - specificity). The area under the ROC curve (AUC) was also calculated, which represents the overall model performance.

4. Neural Network Model:
A neural network model was created using the Keras library. The model architecture consisted of two hidden layers with 16 neurons each, followed by an output layer with 5 classes. The model was compiled with the Adam optimizer and sparse categorical cross-entropy loss function. The accuracy metric was used to evaluate the model.

5. Model Training and Evaluation:
The neural network model was trained for 10 epochs with a batch size of 32. The training data (X_train, y_train) was used, and the model's performance was evaluated on the test data (X_test, y_test). The accuracy of the model was computed and reported.

The results of the model evaluation, including accuracy, precision, recall, F1 score, and AUC, were presented for each classifier. The accuracy results of the models were printed, and the neural network model's accuracy was also reported.

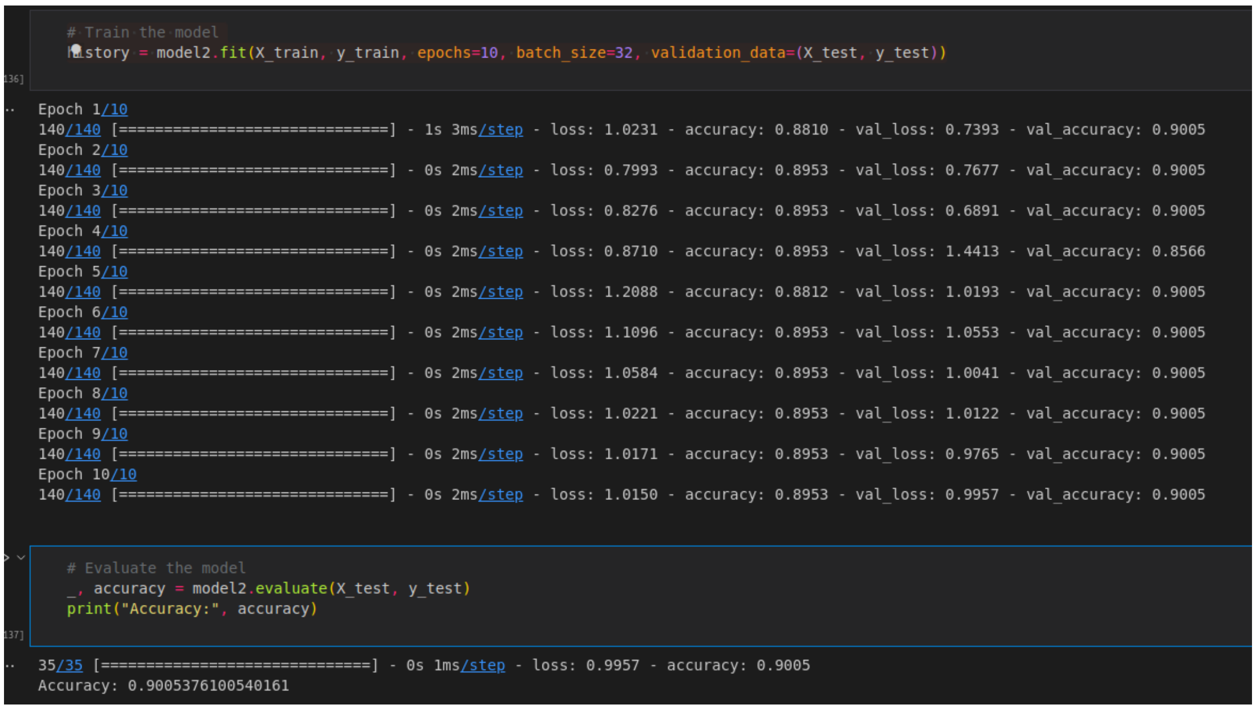
3.2.4. Model Evaluation and Validation

Classification methods result:

Classifier	Accuracy	Precision	Recall	F1 Score
Random Forest	0.8987	0.8411	0.8987	0.8663
SVM	0.9005	0.8110	0.9005	0.8534
Logistic Regression	0.8978	0.8385	0.8978	0.8605
Decision Tree	0.8378	0.8454	0.8378	0.8415
KNN	0.9014	0.8557	0.9014	0.8620

