

Article

SynergyAI: A Human–AI Pair Programming Tool Based on Dataflow [†]

Le Jiang ¹, Shingo Yamaguchi ^{1,*}  and Mohd Anuaruddin Bin Ahmadon ² 

¹ Graduate School of Sciences and Technology for Innovation, Yamaguchi University, 2-16-1 Tokiwadai, Ube 755-8611, Japan; c503vgw@yamaguchi-u.ac.jp

² Faculty of Sciences and Information Technology, Universiti Teknologi PETRONAS, Seri Iskandar 32610, Perak Darul Ridzuan, Malaysia; anuaruddin.ahmadon@utp.edu.my

* Correspondence: shingo@yamaguchi-u.ac.jp; Tel.: +81-836-85-9510

[†] This paper is an extension of a conference papers: (1) Zhang, D.; Bin Ahmadon, M.; Yamaguchi, S. Human-AI Pair Programming by Data Stream and Its Application Example. In Proceedings of the 2022 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Yeosu, Republic of Korea, 26–28 October 2022; pp. 1–4. (2) Jiang, L.; Bin, Ahmadon, M.; Zhang, D.; Fukada, T.; Yamaguchi, S. SynergyAI: An Human-AI Pair Programming Design Tool Based on Program Net. In Proceedings of the 2023 11th International Conference on Information and Education Technology (ICIET), Fujisawa, Japan, 18–20 March 2023; pp. 210–214.

Abstract: This paper proposes SynergyAI, an AI–human pair programming tool that represents predictive models as dataflows composed of AI, input, and output nodes. By visualizing decision tree models and integrating them with dataflow diagrams, SynergyAI effectively addresses the machine learning black-box problem. Additionally, the tool leverages comprehensive prediction algorithms and ensemble learning to simplify the operation of complex dataflows and mitigate overfitting risks. SynergyAI also features an AI assistant that utilizes scatter plot matrices and data correlation analysis to help programmers select data and optimize model structures. Experimental results demonstrate that, through human–AI collaboration, SynergyAI achieved an accuracy of 85% in predicting mechanical failures in a chocolate factory, providing an efficient and powerful tool for programming and data analysis.

Keywords: data flow; decision tree; human–AI collaboration; explainability



Academic Editors: Emilio Matricciani,
Heming Jia and Zhigang Chu

Received: 13 January 2025

Revised: 24 February 2025

Accepted: 24 February 2025

Published: 26 February 2025

Citation: Jiang, L.; Yamaguchi, S.; Bin Ahmadon, M.A. SynergyAI: A Human–AI Pair Programming Tool Based on Dataflow. *Information* **2025**, *16*, 178. <https://doi.org/10.3390/info16030178>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

AI (Artificial Intelligence) technology is now widely used in fields such as medical diagnostics and customer service. For example, AI-based image recognition helps detect cancer cells [1], and natural language processing powers chatbots for customer service and translation apps for real-time multilingual conversion [2]. Many companies are integrating AI to improve office tasks, enhancing work efficiency and quality. However, the black-box problem [3] in machine learning challenges model explainability and accuracy verification. This lack of transparency can lead to untraceable accountability when AI is relied upon for critical decisions [4]. For instance, autonomous vehicles may fail in complex or adverse conditions, posing safety hazards [5]. Additionally, black-box models may misjudge conflicting data, leading to cumulative errors and ethical concerns.

Existing white-box models aim to enhance transparency but rely on manual data collection and training, leading to trial-and-error processes and resource wastage [6,7]. Therefore, how to improve development efficiency and reduce resource waste while maintaining model explainability has become an urgent issue. To address this, we propose SynergyAI 2024 (hereinafter referred to as SynergyAI). The primary goal of SynergyAI is to

facilitate collaboration between humans and AI, while its secondary goal is to achieve AI model explainability [8]. To address these issues, we developed an AI model-building tool focused on explainability and human–AI collaboration to improve transparency and user trust, thereby driving further AI development.

SynergyAI distinguishes itself by enabling the construction of complex white-box models while incorporating an AI assistant that aids users in selecting relevant data, refining model structures, and optimizing performance, bridging the gap between AI automation and human interpretability.

