International journal of science, engineering, and information technology
Volume 09, Issue 02, July 2025
Journal homepage: https://journal.trunojoyo.ac.id/ijseit

Improving Root Cause Analysis of Production Defects Using AI: A Case Study in An Automotive Manufacturing Plant

Muhammad Najib, Emon Rifa'i

Departement of Informatics, Universitas KH. Bahaudin Mudhary Madura, Sumenep, Indonesia

A B S T R A C T

In automotive manufacturing, repetitive defects often occur across different time periods, creating a valuable historical dataset containing defect names and their corresponding root causes. Traditionally, identifying the root cause of a production defect relied heavily on human analysis, requiring significant time and on-site inspection. This often led to delayed countermeasures, increased production downtime, and additional issues such as line stops. This study presents an AI-based approach to assist root cause analysis using historical defect data, aiming to reduce the analysis time and improve feedback accuracy. The implementation focused on enabling faster and more accurate identification of root causes by integrating a machine learning model into the factory's defect recording system (ATPPM, *Analisa Tindakan Penanggulangan dan Pencegahan Masalah*). The development process involved data preprocessing, model training, and API deployment. The original dataset consisted of 3,128 records, which were cleaned and reduced to 1,449 labeled entries, each annotated with one of 161 unique root cause labels. Eleven machine learning models were evaluated, including Logistic Regression, Random Forest, SVM, and RNN. Initial evaluation using F1-score, precision, and recall showed Logistic Regression achieving the best F1-score of 0.83. Further validation using 5-Fold Cross Validation identified the Support Vector Machine (SVM) as the best-performing model, with an average accuracy of 89.1%. This model was deployed via a Python Flask API and integrated into the existing ATPPM system. The AI-powered system significantly accelerated the root cause analysis process, reducing the average analysis time by 228 minutes. Potential future enhancements involve automating the model's training process on a regular schedule (weekly or daily), integrating additional data sources including big data and quality management systems, and scaling the current API implementation to multiple production lines for wider impact.

Keywords: Root Cause Analysis, Production Defect, Machine Learning, Defect Prediction.

Article History
Received 02 April 25 Received in revised form 12 May 25 Accepted 29 June 25

1. Introduction

In the automotive manufacturing industry, maintaining high product quality and operational efficiency is critical. Production defects, even minor ones, can lead to significant consequences, including increased rework, production line stops, and customer dissatisfaction. Root Cause Analysis (RCA) is a key method used to identify the underlying causes of defects and to implement effective corrective actions. However, traditional RCA relies heavily on human expertise and manual investigation, which often results in delayed responses, inconsistent results, and increased operational costs [7]. In many manufacturing environments, particularly in developing regions, operators face difficulty in determining the root causes of production defects due to limited analytical tools and time constraints. Although manufacturing systems often record defect histories, these datasets are typically underutilized. In such cases, repetitive defects with previously known solutions continue to consume resources and time during reanalysis. This study addresses that gap by applying Artificial Intelligence (AI) to leverage historical defect data and assist in predicting root causes faster and more accurately.

The emergence of AI and Machine Learning (ML) techniques has enabled predictive analytics to be applied effectively in various industries, including manufacturing [15]. Predictive models can process large volumes of past data, extract meaningful patterns, and support real-time decision-making, particularly in defect classification, anomaly detection, and process optimization [18][20]. Prior works have shown that ML techniques such as Support Vector Machines (SVM), Random Forest, and Neural Networks are effective in defect detection and classification tasks [3][10]. However, limited research exists on using AI specifically for root cause prediction in the automotive sector.

This research proposes the development and deployment of an AI-based system to support root cause analysis in a car manufacturing plant in Indonesia. By training machine learning models on historical data consisting of defects and their confirmed root causes, the system aims to assist production operators by providing automated predictions for new defect entries. This AI assistance reduces analysis time significantly and improves the consistency and accuracy of feedback given during problem-solving processes.

The objectives of this study are twofold: (1) to reduce the time required for analyzing and acting on production defects by using AI predictions, and (2) to build a scalable AI architecture that can be integrated with existing systems and potentially expanded across other production lines. This paper also presents a comparison of multiple ML models, highlighting the most effective one for root cause prediction, and discusses the practical

* *Corresponding author.* Phone : +6285645123565
E-mail address: muhammadnajib@unibamadura.ac.id