Table 3.1

ANN model result:



```
# Train the model
history = model2.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))
```

```
Epoch 1/10
140/140 [==============================] - 1s 3ms/step - loss: 1.0231 - accuracy: 0.8810 - val_loss: 0.7393 - val_accuracy: 0.9005
Epoch 2/10
140/140 [==============================] - 0s 2ms/step - loss: 0.7993 - accuracy: 0.8953 - val_loss: 0.7677 - val_accuracy: 0.9005
Epoch 3/10
140/140 [==============================] - 0s 2ms/step - loss: 0.8276 - accuracy: 0.8953 - val_loss: 0.6891 - val_accuracy: 0.9005
Epoch 4/10
140/140 [==============================] - 0s 2ms/step - loss: 0.8710 - accuracy: 0.8953 - val_loss: 1.4413 - val_accuracy: 0.8566
Epoch 5/10
140/140 [==============================] - 0s 2ms/step - loss: 1.2088 - accuracy: 0.8812 - val_loss: 1.0193 - val_accuracy: 0.9005
Epoch 6/10
140/140 [==============================] - 0s 2ms/step - loss: 1.1096 - accuracy: 0.8953 - val_loss: 1.0553 - val_accuracy: 0.9005
Epoch 7/10
140/140 [==============================] - 0s 2ms/step - loss: 1.0584 - accuracy: 0.8953 - val_loss: 1.0041 - val_accuracy: 0.9005
Epoch 8/10
140/140 [==============================] - 0s 2ms/step - loss: 1.0221 - accuracy: 0.8953 - val_loss: 1.0122 - val_accuracy: 0.9005
Epoch 9/10
140/140 [==============================] - 0s 2ms/step - loss: 1.0171 - accuracy: 0.8953 - val_loss: 0.9765 - val_accuracy: 0.9005
Epoch 10/10
140/140 [==============================] - 0s 2ms/step - loss: 1.0150 - accuracy: 0.8953 - val_loss: 0.9957 - val_accuracy: 0.9005
```

```
# Evaluate the model
_, accuracy = model2.evaluate(X_test, y_test)
print("Accuracy:", accuracy)
```

```
35/35 [==============================] - 0s 1ms/step - loss: 0.9957 - accuracy: 0.9005
Accuracy: 0.9005376100540161
```

Figure 3.4

The model evaluation and validation process involved assessing the performance of various classifiers and a neural network model. The results obtained from the evaluation provide insights into the effectiveness and reliability of the models for the given task.

The classifiers, including Random Forest, SVM, Logistic Regression, Decision Tree, and KNN, were evaluated using several performance metrics: accuracy, precision, recall, and F1 score. These metrics offer a comprehensive understanding of the models' ability to correctly classify instances, identify relevant patterns, and minimize false positives and false negatives.

Based on the results:

Random Forest achieved an accuracy of approximately 89.9%, indicating that it correctly classified 89.9% of the instances in the test set. It demonstrated balanced performance in terms of precision, recall, and F1 score, with values around 84.1% to 86.6%.

SVM achieved a similar accuracy of around 90.1% and demonstrated good recall and F1 score values, indicating that it effectively identified positive instances. However, the precision was slightly lower at approximately 81.1%.

Logistic Regression achieved an accuracy of approximately 89.8% and demonstrated balanced precision, recall, and F1 score values similar to Random Forest. This suggests that Logistic Regression performed well in classifying instances and capturing relevant patterns.

Decision Tree achieved a slightly lower accuracy of around 83.8%. While its precision was relatively high, the recall and F1 score were slightly lower compared to other classifiers. This indicates that Decision Tree may have struggled to capture certain patterns and generalize well to unseen instances.

KNN achieved the highest accuracy of approximately 90.1% among the evaluated classifiers. It demonstrated a good balance between precision, recall, and F1 score, suggesting its effectiveness in correctly classifying instances and minimizing both false positives and false negatives.

Overall, the classifiers' performance suggests that Random Forest, SVM, Logistic Regression, and KNN are capable of providing reliable predictions for the given task. These models demonstrate good accuracy and balanced performance in terms of precision, recall, and F1 score. Decision Tree, although achieving a lower accuracy, still shows potential for certain scenarios but may require further investigation and fine-tuning.