Drawing on large-scale research [9] into the usability of AI programming assistants, our tool design centers on how users interact with AI and addresses the need for interpretability and transparency. To enhance users' understanding and trust in the system, we have incorporated two key features: first, we generate structure modification suggestions for user-designed models based on correlation coefficients, helping to rapidly build and iterate complex models; second, we adopt a white-box design that allows users to view each input, output, and the AI's decision-making process in real time. The former significantly improves model design efficiency, while the latter effectively responds to the emphasized requirements for transparency and control, enabling users to leverage AI-generated recommendations while fully overseeing the model's final outputs and decision logic.

The purpose of this study is to optimize and expand the human–AI pair programming model proposed by Zhang et al. [10], resulting in the development of SynergyAI. This web application improves model interpretability, enhances AI assistant functionalities, and boosts performance. This study addresses the black-box problem in traditional AI models while improving predictive model efficiency through algorithm optimization and dataflow design.

To enhance model interpretability, SynergyAI integrates visual dataflow representations with decision tree models. Unlike traditional AI tools that rely on opaque neural networks, SynergyAI decomposes predictive models into modular components, allowing users to trace decision pathways and understand model behavior more intuitively. Additionally, by incorporating ensemble learning techniques, SynergyAI mitigates overfitting risks and enhances model reliability.

The AI assistant in SynergyAI plays a crucial role in improving user experience and model performance. It includes the following:

- Scatter Plot Matrices and Correlation Analysis: assists in feature selection by highlighting relationships between variables.
- Automated Hyperparameter Tuning: suggests optimal configurations to enhance model efficiency.
- Dataflow Optimization Suggestions: recommends adjustments to improve model accuracy and stability.

By introducing transparent dataflow models and intelligent AI assistance, SynergyAI effectively tackles complex social issues and IoT challenges, enhancing decision-making, production efficiency, and model reliability. Ultimately, this study contributes to efficient human–AI collaboration, model optimization, and the practical application of AI.

The structure of this paper is as follows. Section 2 discusses the difficulties in AI programming, including the black-box problem, high trial-and-error costs, and the lack of human–AI collaboration. Section 3 introduces the concept of SynergyAI, providing detailed explanations of its dataflow design, frontend and backend development, modularization, and ensemble learning functionalities. Section 4 describes the AI assistant in SynergyAI, explaining how it helps users optimize data selection, improve dataflow structures, and enhance the performance of decision tree models, thereby facilitating more

efficient human–AI collaboration. Together, Sections 3 and 4 illustrate how SynergyAI addresses the challenges presented in Section 2. Section 5 presents application examples, using chocolate factory mechanical failure prediction as a case study to demonstrate the effectiveness of SynergyAI in real-world scenarios. Section 6 concludes this paper by summarizing SynergyAI’s contributions and its potential impact on AI model development and practical applications.

2. AI Programming and Its Difficulties

2.1. Related Work

This chapter presents related research on the issues discussed in Section 1, including model explainability, trial and error in design, and the lack of human–AI collaboration [11].

(A) Human Programming Model

Many pre-trained machine learning models exist. For example, Krizhevsky et al. [12] achieved remarkable results in the ImageNet competition, laying the foundation for deep learning in image classification. However, due to their complexity, these models are often regarded as black boxes, lacking explainability and transparency. Humans can use these models to select input data and obtain results, but the underlying mechanisms remain opaque.

Humans can also employ AI programming to train and create prediction models, but this process is time-consuming and requires gathering relevant data [13]. Additionally, humans often struggle to identify correlations between data, making model optimization challenging. These issues are summarized as the black-box problem, trial and error, and redundant data.

(B) Human–AI Programming Model

To address challenges like trial and error [14] and redundant data [15], we adopt human–machine interaction principles in building our AI model. We draw upon Scibilia et al.’s work [16], which emphasizes mimicking human control behavior in human–machine interaction. Zhang et al. [10] proposed a dataflow design tool for IoT systems (Figure 1), where AI analyzes models built by humans and provides modification advice, combining human experience with AI analysis.