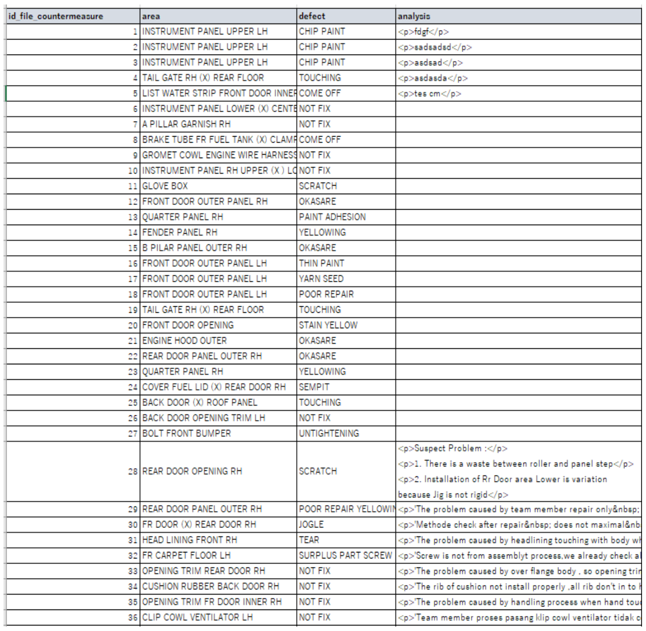
implementation of the AI model within the factory's production environment.

2. Materials and Methods

2.1. Materials

The primary material used in this study is a dataset extracted from the defect reporting system of a production line from automotive manufacturing plant in Indonesia. The dataset contains historical records of production defects, including the defect name, its corresponding area, and the analyzed root cause. This dataset is described in Table 1.

Table 1. Dataset of defect and root cause historical record.



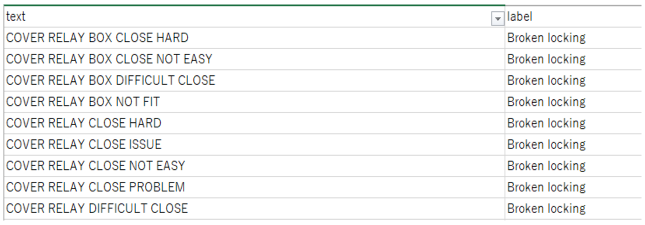
id_file_countermeasure	area	defect	analysis
1	INSTRUMENT PANEL UPPER LH	CHIP PAINT	<p>fdgf</p>
2	INSTRUMENT PANEL UPPER LH	CHIP PAINT	<p>sadsadsd</p>
3	INSTRUMENT PANEL UPPER LH	CHIP PAINT	<p>asdsad</p>
4	TAIL GATE RH (X) REAR FLOOR	TOUCHING	<p>asdasda</p>
5	LIST WATER STRIP FRONT DOOR INNER	COME OFF	<p>tes cm</p>
6	INSTRUMENT PANEL LOWER (X) CENTE	NOT FIX	
7	A PILLAR GARNISH RH	NOT FIX	
8	BRAKE TUBE FR FUEL TANK (X) CLAMP	COME OFF	
9	GROMET COWL ENGINE WIRE HARNESS	NOT FIX	
10	INSTRUMENT PANEL RH UPPER (X) LO	NOT FIX	
11	GLOVE BOX	SCRATCH	
12	FRONT DOOR OUTER PANEL RH	OKASARE	
13	QUARTER PANEL RH	PAINT ADHESION	
14	FENDER PANEL RH	YELLOWING	
15	B PILAR PANEL OUTER RH	OKASARE	
16	FRONT DOOR OUTER PANEL LH	THIN PAINT	
17	FRONT DOOR OUTER PANEL LH	YARN SEED	
18	FRONT DOOR OUTER PANEL LH	POOR REPAIR	
19	TAIL GATE RH (X) REAR FLOOR	TOUCHING	
20	FRONT DOOR OPENING	STAIN YELLOW	
21	ENGINE HOOD OUTER	OKASARE	
22	REAR DOOR PANEL OUTER RH	OKASARE	
23	QUARTER PANEL RH	YELLOWING	
24	COVER FUEL LID (X) REAR DOOR RH	SEMPIT	
25	BACK DOOR (X) ROOF PANEL	TOUCHING	
26	BACK DOOR OPENING TRIM LH	NOT FIX	
27	BOLT FRONT BUMPER	UNTIGHTENING	
28	REAR DOOR OPENING RH	SCRATCH	<p>Suspect Problem :</p> <p>1. There is a waste between roller and panel step</p> <p>2. Installation of Rr Door area Lower is variation because Jig is not rigid</p>
29	REAR DOOR PANEL OUTER RH	POOR REPAIR YELLOWIN	<p>'The problem caused by team member repair only
30	FR DOOR (X) REAR DOOR RH	JOGLE	<p>'Methode check after repair does not maximal&nb
31	HEAD LINING FRONT RH	TEAR	<p>'The problem caused by headlining touching with body wh
32	FR CARPET FLOOR LH	SURPLUS PART SCREW	<p>'Screw is not from assemblyt process,we already check al
33	OPENING TRIM REAR DOOR RH	NOT FIX	<p>'The problem caused by over flange body , so opening trin
34	CUSHION RUBBER BACK DOOR RH	NOT FIX	<p>'The rib of cushion not install properly ,all rib don't in to h
35	OPENING TRIM FR DOOR INNER RH	NOT FIX	<p>'The problem caused by handling process when hand tou
36	CLIP COWL VENTILATOR LH	NOT FIX	<p>'Team member proses pasang klip cowl ventilator tidak c

