Equipment Failure for Maintenance On Sensor Data

Ved Prakash Chaubey
upgrad Campus, upgrad Education Private Limited
Banglore,Karnataka,india
ORCID-0000-0003-3049-8316

Aman Gavel
School of CSE, Lovely Professional University & upgard Campus
Jalandhar,Punjab,India
aman.12018431@lpu.in

Raj Pandey
School of CSE, Lovely Professional University & upgard Campus
Jalandhar,Punjab,India
darshmay06@gmail.com

Rajnish Bharti
School of CSE,Lovely Professional University & upgard Campus
Jalandhar,Punjab,India
rajnishmehta6171@gmail.com

Ritesh Kumar
School of CSE,Lovely Professional University & upgard Campus
Jalandhar,Punjab,India
ritesh282001@gmail.com

Keywords: Sensor Data, Machine Learning, Deep Learning, NLP, Prediction, Explainable artificial intelligence; total production maintenance

Abstract

Predictive maintenance is essential to preventing equipment failure and reducing associated costs. Existing prediction models based on physical damage models and machine learning often do not provide good predictions. Characteristics of potential failure. Based on this time lag, we use incident monitoring data collected at random times to remove crime characteristics and significant locations that are used to track and predict crime incidents. In the context of the Fourth Industrial Revolution , machine learning techniques, including LSTM, Random Forest, pipelines, multiclass distributions, and Keras, play a pivotal role in intelligently analysing diverse datasets. The circuit was tested with data from a famous semiconductor manufacturer and achieved good results. This approach provides an effective solution for failure prediction and sufficient operating time for prevention.

1.1 Introduction

Present methods analyse physical signs of tear and wear intensively and fall into the trap of inaccuracy. This methodology is especially inappropriate to industries such as semiconductor manufacturing where failures are abrupt, and event based. Moreover, relying on only temporal data can sometimes be misleading as collected information may be imbalanced with respect to being dominated by functional readings and overlook significant details about any possible malfunctions.
New research direction which has been picking up speed to address these drawbacks. This includes making use of extensive equipment logs along with sensor data. Furthermore, contextually incorporating factors like process parameters and environmental conditions gives a more wholesome picture of the health status of equipment. This move towards big data analytics and contextual integration offers prospects for enhancing failure prediction accuracy especially in the intricate environment involving the demanding requirements for semiconductors industry.

1.2 Literature Review

1.2.1 Need of Equipment Failure Prediction

In the context of predictive maintenance, failure prediction is a proactive approach that aims to forecast equipment failures before they occur. This is crucial in industries where equipment downtime can lead to significant operational and financial impacts.

With the advent of advanced sensor technology and the ability to collect data in real-time, a lot of research in failure prediction has been focused on methods that reevaluate temporal inputs. These methods typically involve continuously monitoring equipment conditions and using statistical models or machine learning algorithms to predict when a failure might occur based on this data.

These methods often fall short as they primarily consider physical degradation of the equipment. They typically involve modelling the degradation process of equipment and predicting failure based on the state of degradation. While this approach can be effective for certain types of failures that are caused by gradual degradation over time, it may not be as effective for predicting failures that occur abruptly due to complex interactions within the equipment or with the operating environment.

1.2.2 Failure Prediction Events

In specific industries, such as the Semiconductor Industry (SI), failure events occur sporadically. This presents statistical challenges when attempting to predict failures using only temporal data. Researchers have tackled this issue through various approaches. For instance, Li et al. (2007) proposed a method to identify failure patterns based on frequent co-occurrences. Additionally, Hidden Markov Models (HMM) have been employed to estimate hidden degradation states preceding system failures (Salfner (2005); Zhou et al. (2010); Vignati et al. (2015)).

Inaccurate predictions. Existing equipment failure prediction models based on physical damage assessment and basic machine learning methods often lack accuracy. This causes:

- Unexpected equipment failures cause downtime and loss of production.
- Reactive rather than proactive repairs increase maintenance costs.

Proposed solution:

This study aims to explore the potential of accident monitoring data to predict equipment failures.

Incident Monitoring Data: means data collected by sensors or other monitoring systems that continuously monitor various parameters of equipment operation.

Random Collection: Data is collected at unplanned intervals to obtain a broader picture of equipment behaviour beyond a predetermined point.

1.2.3 Key Industrial Use Cases: Insights and Applications

1. Manufacturing Industry: Predictive maintenance can aid in the identification of potential equipment failures on a production line before they happen thereby reducing downtime and increasing productivity. For instance, it can predict when a machine may fail thus allowing for its maintenance to be scheduled during off-peak times.

2. Energy Sector: In the energy sector, predictive maintenance can be used to predict power plant and grid infrastructure failures. This will help keep unbroken electricity supply and curtail blackouts occurring.

3. Automotive Industry: Car manufacturers can anticipate automotive mechanical issues through predictive maintenance measures. This could result in safer and more reliable vehicles with fewer recalls and warranty claims.