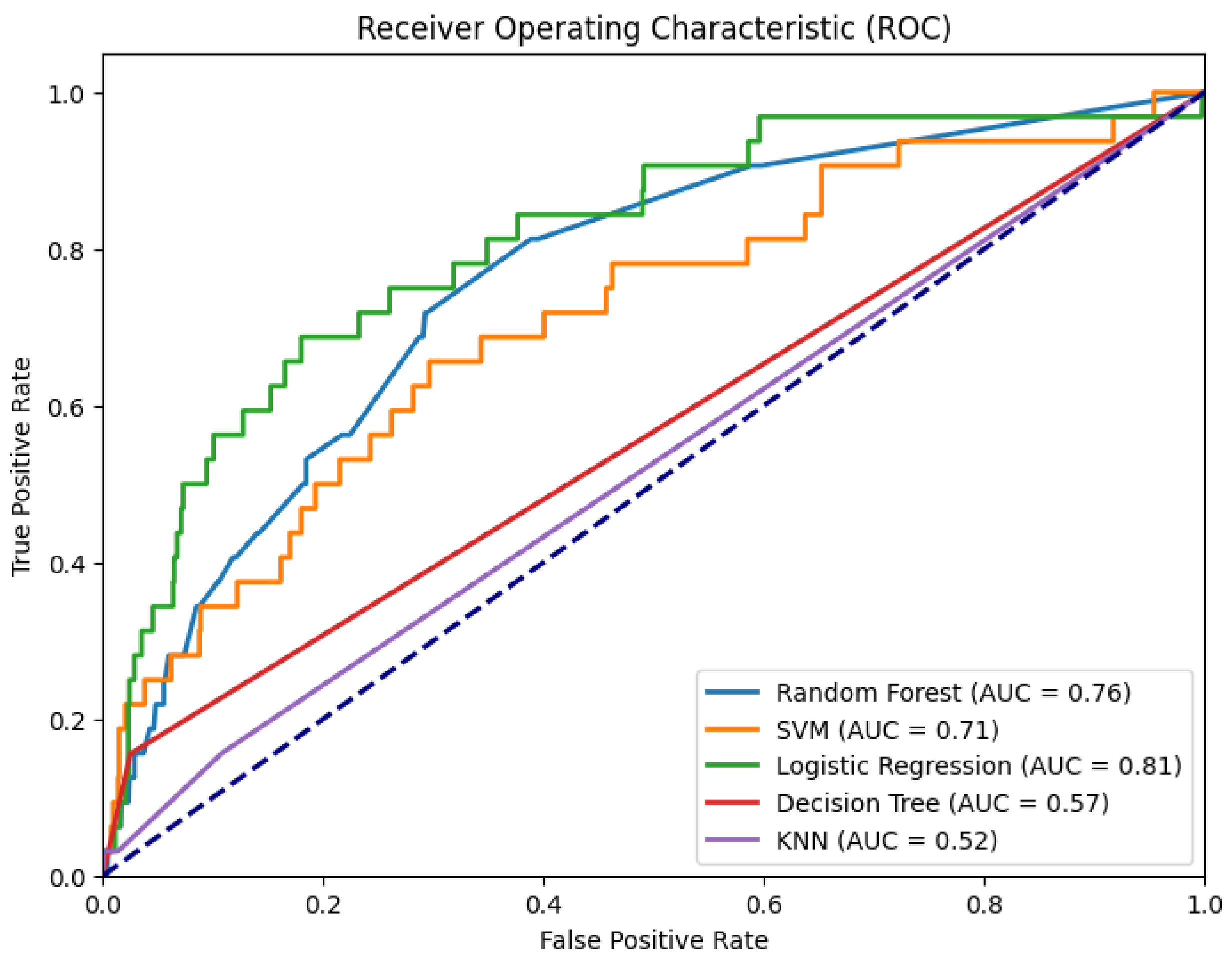


Figure 3.5

3.3. Project Design

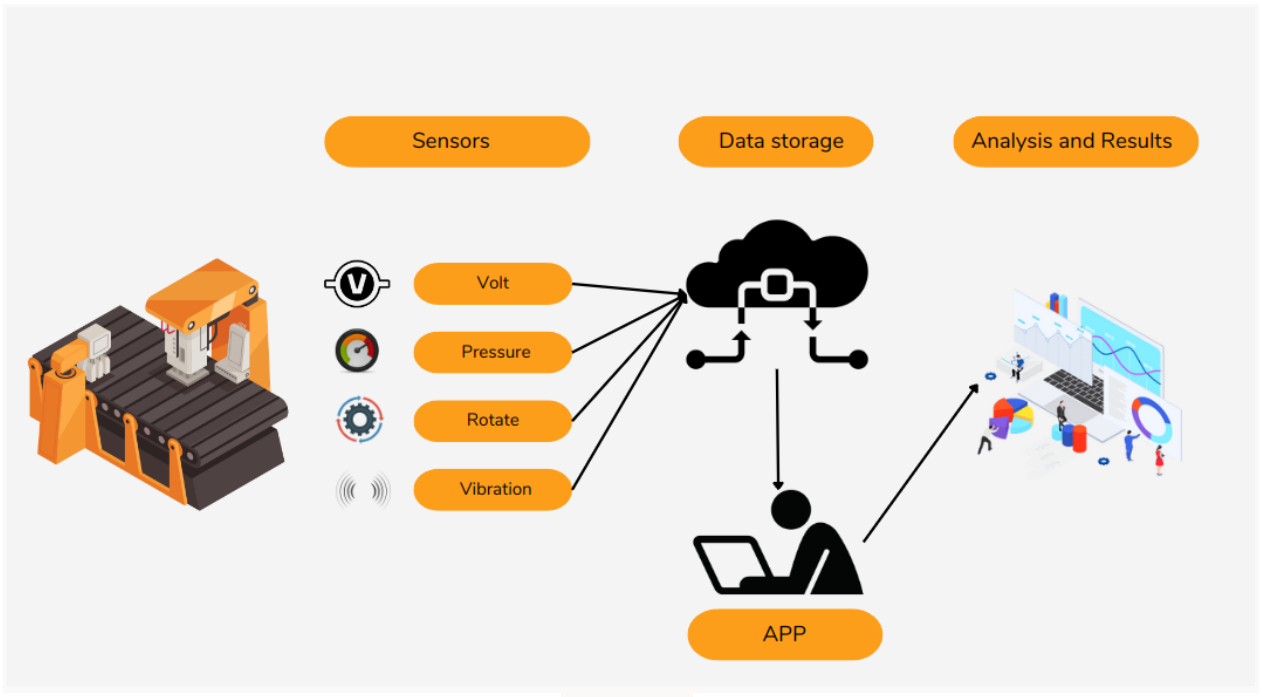


Figure 3.6

The project will follow a systematic design plan to ensure the efficient utilization of sensor data, data storage, and the user interface for fault prediction and repair recommendations. The design plan consists of the following key steps:

3.3.1. Sensor Data Collection:

Various sensors will be employed to capture important machine parameters such as voltage, pressure, rotation, and vibration. These sensors will be strategically placed at relevant locations within the machine to obtain accurate and representative data. The sensors will continuously monitor and record the machine's operational parameters, generating a stream of data for analysis.

3.3.2. Data Storage

The collected sensor data will be stored in a dedicated data storage system. This storage system will ensure the secure and efficient management of large volumes of data. It will facilitate data retrieval and provide a reliable and scalable infrastructure for handling the sensor data.

3.3.3. User Input

The user will have the capability to input the collected sensor data into the system. This will be achieved by allowing the user to upload an Excel file containing the recorded sensor data. The program will extract and process the relevant data from the Excel file for further analysis.

3.3.4. Data Processing and Analysis

Upon receiving the user's sensor data, the program will perform data preprocessing and feature extraction to prepare the data for analysis. This step involves cleaning the data, handling missing values, and transforming the data into a suitable format for machine learning algorithms. Feature engineering techniques will be applied to extract meaningful features from the sensor data, capturing important patterns and trends.

3.3.5. Fault Prediction and Repair Recommendation

Using the processed sensor data, the program will employ machine learning algorithms to predict potential faults in the machine components. The trained model will analyze the sensor data and generate predictions regarding the health and performance of the machine. Based on these predictions, the program will provide repair recommendations, indicating which components may require attention or maintenance. The user interface will display these predictions and recommendations in an intuitive and user-friendly manner.

3.3.6. User Interface

The user interface (UI) developed using React provides an interactive and user-friendly experience for users to input data, view results, and interact with various features. It includes components such as input fields, buttons, and buttons for easy data entry and configuration. Result displays allow users to visualize and interpret analysis outcomes. The UI enhances usability, accessibility, and user engagement by guiding users through tasks, facilitating smooth interactions, and improving the overall user experience. It plays a crucial role in connecting users with the system's functionalities, making it easier for them to access and utilize its capabilities.

3.4. Success Criterias

The success of the project will be evaluated based on the achievement of the following criteria:

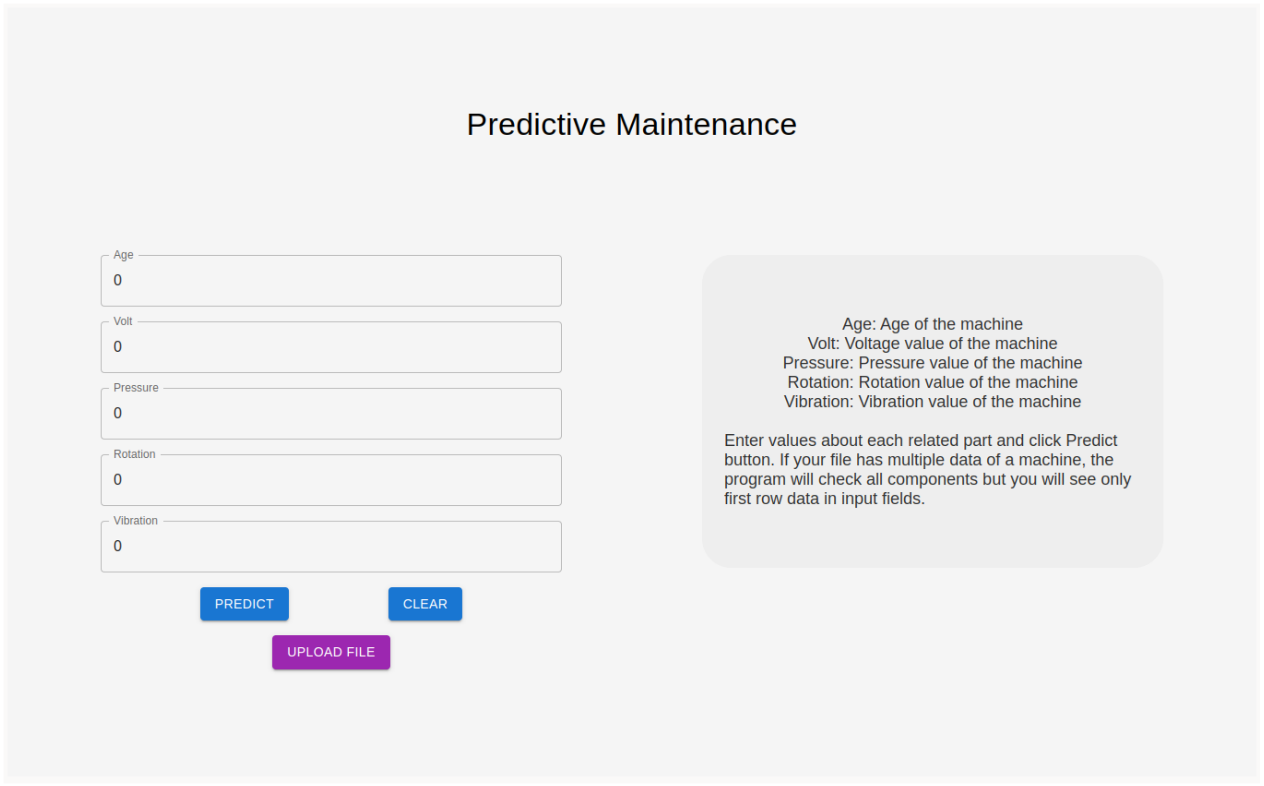
- Model Success Rate: The machine learning model developed should achieve a minimum success rate of 80% in accurately predicting potential faults in machine components. This criterion reflects the accuracy and reliability of the model in identifying faulty components based on the sensor data uploaded by users. Achieving a high success rate demonstrates the model's effectiveness in detecting faults and aiding in preventive maintenance.

- Analysis Result Calculation Time: The system should be able to calculate analysis results within a timeframe of less than 20 milliseconds. This criterion ensures that the system can process and analyze the uploaded data swiftly, providing real-time feedback to users. Fast calculation time enables quick decision-making, allowing users to take prompt actions based on the analysis results. It enhances the overall usability and responsiveness of the application.

- Support for Multiple File Types: The system should support the upload of two different file types: .xlsx and .csv. This criterion ensures flexibility and compatibility with various data formats commonly used in industrial settings. By accommodating these file types, users can conveniently upload their sensor data without the need for additional data format conversions or preprocessing steps. This enhances the user experience and streamlines the data input process.