Initially, 3,128 records were collected from the system's database. However, due to the presence of inconsistent, duplicated, or irrelevant entries, a data cleaning process was conducted. This involved:

- Data Deduplication
- Manual Annotation and grouping similar defect/root cause terms
- Stopword Removal and Text Normalization

After cleaning, the resulting dataset consisted of 1,449 valid records, annotated with 161 unique root cause labels, each label having at least 9 records. The final dataset is described in Table 2 that simplified to Text and Label columns. Text column contains combination of Area and Defect Name, and label contains the root cause category as manual annotation. This final dataset was split into 70% for training and 30% for testing.

Table 2. Final dataset of defect and root cause group (1 label).



text	label
COVER RELAY BOX CLOSE HARD	Broken locking
COVER RELAY BOX CLOSE NOT EASY	Broken locking
COVER RELAY BOX DIFFICULT CLOSE	Broken locking
COVER RELAY BOX NOT FIT	Broken locking
COVER RELAY CLOSE HARD	Broken locking
COVER RELAY CLOSE ISSUE	Broken locking
COVER RELAY CLOSE NOT EASY	Broken locking
COVER RELAY CLOSE PROBLEM	Broken locking
COVER RELAY DIFFICULT CLOSE	Broken locking

2.2. Machine Learning Models

In the implementation phase of this study, several established machine learning models were selected to address the defect classification problem. These models have been widely applied in related research domains. The following section provides a concise overview of each model:

2.2.1. K-Nearest Neighbors (KNN)
A non-parametric, instance-based learning algorithm that classifies a data point based on the majority class of its k-nearest neighbors in the feature space [1].

2.2.2. XGBoost (Extreme Gradient Boosting)
A scalable, tree-based ensemble method that uses gradient boosting techniques for classification and regression tasks, known for its performance in structured data problems [5].

2.2.3. LightGBM (Light Gradient Boosting Machine)
A highly efficient gradient boosting framework that uses histogram-based algorithms, enabling faster training and lower memory usage than XGBoost [12].

2.2.4. Decision Tree
A tree-structured model where each internal node represents a feature condition and each leaf node represents a class label. It is interpretable and simple to implement [16].

2.2.5. Random Forest
An ensemble of decision trees trained on different subsets of the data. It improves accuracy and reduces overfitting by aggregating predictions [4].

2.2.6. Naïve Bayes
A probabilistic classifier based on Bayes' theorem with the assumption of feature independence. It performs well on high-dimensional data [8].

2.2.7. Linear Discriminant Analysis (LDA)
A classification technique that models the difference between classes using linear combinations of features, assuming Gaussian distributions [9].

2.2.8. Support Vector Machine (SVM) Linear
A supervised learning algorithm that finds the optimal hyperplane to separate classes using a linear decision boundary [6].

2.2.9. Support Vector Machine (SVM) Non-Linear (RBF Kernel)
Uses the Radial Basis Function (RBF) kernel to map input data to a higher-dimensional space for classification of non-linearly separable data [6].

2.2.10. Logistic Regression

A statistical model used for binary classification tasks that estimates the probability of a class label using a logistic function [11].

2.2.11. Recurrent Neural Network (RNN)

A type of neural network suitable for sequential data, capable of retaining memory of previous inputs using hidden states [14].

2.3. Evaluation Metrics

As this study addresses a real-world quality control problem in a manufacturing environment, it is essential to ensure that the machine learning models not only achieve high overall accuracy, but also balance precision and recall. For this reason, appropriate evaluation metrics are applied as described below.

2.3.1. Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positives. It is a measure of the model's exactness, as formula mentioned in Equation 1 that TP stands for True Positive, and FP stands for False Positive.

$$\text{Precision} = \frac{TP}{TP+FP} \tag{1}$$

2.3.2. Recall (Sensitivity)

Recall is a measure of the model's completeness. It is the ratio of correctly predicted positive observations to all actual positives, so that it's include FN (False Negative) in the formula that express in Equation 2.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2}$$

2.3.3. F1-Score

F1-Score is the harmonic mean of Precision and Recall. It balances both metrics and is useful in imbalanced class scenarios. The formula is described in Equation 3.