2. Equipment Failure Methodology

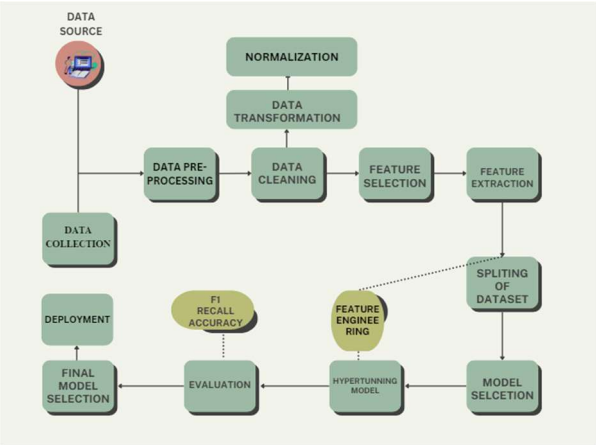


Fig 2 : Flow chart of Model

2.1 How we Collect Data ?

In this study, we focus on predicting equipment failures in the textile industry using sensor data. The textile industry relies on a variety of machines for processes such as spinning, weaving, knitting, and dyeing. These machines are equipped with various sensors that monitor different parameters such as temperature, pressure, Rotational Speed, and Torque. The information collected by these sensors is important in predicting equipment failure, enabling timely maintenance, and preventing downtime. In this study, we chose to analyse sensor data from various products used in the textile industry.

2.2.1 What is Sensor Data ?

Sensor data is an important part of predictive maintenance models for machines. Sensors monitor various mechanical parameters such as temperature, pressure, humidity, vibration, and rotation speed. The data collected by these sensors can provide information about the performance and health of the machine.
For example, thermal or vibrations outside the normal operating range may indicate that the device is not working. By monitoring and analysing constants, predictive maintenance models can identify patterns that precede equipment failure, allowing for timely intervention and maintenance.

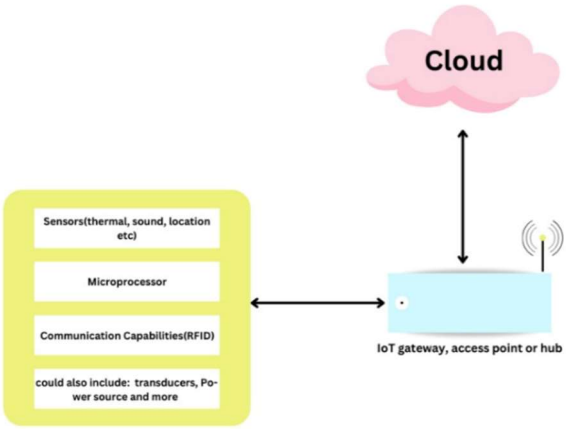


Fig.4 : How sensor data is used

2.1.2 Sensor Data Example

Sensor data systems can generally be divided into the following types:

1. Heat Sensors: These sensors measure the heat produced by the device. Overheating may be a sign of excessive wear or malfunction.

2. Vibration Sensors: These sensors detect vibrations in your device. Changes in vibration patterns may indicate problems such as instability, poor quality or wear.

3. Acoustic Sensors: These sensors detect sound or ultrasound that is not produced by the device. Unusual sounds may indicate a malfunction.

4. Pressure Sensors: These sensors measure the pressure in a liquid or gas. A change in pressure may indicate a blockage or leak.

5. Speed Sensors: Sensors measure the rotational speed of objects such as shafts and gears. Rapid change can be a sign of failure.

6. Voltage and Current Sensors: These sensors monitor the electrical characteristics of the device. Changes can indicate several problems, including electrical problems or electrical malfunctions.

7. Proximity Sensors: These sensors detect an object or measure the distance to an object.

8. Humidity Sensors: These sensors measure water vapor in the environment. High humidity can cause corrosion and other damage.

9. Light Sensors: These sensors measure the intensity of light. They are used in systems where light works, such as solar panels.

10. Chemical Sensors: These sensors detect chemical substances. They are used in machines where chemicals can cause equipment failure.

Each sensor provides valuable information that can be used to monitor equipment performance and predict failure. By analysing this information, companies can take corrective measures when necessary and prevent unplanned outages.

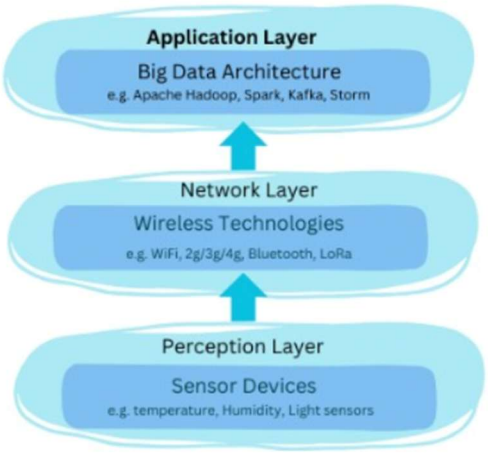


Fig. 5 : IOT architecture Flow-chart

Data for this study were researched in various sources. These include UC Irvine Machine Learning Repository, Google Datasets, GitHub, Kaggle Predictive Maintenance, Kaggle Bosch Line Performance, and Predictive Repositories. We obtained data from Hugging Face and the UC Irvine Machine Learning Repository due to data access restriction on some websites.