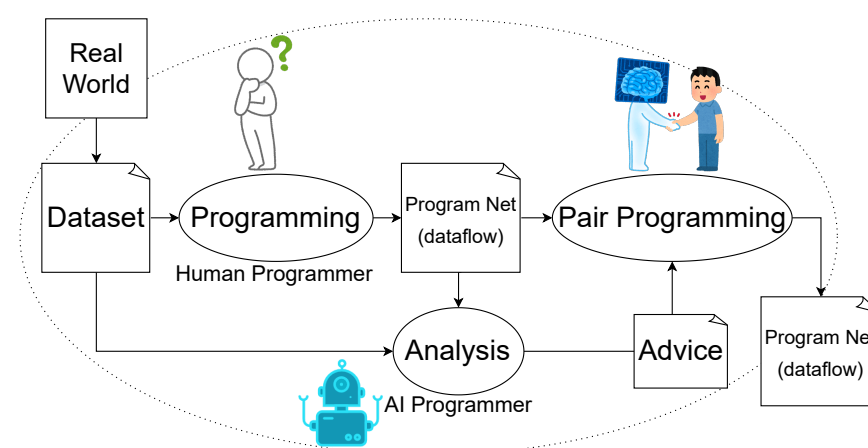


Figure 1. Human–AI model from Zhang et al. [10].

Building on this model, we developed SynergyAI 2023 [17], a visualization tool that enhances human–AI collaboration. SynergyAI 2023 uses dataflow representations, allowing users to construct AI models through dataflow editing. It also includes an AI system that

suggests dataflow modifications based on correlation coefficients, enabling users to quickly optimize models with AI assistance. However, the initial version of SynergyAI 2023 did not fully address black-box and explainability issues due to transparent AI nodes. The AI advice, based solely on correlation coefficients, lacked visualization capabilities, limiting the model's explainability and credibility.

(C) Comparison with Different Tools

We compared SynergyAI with existing AI model-building tools and intelligent code assistance tools, as shown in Table 1. Traditional AI model-building frameworks like TensorFlow [18] allow for the creation of complex neural network models but are inherently black-box systems, making troubleshooting difficult. PyCaret [19], on the other hand, supports white-box models but lacks AI-assisted guidance for data selection and model optimization. Furthermore, code assistance tools such as GitHub Copilot, Tabnine, and Visual Studio IntelliCode provide code completion based on context-aware AI, while code generation tools like DeepCode and Codeium assist in code refactoring and error detection. However, these tools do not focus on model explainability or AI–human collaboration in predictive modeling.

We also examined how SynergyAI differs from popular engineering and IoT development platforms such as LabVIEW [20], ThingWorx [21], and Simulink [22]. LabVIEW is renowned for its graphical programming environment and hardware integration features, focusing on test measurement and control systems. ThingWorx, developed by PTC Inc., emphasizes IoT device connectivity, real-time monitoring, and data management in industrial settings. Meanwhile, Simulink provides a multi-domain simulation environment via block diagrams, integrating closely with MATLAB for model-based design. Although each platform addresses specific domain requirements (e.g., instrumentation, IoT connectivity, or multi-domain simulation), none of them feature an embedded AI assistant dedicated to automatically recommending and optimizing dataflow architectures, nor do they emphasize complete transparency in model building and inference. Dataflow can also be designed to connect components in systems for rapid deployment [23]. By contrast, SynergyAI integrates dataflow suggestions, correlation-based feature selection, and a white-box visualization approach, thereby offering a unique combination of efficiency and explainability within AI-driven predictive modeling.

Table 1. Comparison of AI development tools: SynergyAI's advantages in explainability and AI assistance.

	Dataflow	White Box	AI Assistant	Data Selection	Code Assistance
TensorFlow	✓	×	×	Human only	×
PyCaret	✓	✓	×	Human only	×
GitHub Copilot	×	×	✓	×	✓
Tabnine	×	×	✓	×	✓
DeepCode	×	×	✓	×	✓
SynergyAI	✓	✓	✓	Human + AI	×

Note: ✓ indicates support, and × indicates lack of support.

2.2. Difficulties in AI Programming

(A) Black-Box Problem

AI models, despite their impressive performance, are difficult to explain and understand. Humans can only observe the input and output, not the internal workings, creating opacity that leads to a lack of trust. This is especially problematic in critical decision-making processes, where explainability is crucial for acceptance and confidence in AI systems [8,24].

(B) Trial and Error

AI model training often relies on intuition and experience for data selection [25], which can be flawed and result in suboptimal performance. This leads to a time-consuming trial-and-error process, where parameters are adjusted, models are retrained, and results are validated. To improve efficiency, more systematic and data-driven methods are needed to optimize model training and reduce reliance on trial and error.