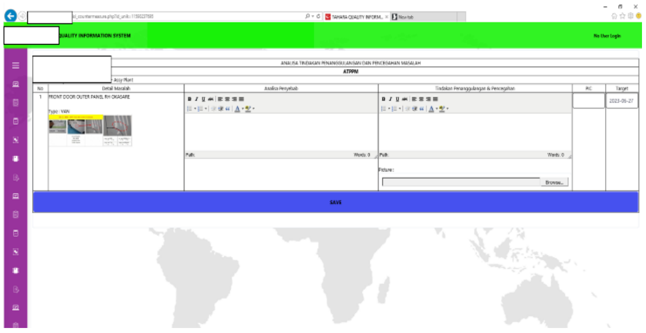
$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

2.3.4. K-Fold Cross Validation

A resampling technique used to evaluate models by dividing the data into *k* equal-sized folds. Each fold is used once as the test set, while the remaining *k−1* folds are used for training. The final performance is the average of the scores across all *k* iterations [13].

2.4. System and Tools

The AI model was built using *Python* programming language, leveraging the Flask framework to develop a lightweight REST API. The API was deployed on a Windows Server environment to integrate directly with the factory's existing ATPPM (*Analisa Tindakan Penanggulangan dan Pencegahan Masalah*) system, whose interface is shown in Figure 1. The ATPPM system records analysis reports of production defects, including the corrective and preventive actions taken. This system has been in operation for the past 3 years on the production line that serves as the object of this study.



ANALISA TINDAKAN PENANGGULANGAN DAN PENCEGAHAN MASALAH

ATPPM

Assy Plant

No	Detail Masalah	Analisa Penyebab	Tindakan Penanggulangan & Pencegahan	PIC	Target
1	FRONT DOOR OUTER PANEL RH OKASARE Type : VAN	Path: Words: 0	Path: Words: 0 Picture: Browse...		2023-06-27

Figure 1: The Interface of ATPPM system to input Root Cause Analysis of Defect.

The system's Natural Language Processing (NLP) components were developed using the NLTK library, focusing on basic preprocessing tasks such as tokenization, stopword removal, and basic synonym handling to handle variations in defect/root cause descriptions [2].

2.5. Methodology

The methodology for this research is explained in Figure 2, that involved the following stages:

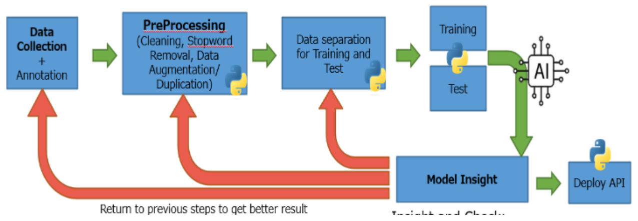

Data Collection + Annotation
PreProcessing (Cleaning, Stopword Removal, Data Augmentation/ Duplication)
Data separation for Training and Test
Training
Test
AI
Model Insight
Deploy API
Return to previous steps to get better result

Figure 2: Methodology of AI Implementation Research for Root Cause Analysis in Manufacturing Plant.

2.5.1. Data Collection & Annotation

Defect and root cause data were collected and manually annotated into meaningful categories.

2.5.2. Data Preprocessing, Data Augmentation

Standard NLP preprocessing was applied, including cleaning, tokenization, and manual correction for mislabeled or ambiguous terms [19]. Data Augmentation is needed to address the issue of data imbalance, data duplication was used for underrepresented labels.

2.5.3. Data Separation for Training and Test

The targeted dataset was split into 70% for training and 30% for testing.

2.5.4. Model Training & Evaluation (Model Insight)

Eleven popular classification models were selected for benchmarking: K-Nearest Neighbors (KNN), XGBoost, LightGBM, Decision Tree, Random Forest, Naïve Bayes, Linear Discriminant Analysis (LDA), Support Vector Machine (SVM) Linear, SVM Non-Linear (using RBF Kernel), Logistic Regression, and Recurrent Neural Network (RNN). These models were evaluated using Precision, Recall, and F1-score metrics [17]. Additionally, a K-Fold Cross Validation was performed to validate model consistency and avoid overfitting.

2.5.5. Model Selection & Deployment

Multiple machine learning models were trained and evaluated using consistent metrics such as precision, recall, and F1-score. To ensure performance stability and avoid overfitting, 5-Fold Cross Validation was applied during the evaluation phase. The model with the best overall performance was selected and deployed as a REST API using the Flask framework. This API was integrated into the ATPPM system to provide real-time root cause predictions for newly reported defects. The deployment design allows for future model updates without disrupting the existing system.