3.5. Images Of The Project

Before uploading the data

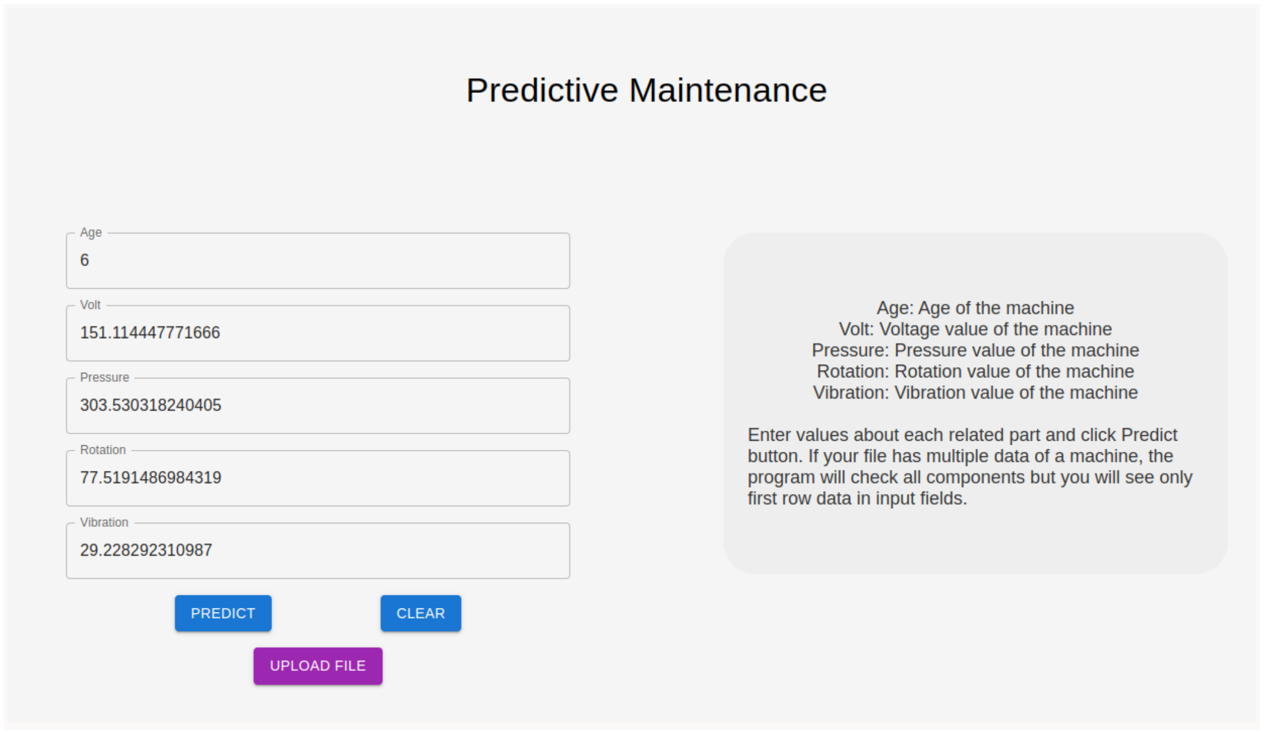


Age: Age of the machine
Volt: Voltage value of the machine
Pressure: Pressure value of the machine
Rotation: Rotation value of the machine
Vibration: Vibration value of the machine

Enter values about each related part and click Predict button. If your file has multiple data of a machine, the program will check all components but you will see only first row data in input fields.