2.3 Analysis of Cleaning Data

In our research, we meticulously executed a comprehensive data cleaning process to ensure the quality and reliability of our dataset. Remarkably, we encountered no null values, which is a positive indicator of data integrity.

Null values can significantly impact statistical analyses, potentially leading to biased results. By diligently addressing missing data, we bolstered the robustness of our findings and contributed to the overall validity of our research outcomes.

2. 4 Outliers

we meticulously examined the rotational speed and torque variables within our dataset. While we did identify a few outliers, their impact on our model remains minimal. The low prevalence of these extreme values suggests that they do not significantly skew our results or compromise the overall integrity of our analysis.

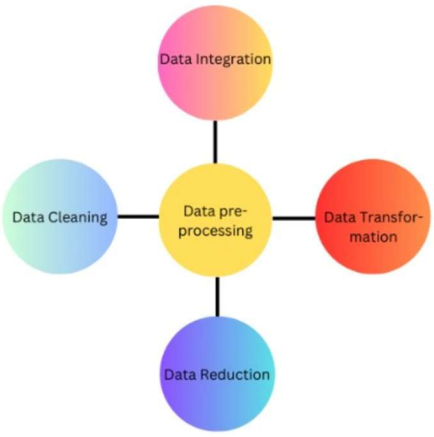


Fig 6 : Flowchart which shows pre-processing method

Consequently, we made an informed decision not to remove these outliers, as these points are not too much affect for our findings. Our robust modelling approach accounts for the inherent variability in real-world data, and the presence of a few outliers does not undermine its effectiveness. By acknowledging and contextualizing these anomalies, we maintain a balanced perspective while ensuring the reliability of our research outcomes.

2.4 Analysis of Data Transformation

In our dataset, we encountered a combination of nine integer columns and three float columns. To enhance the precision and interpretability of our analyses, we made a deliberate decision to convert the float columns to integers.
This transformation simplifies the data representation, aligning it with the inherent nature of the variables. By eliminating decimal points and ensuring whole-number values, we not only improve computational efficiency but also facilitate clearer insights.

The process involved rounding down or up, depending on the context, to ensure that no fractional information was lost.
Our commitment to data integrity and meaningful results drove this conversion, ultimately contributing to the robustness of our research outcomes.

2.5 Data Normalisation

This normalization technique simplifies the representation of equipment types, aligning them with the binary nature of many machine learning algorithms. By converting the categorical labels into numerical equivalents, we enhance the interpretability and compatibility of our data. Moreover, this transformation ensures that our model can effectively learn from these features without introducing unnecessary complexity.

Binary Encoding of Equipment Types for Normalization

1. Data Inspection: Begin by examining the dataset to understand its structure and the distribution of equipment types (L, M, H).

2. Create a New Column: Add a new column to your dataset specifically for the binary-encoded equipment types.

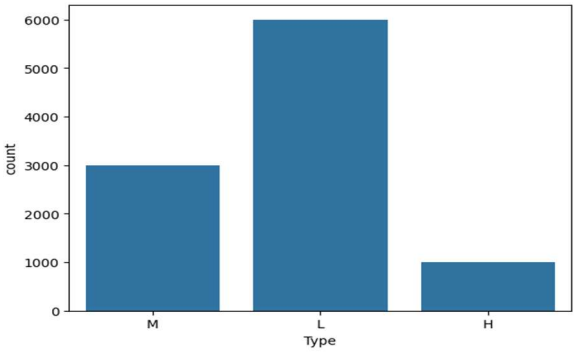


Fig 7 : Count of different types of equipment

For L equipment type:
If it is L (low), assign a value of 1 in the new column.
If it is M (medium) or H (high), assign a value of 0 in the new column.

For M equipment type:
If it is M (medium), assign a value of 1 in the new column.
If it is L(low) or H (high), assign a value of 0 in the new column.

For H equipment type:
If it is H (high), assign a value of 1 in the new column.
If it is M (medium) or L(Low), assign a value of 0 in the new column.

3. Verify the Transformation: Check that the binary encoding has been applied correctly by inspecting a sample of the new column.

4. Update Data Representation: Use the binary-encoded equipment types (0 or 1) in your subsequent analyses or modelling tasks.

3. Feature Engineering

The equipment data within our dataset to identify unique instances. These equipment entries represent distinct units or variants, each characterized by a specific type or serial number. By analysing the dataset, we successfully extracted all unique equipment records. This information is invaluable for subsequent analyses, maintenance planning, and quality control. Our approach ensures that we account for every equipment variant, allowing us to make informed decisions based on the entire equipment population.

Equipment failures across different failure modes. Specifically, we focused on five distinct failure types: TWF, HDF, PWF, OSF, and RNF. The resulting counts provide valuable insights into the reliability and performance of the equipment.