2.5.6. API Integration

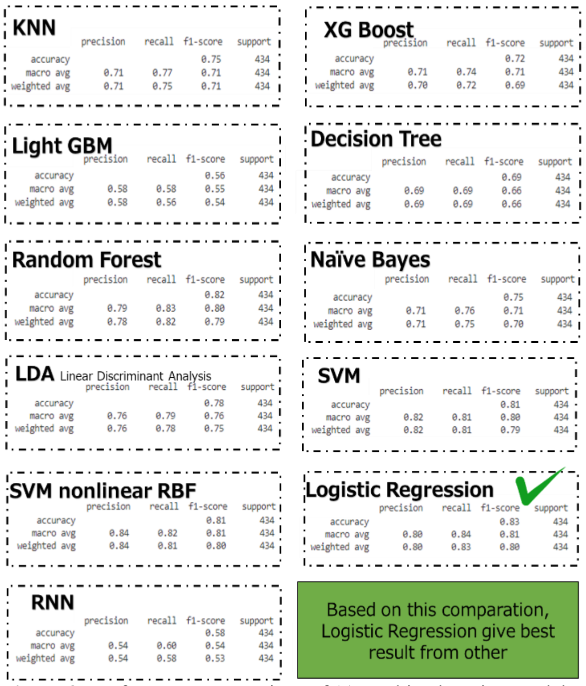
The final model was integrated into the ATPPM system via an internal REST API. When a new defect is recorded in the system, the API suggests possible root causes and enables operators to validate and take action accordingly.

3. Result

The model evaluation phase was carried out using a combination of performance metrics, including Precision, Recall, and F1-Score, as well as K-Fold Cross Validation. Totally eleven machine learning models were tested during this phase to identify the most suitable model for deployment.

3.1 Model Performance Comparison

A total of 11 machine learning models were trained and evaluated using the preprocessed dataset. The initial evaluation was based on the performance on the test set, with results summarized in **Figure 3**. Evaluation that measured were Precision, Recall, and F1-score metrics. These metrics helped assess the ability of each model to detect defect patterns accurately.



KNN
```
              precision    recall  f1-score   support
    accuracy                           0.75       434
   macro avg       0.71      0.77      0.71       434
weighted avg       0.71      0.75      0.71       434
```

XG Boost
```
              precision    recall  f1-score   support
    accuracy                           0.72       434
   macro avg       0.71      0.74      0.71       434
weighted avg       0.70      0.72      0.69       434
```

Light GBM
```
              precision    recall  f1-score   support
    accuracy                           0.56       434
   macro avg       0.58      0.58      0.55       434
weighted avg       0.58      0.56      0.54       434
```

Decision Tree
```
              precision    recall  f1-score   support
    accuracy                           0.69       434
   macro avg       0.69      0.69      0.66       434
weighted avg       0.69      0.69      0.66       434
```

Random Forest
```
              precision    recall  f1-score   support
    accuracy                           0.82       434
   macro avg       0.79      0.83      0.80       434
weighted avg       0.78      0.82      0.79       434
```

Naïve Bayes
```
              precision    recall  f1-score   support
    accuracy                           0.75       434
   macro avg       0.71      0.76      0.71       434
weighted avg       0.71      0.75      0.70       434
```

LDA Linear Discriminant Analysis
```
              precision    recall  f1-score   support
    accuracy                           0.78       434
   macro avg       0.76      0.79      0.76       434
weighted avg       0.76      0.78      0.75       434
```

SVM
```
              precision    recall  f1-score   support
    accuracy                           0.81       434
   macro avg       0.82      0.81      0.80       434
weighted avg       0.82      0.81      0.79       434
```

SVM nonlinear RBF
```
              precision    recall  f1-score   support
    accuracy                           0.81       434
   macro avg       0.84      0.82      0.81       434
weighted avg       0.84      0.81      0.80       434
```

Logistic Regression ✓
```
              precision    recall  f1-score   support
    accuracy                           0.83       434
   macro avg       0.80      0.84      0.81       434
weighted avg       0.80      0.83      0.80       434
```