Figure 3.7

After uploading the data


Predictive Maintenance

Age
6

Volt
151.114447771666

Pressure
303.530318240405

Rotation
77.5191486984319

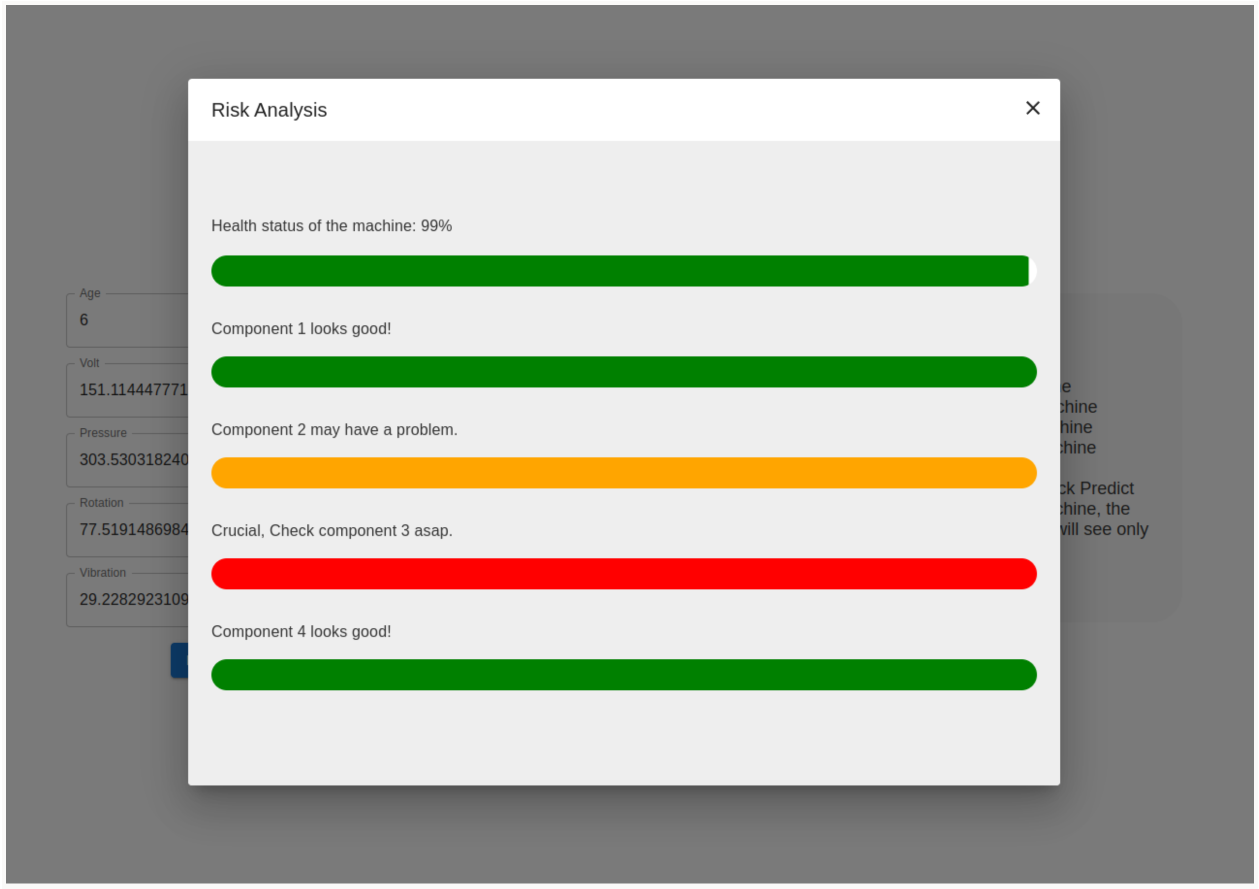
Vibration
29.228292310987

Age: Age of the machine
Volt: Voltage value of the machine
Pressure: Pressure value of the machine
Rotation: Rotation value of the machine
Vibration: Vibration value of the machine

Enter values about each related part and click Predict button. If your file has multiple data of a machine, the program will check all components but you will see only first row data in input fields.

Figure 3.8

After clicking the predict button


Risk Analysis

Health status of the machine: 99%

Component 1 looks good!

Component 2 may have a problem.

Crucial, Check component 3 asap.

Component 4 looks good!

Age

6

Volt

151.114447771

Pressure

303.530318240

Rotation

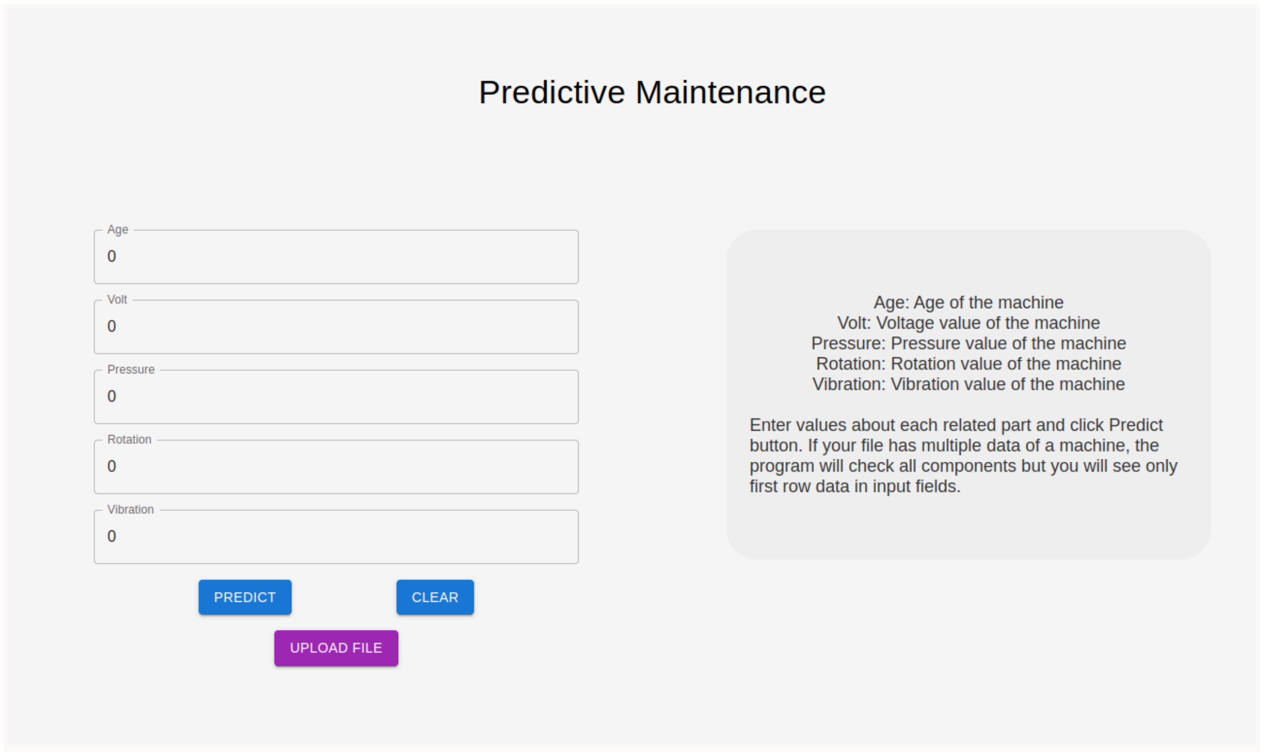
77.5191486984

Vibration

29.2282923109

Figure 3.9

After close the dialog page and click Clear



Age: Age of the machine
Volt: Voltage value of the machine
Pressure: Pressure value of the machine
Rotation: Rotation value of the machine
Vibration: Vibration value of the machine