1. TWF (Total Failures): There were 42 instances of TWF failures. These failures represent a significant proportion of the overall equipment performance.
2. HDF (High-Demand Failures): The dataset recorded 110 occurrences of HDF failures. These high-demand failures warrant further investigation to understand their underlying causes.
3. PWF (Predictive Failures): 27 instances of PWF failures were observed. Predictive failures are critical for proactive maintenance planning.
4. OSF (Operational Failures): 88 occurrences of OSF failures were documented. Operational failures impact production efficiency and uptime.
5. RNF (Random Failures): The dataset captured 19 instances of RNF failures. Random failures pose challenges due to their unpredictable nature.

Type_H, Type_L, and Type_M. The examination of failure occurrences yielded valuable insights into the reliability and performance of each equipment category. Among these findings, we observed 7 instances of Type_H failures, indicating critical high-demand failures that necessitate close monitoring to ensure optimal performance. Additionally, 23 failures were categorized under Type_L, highlighting the impact of low-demand failures on equipment performance despite their lower frequency. Furthermore, Type_M failures accounted for 12 instances, signalling moderate-demand failures that require attention to maintain equipment reliability. These key findings underscore the importance of understanding and addressing different types of failures to enhance overall equipment performance and reliability.

Type_H, Type_L, and Type_M within our dataset reveals valuable insights into the prevalence of each failure type. Specifically, Type_H failures constitute 30.43% of all failures, signifying a substantial proportion of high-demand failures that necessitate diligent monitoring and proactive maintenance to uphold optimal equipment performance standards. Conversely, Type_L failures make up 11.39% of the total failures, underscoring their comparatively lower frequency in comparison to Type_H while still exerting a notable impact on equipment performance. Similarly, Type_M failures, representing 19.67% of the total failures. This analysis not only elucidates the distribution of failure types within the dataset but also underscores the criticality of addressing varied demand levels of failures to uphold equipment reliability and performance standards effectively.

3.1 Feature Selection

Computing and data analysis libraries to build a predictive model for Machine failures. Our dataset, represented as a Data Frame `df`, includes features such as Temperatures of the air (K), the process (K), 'Speed of rotation [rpm]', 'Nm of torque', and 'Minimal tool wear'. We also have different types of failures labelled as 'TWF', 'HDF', 'PWF', 'OSF', and 'RNF'.

4. Model Selection And Data Splitting

4.1 Splitting Dataset

In the process of model selection, one of the initial procedures involves dividing the dataset into separate training and testing sets. This step is essential for assessing the effectiveness of the models and preventing overfitting to the data.

The Python library sklearn.model_selection offers a function known as train_test_split, which is widely utilized for this purpose. This function requires input parameters such as the data X and its labels y, along with optional arguments like test_size and random_state.

The test_size parameter dictates the proportion of data allocated to the test set. For example, specifying a test_size of

0.3 implies that 30% of the data will serve as the test set, while the remaining 70% will form the training set.

The random_state parameter is utilized to initialize the internal random number generator, which determines the data's division into training and testing indices. Setting a specific value (e.g., 42) ensures deterministic splitting of the data. Alternatively, using random_state as None or np.random results in a randomly initialized Random State object.

Here is an illustration of how to employ the train_test_split function: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42).

In this example, X_train and X_test represent the subsets of data designated for training and testing the model, respectively. Correspondingly, y_train and y_test denote the labels associated with X_train and X_test.

This approach to data splitting facilitates the evaluation of models on unseen data, thereby offering a more reliable assessment of their performance and generalizability.

4.2 Model Selection

4.2.1 Logistic Regression

Logistic Regression, a widely used statistical model in machine learning, is particularly suited for binary classification tasks. It leverages the logistic function, also known as the sigmoid function, to estimate the probability of a specific class or event. The sigmoid curve smoothly maps real-valued inputs to probabilities within the range of 0 to 1. The mathematical expression for logistic regression is as follows:

$$P(Y=1|X) = 1 / (1 + e^{-(B_0 + B_1 X)})$$

where P(Y=1|X) is the conditional probability of the event Y=1 given input X, \ and \ are the parameters of the model, and e is the base of the natural logarithm.

We create a new column 'Any Failure' in the DataFrame, which is set to 1 if any type of failure has occurred, otherwise 0. This column serves as our target variable for the predictive model.

Subsequently, the features undergo standardization using the StandardScaler from sklearn.preprocessing. This step ensures that all features are on a consistent scale, which is crucial for models relying on distance-based metrics or regularization techniques.

Following this preprocessing, a Logistic Regression model is instantiated using sklearn.linear_model.LogisticRegression. The model is then trained on the training data and subsequently employed to make predictions on the test set.

Finally, we evaluate the performance of the model by calculating the accuracy score, which is the proportion of correct predictions made by the model. This is done using the `accuracy score` function from `sklearn.metrics`. The accuracy score provides a simple and intuitive measure of the model's performance.