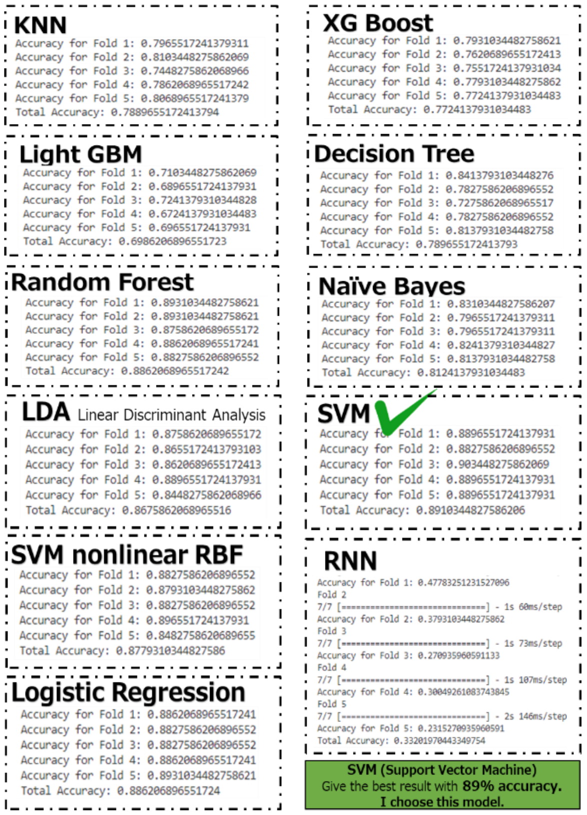
RNN
```
              precision    recall  f1-score   support
    accuracy                           0.58       434
   macro avg       0.54      0.60      0.54       434
weighted avg       0.54      0.58      0.53       434
```

Based on this comparation, Logistic Regression give best result from other

Figure 3: Performance comparison of 11 machine learning models.

3.2 K-Fold Cross Validation

To ensure the generalizability and robustness of each model, K-Fold Cross Validation was applied with **k = 5**. The result of accuracy across all folds for each algorithm is shown in Figure 4, providing a more reliable comparison of the models' consistency across subsets of data.



KNN
```
Accuracy for Fold 1: 0.7965517241379311
Accuracy for Fold 2: 0.8103448275862069
Accuracy for Fold 3: 0.7448275862068966
Accuracy for Fold 4: 0.7862068965517242
Accuracy for Fold 5: 0.8068965517241379
Total Accuracy: 0.7889655172413794
```

XG Boost
```
Accuracy for Fold 1: 0.7931034482758621
Accuracy for Fold 2: 0.7620689655172413
Accuracy for Fold 3: 0.7551724137931034
Accuracy for Fold 4: 0.7793103448275862
Accuracy for Fold 5: 0.7724137931034483
Total Accuracy: 0.7724137931034483
```

Light GBM
```
Accuracy for Fold 1: 0.7103448275862069
Accuracy for Fold 2: 0.6896551724137931
Accuracy for Fold 3: 0.7241379310344828
Accuracy for Fold 4: 0.6724137931034483
Accuracy for Fold 5: 0.696551724137931
Total Accuracy: 0.6986206896551723
```

Decision Tree
```
Accuracy for Fold 1: 0.8413793103448276
Accuracy for Fold 2: 0.7827586206896552
Accuracy for Fold 3: 0.7275862068965517
Accuracy for Fold 4: 0.7827586206896552
Accuracy for Fold 5: 0.8137931034482758
Total Accuracy: 0.789655172413793
```

Random Forest
```
Accuracy for Fold 1: 0.8931034482758621
Accuracy for Fold 2: 0.8931034482758621
Accuracy for Fold 3: 0.8758620689655172
Accuracy for Fold 4: 0.8862068965517241
Accuracy for Fold 5: 0.8827586206896552
Total Accuracy: 0.8862068965517242
```

Naïve Bayes
```
Accuracy for Fold 1: 0.8310344827586207
Accuracy for Fold 2: 0.7965517241379311
Accuracy for Fold 3: 0.7965517241379311
Accuracy for Fold 4: 0.8241379310344827
Accuracy for Fold 5: 0.8137931034482758
Total Accuracy: 0.8124137931034483
```

LDA Linear Discriminant Analysis
```
Accuracy for Fold 1: 0.8758620689655172
Accuracy for Fold 2: 0.8655172413793103
Accuracy for Fold 3: 0.8620689655172413
Accuracy for Fold 4: 0.8896551724137931
Accuracy for Fold 5: 0.8448275862068966
Total Accuracy: 0.8675862068965516
```

SVM ✓
```
Accuracy for Fold 1: 0.8896551724137931
Accuracy for Fold 2: 0.8827586206896552
Accuracy for Fold 3: 0.903448275862069
Accuracy for Fold 4: 0.8896551724137931
Accuracy for Fold 5: 0.8896551724137931
Total Accuracy: 0.8910344827586206
```

SVM nonlinear RBF
```
Accuracy for Fold 1: 0.8827586206896552
Accuracy for Fold 2: 0.8793103448275862
Accuracy for Fold 3: 0.8827586206896552
Accuracy for Fold 4: 0.896551724137931
Accuracy for Fold 5: 0.8482758620689655
Total Accuracy: 0.8779310344827586
```