(C) Lack of Human–AI Cooperative Ability

Conventional AI tools involve minimal collaboration between humans and AI after the initial model design [26]. Integrating an AI assistant into the design process can enhance collaboration by providing real-time feedback [27], suggesting improvements, and identifying potential pitfalls. This approach not only improves model quality but also empowers human designers to make more informed decisions, leading to more effective AI solutions.

3. SynergyAI Concept

This chapter introduces the principles of SynergyAI, including its model construction, frontend and backend development, and key functionalities such as model training, prediction, and ensemble learning. We will demonstrate these functionalities using detailed algorithms and examples.

Figure 2 illustrates the system architecture of SynergyAI. The frontend enables users to design models, send requests to the backend for predictions or data analysis, and retrieve results. The backend, implemented with Flask, consists of a model server for training models and a file server for storing models and results.

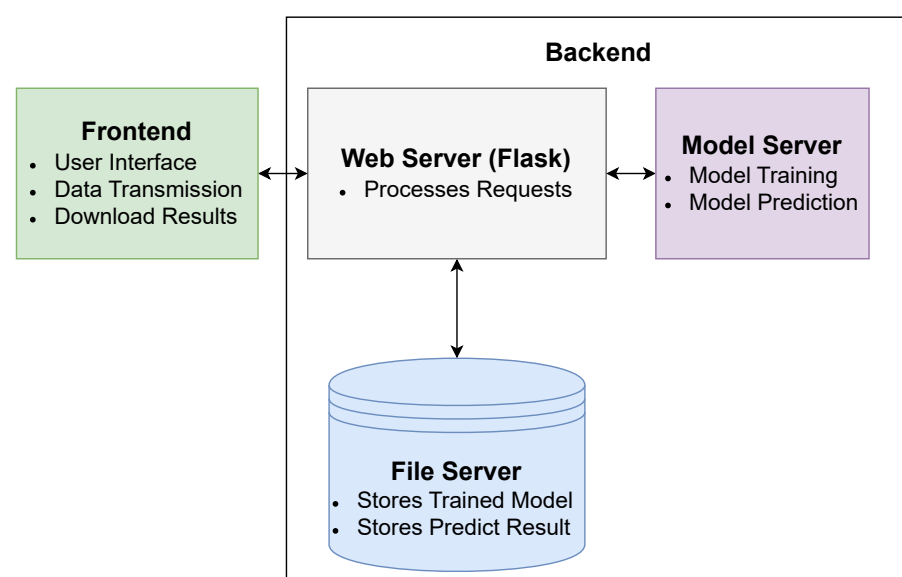


Figure 2. System architecture of SynergyAI.

3.1. DataFlow

SynergyAI is a supervised learning tool that uses dataflow to represent models, as shown in Figure 3. The dataflow consists of three types of nodes: input, AI, and output, each playing a distinct role in the model. Each node is labeled with a data identifier, and arrows indicate the direction of dataflow. AI nodes are further labeled with the corresponding models, such as DT (decision tree), RF (Random Forest), and DL (deep learning).

The explainability of SynergyAI is divided into two levels: the dataflow level and the node level. At the dataflow level, models designed with SynergyAI provide a clear visualization of the input–output relationships of each node. Users can insert intermediate layers between inputs and outputs to complete a series of predictions, exploring how inputs are progressively transformed into outputs. As shown in Figure 3, a1, a2, and a4 are used as inputs to predict b1 with a decision tree. Similarly, a2, a3, and a4 serve as inputs to predict b2 using a deep learning model. Finally, the predictions of b1 and b2 are combined to predict c1 using another decision tree.

At the node level, to ensure a fully transparent (white-box) model, users can choose to train models using decision trees. In this case, not only are the inputs and outputs of each node visible, but the decision-making process within each AI node is also presented to the user.