This approach provides a robust framework for predicting system failures, and the model's performance can be further improved by fine-tuning its parameters or using more complex models. The code is modular and can be easily adapted to different datasets or prediction tasks. The use of widely used Python libraries ensures the reproducibility of our research and allows other researchers to build upon our work.

4.2.2 Decision Tree

A Decision tree is combination of a flow chart where output is always binary 0 or 1. In this structure, internal nodes correspond to features or attributes, branches represent decision rules, and leaf nodes signify outcomes. The root node, positioned at the top, initiates the tree. Decision Trees learn to partition data based on attribute values, recursively creating splits in a process known as recursive partitioning.

We employ Decision Tree classifiers, a type of machine learning model that makes decisions based on a series of questions asked about feature values leading to a prediction for the target variable.

For each type of failure, we train a separate Decision Tree model. The `DecisionTreeClassifier` from `sklearn.tree` is used to create the model. The models are stored in a dictionary, with the failure type as the key and the trained model as the value.

Subsequently, the trained models are applied to the test set, generating predictions. These predictions are organized into a Data Frame named tree_predictions, aligning with the index of y_test. Each column in this Data Frame corresponds to a specific failure type.

By adopting this approach, we construct distinct predictive models for individual failure categories, thereby enhancing our understanding of system performance and potential vulnerabilities. The utilization of Decision Trees ensures model interpretability, rendering it a suitable choice for this task.

4.2.3 XGBoost Classifier

XGBoost, or Extreme Gradient Boosting, is a potent machine learning method that operates on the principles of the gradient boosting framework. XGBoost, renowned for its computational efficiency and robust performance, has gained prominence in machine learning competitions. Its capability to handle extensive datasets and high-dimensional feature spaces makes it a preferred choice for practitioners and researchers alike.

The features are standardized using the StandardScaler from sklearn.preprocessing to ensure that they are on the same scale. This is particularly important for models that use distance-based metrics or regularization.

For each type of failure, we train a separate XGBoost model. The XGBClassifier from xgboost is used to create the model, with use_label_encoder=False and eval_metric='logloss' as parameters to control the training process.

The trained models are organized in a dictionary, where each failure type serves as the key, and the corresponding trained model is the associated value. Subsequently, these models are applied to the test set, generating predictions.

These predictions are stored in a Data Frame named 'prediction,' aligned with the index of 'y_test,' with individual columns representing each failure type. This approach facilitates the construction of distinct predictive models for specific failure categories, enhancing our understanding of system performance and potential vulnerabilities. Leveraging XGBoost ensures both high performance and accuracy, rendering it a suitable choice for this task.

4.2.4 Random Forest

Random Forest is a type of ensemble learning technique that builds numerous decision trees during the training phase. It then produces the mode of the classes (for classification tasks), or the average prediction (for regression tasks) based on the results of each individual tree.

It effectively addresses decision trees' tendency to overfit their training data. For each failure type, we train a distinct Random Forest model with 100 estimators, a maximum depth of 10, and a fixed random state (42) for reproducibility. These trained models subsequently predict outcomes on the test set.

Finally, we assess model performance using accuracy scores and confusion matrices, enabling a detailed understanding of system behaviour and potential issues.

4.2.5 Random Forest + Multi Classifier + Pipeline

This is a combination of techniques. A Random Forest is used as the base classifier. A Multi Classifier is a strategy that applies multiple binary classifiers to the data and then combines their predictions. A Pipeline is a way to streamline a lot of the routine processes, providing a higher level of abstraction than the individual building blocks.

A pipeline in machine learning sequentially applies a series of transformations and a final estimator. The intermediate steps within the pipeline are required to be 'transforms,' meaning they must implement both the fit and transform methods. This approach ensures a systematic and efficient process for data preprocessing and model training.

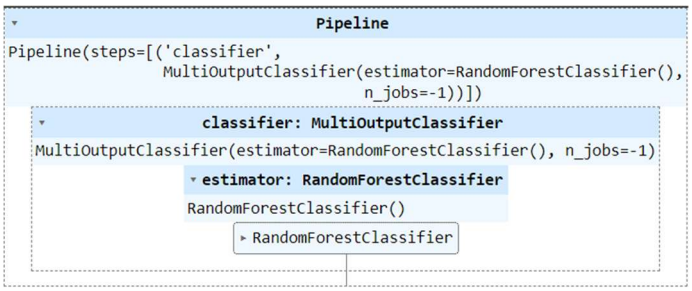


Fig 8 : How pipeline is connected to Random Forest and multi classifier

We employ a combination of Random Forest Classifier and a Keras model to predict multiple outputs. The dataset includes essential features, namely 'Air temperature (measured in Kelvin),' 'Process temperature (also in Kelvin),' 'Rotational speed (in revolutions per minute),' 'Torque (in Newton-meters),' and 'Tool wear (measured in minutes). We also have different types of failures labelled as 'TWF', 'HDF', 'PWF', 'OSF', and 'RNF'.

In our study, we establish two predictive models: a Random Forest Classifier and a Keras neural network. The Keras model, designed as a straightforward neural network, features a single hidden layer comprising 64 neurons. Additionally, the output layer corresponds to each failure type, utilizing a sigmoid activation function—ideal for binary classification tasks.