RNN
```
Accuracy for Fold 1: 0.47783251231527096
Fold 2
7/7 [==============================] - 1s 60ms/step
Accuracy for Fold 2: 0.3793103448275862
Fold 3
7/7 [==============================] - 1s 73ms/step
Accuracy for Fold 3: 0.270935960591133
Fold 4
7/7 [==============================] - 1s 107ms/step
Accuracy for Fold 4: 0.30049261083743845
Fold 5
7/7 [==============================] - 2s 146ms/step
Accuracy for Fold 5: 0.2315270935960591
Total Accuracy: 0.33201970443349754
```

Logistic Regression
```
Accuracy for Fold 1: 0.8862068965517241
Accuracy for Fold 2: 0.8827586206896552
Accuracy for Fold 3: 0.8827586206896552
Accuracy for Fold 4: 0.8862068965517241
Accuracy for Fold 5: 0.8931034482758621
Total Accuracy: 0.886206896551724
```

SVM (Support Vector Machine)
Give the best result with **89% accuracy.**
I choose this model.

Figure 4: Performance comparison of 11 machine learning models.

3.3 Model Deployment

Based on the evaluation results, the model with the most consistent and highest overall performance was selected for deployment. The selected model was integrated into a RESTful API and deployed to interface directly with the ATPPM system used on the production line. The architecture between API and ATPPM system / web app is explained in Figure 5, and the design of ATPPM system that already improved by AI is explained in Figure 6.

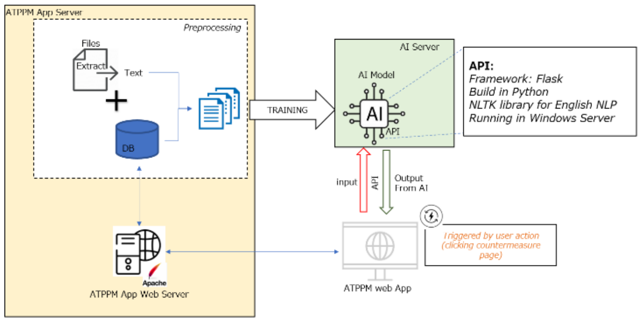


Figure 5: Architecture of ATPPM System and AI API

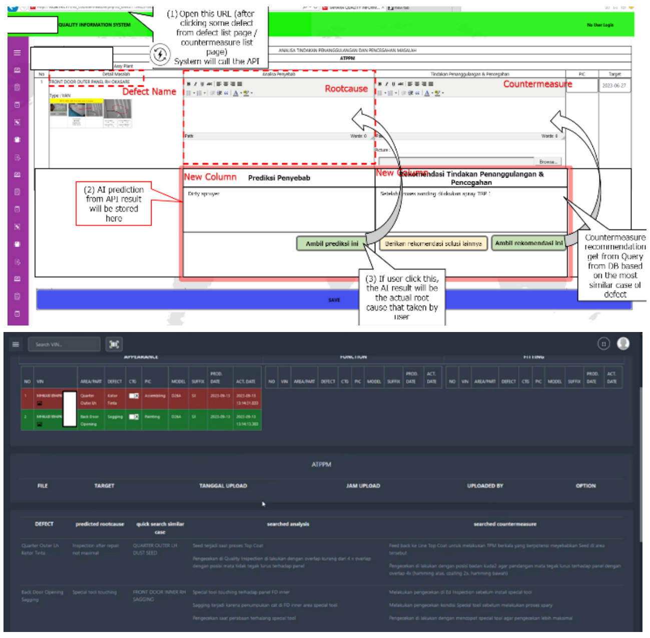

QUALITY INFORMATION SYSTEM
(1) Open this URL (after clicking some defect from defect list page / countermeasure list page) System will call the API
ANALISA TINDAKAN PENANGGULANGAN DAN PENCEGAHAN MASALAH
ATPPM
Defect Name
Rootcause
Countermeasure
FRONT DOOR OUTER PANEL RH OKASARE
New Column
Prediksi Penyebab
New Column
Rekomendasi Tindakan Penanggulangan & Pencegahan
(2) AI prediction from API result will be stored here
Dirty sprayer
Ambil prediksi ini
Berikan rekomendasi solusi lainnya
Ambil rekomendasi ini
(3) If user click this, the AI result will be the actual root cause that taken by user
Countermeasure recommendation get from Query from DB based on the most similar case of defect
SAVE
APPEARANCE
FUNCTION
FITTING
ATPPM
FILE
TARGET
TANGGAL UPLOAD
JAM UPLOAD
UPLOADED BY
OPTION
DEFECT
predicted rootcause
quick search similar case
searched analysis
searched countermeasure

Figure 6: Design and Screenshot of ATPPM system after AI integration

3.4 Operational Impact

The integration of the machine learning-based recommendation system into the ATPPM process has led to significant operational benefits. Previously, manual root cause analysis for a new defect case could take up to 230 minutes per case. With the assistance of AI-generated predictions or recommendations, the analysis time has been reduced to approximately 2 minutes, resulting in a time savings of 228 minutes per case.