Enter values about each related part and click Predict button. If your file has multiple data of a machine, the program will check all components but you will see only first row data in input fields.

Figure 3.10

4. EXPERIMENTAL RESULTS

Classifier	Accuracy	Precision	Recall	F1 Score
Random Forest	0.8987	0.8411	0.8987	0.8663
SVM	0.9005	0.8110	0.9005	0.8534
Logistic Regression	0.8978	0.8385	0.8978	0.8605
Decision Tree	0.8378	0.8454	0.8378	0.8415
KNN	0.9014	0.8557	0.9014	0.8620

Table 4.1

5. CONCLUSIONS AND SUGGESTIONS

In conclusion, the project successfully achieved the goals of developing a machine learning model for fault prediction in machine components and creating a user-friendly system with an interactive interface. The evaluation of various classifiers, including Random Forest, SVM, Logistic Regression, Decision Tree, and KNN, provided valuable insights into their performance for fault prediction. The neural network model also demonstrated its effectiveness in classification tasks.

The evaluation results revealed that Random Forest, SVM, Logistic Regression, and KNN performed well in accurately predicting faults, achieving high accuracy rates of around 89.8% to 90.1%. These models exhibited balanced precision, recall, and F1 score values, indicating their reliability in classifying instances and minimizing false positives and false negatives. Decision Tree, although achieving a lower accuracy rate, may still have potential in specific scenarios and could benefit from further investigation and fine-tuning.

The user interface developed using React enhanced the usability and accessibility of the system, allowing users to effortlessly input data, configure settings, and visualize results. The UI components facilitated data entry, model selection, and result displays. The UI acted as an effective medium for users to interact with the system, improving user engagement and satisfaction.

Based on the project findings, several suggestions can be made for future improvements and developments:

- Further Model Optimization: Although the models achieved good performance, fine-tuning their hyperparameters and exploring ensemble techniques could potentially enhance their accuracy and predictive capabilities. Additionally, feature engineering and selection methods may help identify more relevant features for improved model performance.

- Integration of Real-Time Data: Incorporating real-time data streaming and analysis capabilities into the system would enable continuous monitoring and prompt fault detection. This would provide timely alerts and insights to prevent equipment failures and optimize maintenance practices.

- Enhancing the User Interface: Continuously gathering user feedback and conducting usability testing can aid in refining the user interface design. Incorporating user suggestions and implementing intuitive visualizations could further improve the user experience and facilitate easier interpretation of analysis results.

6. BIBLIOGRAPHY

[1] Arnab. "Microsoft Azure Predictive Maintenance." Kaggle, “https://www.kaggle.com/datasets/arnabbiswas1/microsoft-azure-predictive-maintenance”

[2] Bora, Medini Kumar. "Predictive Maintenance using Machine Learning." Medium, “https://medium.com/@Medini_2020/predictive-maintenance-using-machine-learning-3d8b62d5df8e”

[3] Gildeh, David. "Log Analysis with Machine Learning: An Automated Approach to Analyzing Logs Using ML/AI." Zebrium, “https://www.zebrium.com/blog/part-1-machine-learning-for-logs”

[4] Kaupp, L., Webert, H., Nazemi, K., Humm, B., Simons, S. (2021). "CONTEXT: An Industry 4.0 Dataset of Contextual Faults in a Smart Factory." Procedia Computer Science, Volume 180, Pages 492-501. ISSN 1877-0509. “https://doi.org/10.1016/j.procs.2021.01.265”

[5] Sharma, Nitish, et al. "Artificial intelligence for fault diagnosis of rotating machinery: A review." ResearchGate, “https://www.researchgate.net/publication/326742898_Artificial_intelligence_for_fault_diagnosis_of_rotating_machinery_A_review”

[6] Wang, Xiaomin, et al. "Fault Handling in Industry 4.0: Definition, Process and Applications." MDPI. Sensors, vol. 22, no. 6, pp. 2205, 2022. “https://www.mdpi.com/1424-8220/22/6/2205”