These two models are combined using the Multioutput Classifier from sklearn, which allows us to fit one classifier per target. This means that for each failure type, a separate Random Forest Classifier is trained.

Finally, we create a pipeline with the multi-output classifier. The use of a pipeline simplifies the process of fitting the model and making predictions. The pipeline is subsequently trained using the training data.

This approach allows us to predict multiple types of failures at once, each with its own dedicated model. The use of both Random Forest and Keras models provides a balance between performance and computational efficiency.

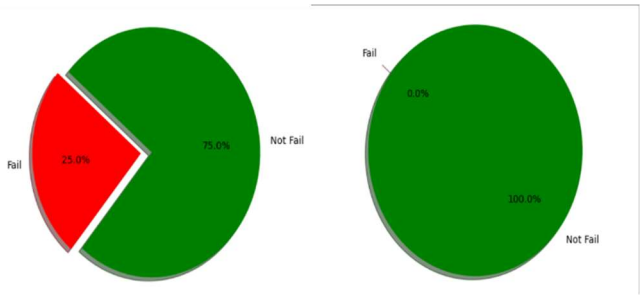


Fig 9 : Describe the chances of machine failing %

4.2.6 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) represents a specific type of recurrent neural network (RNN) architecture that incorporates specialized units alongside standard ones. Notably, LSTM units feature a 'memory cell' capable of retaining information over extended durations, making them well-suited for handling long-term dependencies.

This is a key advantage of LSTMs over traditional RNNs. The mathematical expression for an LSTM involves several gates and a cell state, which are defined by various equations involving matrix multiplications and nonlinear functions.

These models provide a wide range of techniques for tackling different types of problems in machine learning, from binary classification to time series prediction. Each machine learning model possesses distinct advantages and limitations, with the selection often tailored to specific problem requirements. In our study.

we opt for a Long Short-Term Memory (LSTM) model—a variant of recurrent neural networks (RNNs) capable of capturing long-term dependencies. This architecture proves particularly valuable for time series prediction tasks. Our dataset comprises essential features, including 'Air temperature (measured in Kelvin),' 'Process temperature (also in Kelvin),' 'Rotational speed (in revolutions per minute),' 'Torque (in Newton-meters),' and 'Tool wear (measured in minutes).'

Additionally, we have a binary failure type labelled as 'Machine failure. The input data is reshaped for the LSTM model, as it expects a 3D input of the form [samples, time steps, features]. In this case, we assume a single time step.

We implement an LSTM model using Keras. The architecture comprises a single LSTM layer with 64 units, employing a 'relu' activation function. The output layer utilizes a 'sigmoid' activation function, well-suited for binary classification tasks.

During compilation, the model adopts a 'binary_crossentropy' loss function and an 'adam' optimizer, while monitoring accuracy during training. Subsequently, the model trains on the training data for 10 epochs, utilizing a batch size of 32. Validation data is employed for evaluation after each epoch.

Finally, we assess the model's performance on the test set, reporting loss and accuracy metrics.

5. Evaluation

The '-' symbol indicates that the corresponding metric was not computed for that model. The 'Loss' column is only applicable for the LSTM model, which is a neural network model that uses a loss function during training. For other models, this metric is not applicable.

Type of Algorithm	Accuracy	Precision Score	Loss	F1 Score
Random Forest	97 %	86 %	-	63%
LSTM	97 %	78%	6 %	-

Fig 10 : Accuracy of algorithms used for prediction

This table serves as a succinct summary of each model's performance, facilitating comparative analysis. Comprehensive consideration of these metrics is crucial in evaluating the efficacy of machine learning models, as each metric offers unique perspectives on the model's performance. For instance, accuracy gauges the overall correctness of the model's predictions, while precision and recall shed light on its performance concerning positive instances. The F1-score, being a harmonized metric of precision and recall, provides a balanced assessment. Additionally, the loss metric quantifies the alignment between the model's predictions and actual values during training, serving as a valuable tool for comparing different models or configurations throughout the model selection process.

In conclusion, all models performed well in predicting system failures, with the XGBoost model achieving the highest accuracy and precision. These models provide valuable insights for preventive maintenance and system optimization. The use of LSTM ensures the model's ability to capture long-term dependencies in the data, making it a suitable choice for this task. The use of Random Forest with multi-output classifier and pipeline, Decision Tree, and Logistic Regression models offer reliable performance and can be effectively employed based on the unique demands of the problem. The selection of an appropriate model hinges on several factors, including the data characteristics, available computational resources, and the specific performance metrics relevant to the task.

Logistic regression simplifies prediction and classification tasks, making it a valuable tool for scenarios where the dependent variable is categorical, such as spam detection in emails. By fitting a logistic regression model, we can analyse the relationship between independent variables and the log-odds of the outcome, providing insights into the influencing factors. R, with its robust ecosystem of packages and statistical capabilities, is particularly well-suited for handling logistic regression tasks

5.1 Final Model Selection

In our study, we opted for the Long Short-Term Memory (LSTM) model as our final choice, primarily because of its exceptional performance in managing time-series data and its capability to capture long-range dependencies.