Beyond time efficiency, the implementation has also delivered seamless integration into shop floor operations, where operators now have direct access to the system, as illustrated in Figure 7. This practical application demonstrates not only the conceptual effectiveness of the system but also its adaptability and usefulness in real-world production environments.

Figure 6: Implementation of ATPPM System Integrated with AI

This improvement has significantly enhanced operational efficiency by enabling much faster decision-making, allowing analysts to shift their focus from time-consuming, repetitive defect classification tasks to more strategic, preventive actions. Additionally, the use of AI-driven recommendations has introduced greater consistency and objectivity in defect classification, thereby minimizing the influence of human bias in the evaluation process. Furthermore, the integration of this system has strengthened traceability and digital documentation within the ATPPM environment, providing a more structured foundation to support ongoing quality improvements and data-driven decision-making initiatives.

4. Conclusion

In this study, we have demonstrated a practical application of machine learning in the automotive manufacturing domain to enhance root cause analysis for production defects. Historically, the identification of root causes has depended heavily on manual analysis by experienced personnel, which was time-consuming, subjective, and often delayed the implementation of effective countermeasures. By leveraging historical defect data embedded in the ATPPM system—containing thousands of records across time—we have built a data-driven solution capable of significantly improving both the speed and consistency of root cause evaluation.

Through a structured development process involving data cleaning, model training, and integration via API, we successfully implemented a machine learning pipeline that classifies defect cases based on prior knowledge. From the evaluation of eleven models, Support Vector Machine (SVM) emerged as the most robust in cross-validation, achieving an average accuracy of 89.1%, while Logistic Regression offered the highest initial F1-score of 0.83. The chosen model was deployed using a Flask-based REST API and embedded into the factory's existing ATPPM infrastructure.

The impact of this integration has been substantial. With the support of AI-based prediction, the root cause analysis process was reduced from an average of 230 minutes per case to just 2 minutes. This improvement not only enhances operational efficiency but also enables production analysts to act more quickly, shifting focus from repetitive classification tasks to preventive and value-adding activities. Furthermore, the system offers more consistent and objective analysis results, minimizes human bias, and improves traceability and documentation within the production quality system.

This study confirms that integrating machine learning into defect analysis workflows can provide tangible benefits in a manufacturing environment. Looking forward, expanding the system to automatically retrain the model on a regular schedule, incorporating additional data sources such as real-time big data and quality management systems, and scaling the implementation to multiple production lines hold strong potential for wider organizational impact and long-term operational resilience.

REFERENCES

[1] Altman, A., & Krzywinski, M. (2017). The art of the KNN algorithm. *Nature Methods*, 14, 603–604.

[2] Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.

[3] Bousdekis, A., Lepenioti, K., Apostolou, D., & Mentzas, G. (2021). Decision making in predictive maintenance: Literature review and research agenda for Industry 4.0. *Journal of Intelligent Manufacturing*, 32, 1223–1250.

[4] Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5–32.

[5] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD*.

[6] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20, 273–297.

[7] Doggett, A. M. (2005). Root Cause Analysis: A Framework for Tool Selection. *The Quality Management Journal*, 12(4), 34–45.

[8] Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29, 103–130.

[9] Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, **7**(2), 179–188.

[10] Ge, Z., Song, Z., & Gao, F. (2013). Review of recent research on data-based process monitoring. *Industrial & Engineering Chemistry Research*, 52(10), 3543–3562.

[11] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression* (3rd ed.). Wiley.

[12] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 30**.**

[13] Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, 2, 1137–1143.

[14] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521, 436–444.

[15] Lee, J., Bagheri, B., & Kao, H. A. (2015). A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3, 18–23.

[16] Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1), 81–106.

[17] Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4), 427–437.

[18] Wuest, T., Weimer, D., Irgens, C., & Thoben, K. D. (2016). Machine learning in manufacturing: advantages, challenges, and applications. *Production & Manufacturing Research*, 4(1), 23–45.

[19] Zhang, Y., Jin, R., & Zhou, Z.-H. (2010). Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1), 43–52.

[20] Zhang, Y., Ren, S., Liu, Y., & Si, S. (2017). A big data analytics architecture for cleaner manufacturing and maintenance processes of complex products. *Journal of Cleaner Production*, 142, 626–641.