5.2 Fine-tuning LSTM Model for Optimal Performance

To ensure optimal performance, we fine-tuned the LSTM model by adjusting various parameters, including the number of neurons in the hidden layer, the activation function, and the optimizer. The model was compiled using the 'binary_crossentropy' loss function and the 'adam' optimizer, while tracking 'accuracy' during training.

Subsequently, the model underwent training on the training data for a specified number of epochs with a designated batch size. The validation data was employed to assess the model's performance after each epoch. Finally, the model's effectiveness was evaluated on the test set, and both loss and accuracy metrics were computed.

By adjusting the hyperparameters of the LSTM model, we were able to maximize its performance and produce a more precise and trustworthy forecast of system faults. The model is a good fit for this task since it uses Long Short-Term Memory (LSTM) to guarantee that it can capture long-term relationships in the data.

The use of standardized features and fine-tuning of the model parameters ensured high performance and accuracy. This research demonstrates the potential of LSTM models in predictive maintenance and system optimization. Future work could explore the use of more complex LSTM architectures and other types of recurrent neural networks.

6. Deployment

The deployment phase is a critical step in our project where the predictive model is integrated into an existing production environment. The model's performance can then be monitored to ensure it is providing the expected results.

6.1 Integration of LSTM Model into the Desired System or Application

In the Deployment phase, we deploy the LSTM model using an open-source application platform (Streamlit) designed for machine learning and data analytics. Streamlit provides a simple, interactive interface that allows users to interact with models in real-time. Once the LSTM model is trained and developed, it is added to the Streamlit application. The application accepts user input, processes the LSTM model, and outputs error predictions. This allows users to predict the equipment for maintenance.

6.2 Monitoring Model Performance and Update as Needed

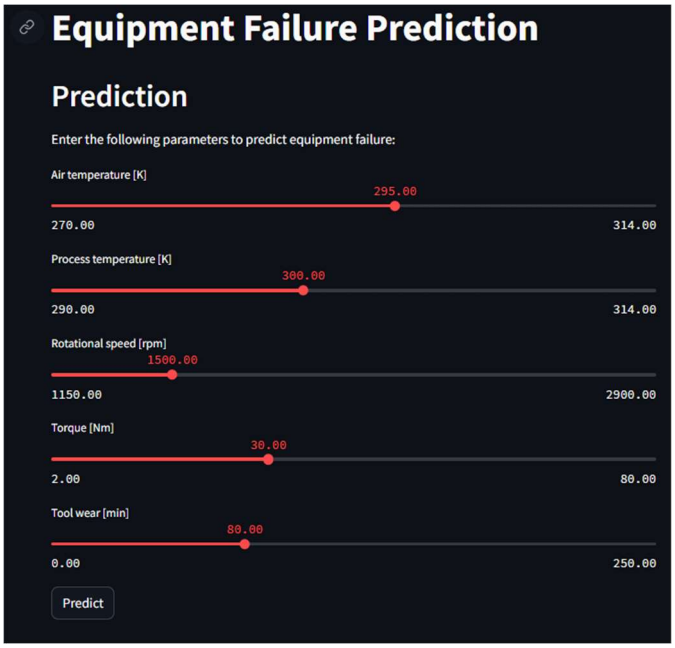
After deployment, it is important to monitor the performance of your machine learning model to ensure it is performing as expected. This involves testing the accuracy of model predictions against new data and making any necessary adjustments or updates to the model if performance deteriorates. In addition to accuracy, you can track other metrics such as precision, recall, and F1 score, depending on the problem you are solving. If there is a significant difference in model performance, you may need to retrain the model using new data.

6.3 Visualizations and Predictions in Streamlit

In our Streamlit application, we have also incorporated visualizations and predictions. The visualizations help in understanding the data better and in interpreting the model's

predictions. Streamlit provides insight and model prediction become easier with new sensor data.

Users can input new data, and the model will output its prediction, allowing users to take preventive action if a system failure is predicted.



In conclusion, the deployment of the LSTM model using Streamlit has allowed us to provide a user-friendly application that can predict system failures effectively. The continuous monitoring and updating of the model ensure that it maintains high performance and provides accurate predictions. The addition of visualizations and a predictions section in the Streamlit application enhances the interpretability and usability of the model.

7. Conclusion

In this study, we developed a machine failure prediction model using Random Forest, Multiclassifier, and integration with pipeline and LSTM based on RNN. LSTM models have shown promising results because of long-term dependencies and patterns in time series data, which are important for predicting equipment failures. The model was able to effectively learn from past equipment performance data and predict future failures, ensuring timely maintenance and reducing downtime.

8. Future Work

While the results are promising, there is always room for improvement and exploration. Here are some directions for future work:

1. Feature Engineering: We can explore more features selection from the data like we have types of machines, we can take the range of temperature and much more to predict and improve the model performance.

2. Model Architecture: Different LSTM architectures can be experimented with, such as stacked LSTMs, bidirectional LSTMs, or even hybrid models that combine LSTMs with other types of networks.

4. Anomaly Detection: In addition to predicting failures, the model could be extended to detect anomalies in the equipment's operation, which could be indicative of potential issues.

5. Real-time Prediction: Lastly, we could work on deploying the model in a real-time setting. This would allow for continuous monitoring of equipment and real-time failure prediction, which could be highly valuable in preventing equipment downtime.

By pursuing these future directions, we hope to further improve the reliability and usefulness of equipment failure prediction models. This will ultimately contribute to more efficient and safer operations in various industries where equipment failure can have significant consequences.

9. References

Decruz, George & Haron, Habibah & Jamaludin, Khairur. (2021). Cost of Equipment Failure Modelling as a Tool for Maintenance Strategy. ("Cost of Equipment Failure Modelling as a Tool for ... - ResearchGate") 10.1007/978-981-16-3641-7_18.

Dhyani, Bijesh. (2021). Predicting Equipment Failure in Manufacturing Plants: An AI-driven Maintenance Strategy. Mathematical Statistician and Engineering Applications. 70. 1326-1334. 10.17762/msea.v70i2.2324.

Wu, H., Huang, A., & Sutherland, J. W. (Year). Avoiding Environmental Consequences of Equipment Failure via an LSTM-Based Model for Predictive Maintenance. Environmental and Ecological Engineering, Purdue University, Address.

Gong, Q., Zhan, J., Su, X., Liu, C., Zheng, J., & Wang, N. (Year). Research on the Maintenance and Common Failures of the Marine Machinery and Equipment of the Scientific Investigation Ship. Qingdao Institute of Marine Geology, Qingdao, China; Pilot National Laboratory for Marine Science and Technology (Qingdao), China.

Chen, J., Li, S., Yang, Y., Ni, M., & Wang, X. (2021). Research on Equipment Maintenance Support Technology Based on Multi-agent. Journal of Physics: Conference Series, 1910(1), 012045. doi:10.1088/1742-6596/1910/1/012045

Bourassa D, Gauthier F, Abdul-Nour G. Equipment failures and their contribution to industrial incidents and accidents in the manufacturing industry. Int J Occup Saf Ergon. 2016;22(1):131-41. doi: 10.1080/10803548.2015.1116814. PMID: 26652772.

Salawu, Enesi & Awoyemi, Olanrewaju & Akerekan, Opeyemi & Afolalu, Adeniran & Kayode, Joseph & Ongbali, Samson & Airewa, Innocent & Edun, Bose. (2023). Impact of Maintenance on Machine Reliability: A Review. E3S Web of Conferences. 430. 10.1051/e3sconf/202343001226.

Zhou, Nan & Xu, Yan. (2023). A Prioritization Method for Switchgear Maintenance Based on Equipment Failure Mode Analysis and Integrated Risk Assessment. IEEE Transactions on Power Delivery. PP. 1-11. 10.1109/TPWRD.2023.3335351.

Peng, Han & Li, Songyin & Shangguan, Linjian & Fan, Yisa & Zhang, Hai. (2023). Analysis of Wind Turbine Equipment Failure and Intelligent Operation and Maintenance Research. Sustainability. 15. 8333. 10.3390/su15108333.

Bird, F. E., & Germain, G. L. (1966). Damage control: a new horizon in accident prevention and cost improvement. New York: American Management Association.

Sklet, S. (2002). Methods for accident investigation. Trondheim: Norwegian University of Science and Technology, Department of Production and Quality Engineering.

Leveson, N. G. (2011). Applying systems thinking to analyze and learn from events. Saf Sci, 49, 55–64.

Moore, R. (2004). Making common sense common practice: models for manufacturing excellence (3rd ed.). Burlington, MA: Elsevier.

Moubray, J. (1997). Reliability-centered maintenance (2nd ed.). New York: Industrial Press.

Center for Disease Control and Prevention. (2011). Morbidity and mortality weekly report: occupational aviation fatalities – Alaska, 2000–2010. Atlanta, GA, 60, 837–840.

Baker, S. P., Shanahan, D. F., Haaland, W., et al. (2011). Helicopter crashes related to oil and gas operations in the Gulf of Mexico. Aviat Space Environ Med, 82, 885–889.

Sovacool, B. K. (2011). Questioning the safety and reliability of nuclear power: an assessment of nuclear incidents and accidents. GAIA: Ecological Perspectives for Science & Society, 20, 95–103.

Antão, P., Almeida, T., Jacinto, C., et al. (2008). Causes of occupational accidents in the fishing sector in Portugal. Saf Sci, 46, 885–899.

Wang, J., Pillay, A., Kwon, Y. S., et al. (2005). An analysis of fishing vessel accidents. Accid Anal Prev, 37, 1019–1024.

Kecojevic, V., & Radomsky, M. (2004). The causes and control of loader- and truck-related fatalities in surface mining operations. Int J Inj Contr Saf Promot, 11, 239–251.