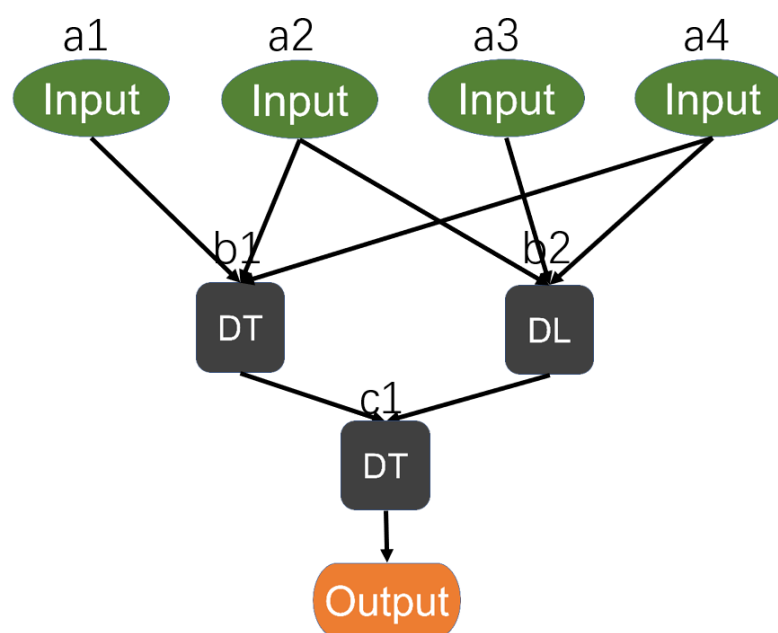


Figure 3. Example of a two-layer dataflow with decision tree (DT) and deep learning (DL) nodes.

3.2. Frontend Development

Cytoscape [28] is an open-source platform for visualizing complex networks and integrating attribute data, widely used in biological and other fields. In this study, we developed a web application using Python and JavaScript, leveraging Cytoscape.js [29] for its robust API that supports event handling, graph layouts, and dynamic updates, enabling the addition, removal, or modification of nodes and edges. Using Cytoscape.js, we designed a canvas for displaying dataflows, along with a toolbar and node panel for creating, editing, and managing dataflows. Key functionalities include node creation, connection, deletion, and dataflow management such as downloading and reading.

The dataflow comprises various elements, including nodes (input nodes, AI nodes, and output nodes) and edges. To ensure prediction processes, we defined connection rules

between nodes in the program, as shown in Table 2. By adhering to these rules, the dataflow can seamlessly proceed from input nodes, through AI nodes for processing, and finally reach the output nodes.

Table 2. Node connection rules in the dataflow.

Node Name	Input	Output
Input Node	None	AI Node
AI Node	Input Node or AI Node	AI Node or Output Node
Output Node	AI Node	None

Note: “None” indicates no input or output is allowed.

3.3. Backend Development

The backend was developed using Python (version 3.10.7)’s Flask framework to manage SynergyAI’s core functionalities and route requests. SynergyAI employs Scikit-learn for algorithms like Random Forest, SVM, and decision tree, while TensorFlow’s API is used for deep learning models. This API supports training, saving, and prediction for diverse AI models.

During training, the frontend sends the relevant ‘csv_column’ and ‘csv_data’ of the AI node to the backend. Input data are organized into a dictionary, mapping ‘csv_column’ as keys to ‘csv_data’ as values. The data are then processed into a model-compatible format using JSON and Pandas, ensuring seamless integration with the AI models. During model training, the system allocates 70% of the dataset for training and reserves the remaining 30% to evaluate the model’s accuracy.

3.4. Dataflow Prediction

When a dataflow contains numerous AI nodes with interdependent input–output relationships, ensuring that each node receives the required input, generates predictions, and passes results to subsequent nodes can be challenging. To address this, we propose a comprehensive prediction algorithm (Algorithm 1) for the entire dataflow. Since the algorithm is implemented in both Python and JavaScript, and JavaScript is predominantly single-threaded, it is designed as a single-threaded algorithm.

The algorithm requires the dataflow to meet the following conditions:

- All nodes must have assigned data.
- Each node must have at least one input or output.
- Each dataflow must start from an input node, traverse through AI nodes, and end at an output node, as shown in Figure 3.

In the algorithm, the function `predict(model, inputlist)` takes a model and a list of input data as parameters and returns the predicted outcome. Similarly, the function `getdata(model)` accepts a model as input and retrieves the input data stream (input list) in the specific sequence required by the model for accurate predictions.

3.5. Dataflow Modularity

Dataflows with numerous nodes and edges can become overly complex, as they are often composed of multiple simpler dataflows, making them difficult to understand and modify. To simplify modifications, we introduced a modular feature that encapsulates dataflows into module nodes. Users can load dataflow files into module nodes and select inputs and outputs to construct new dataflows. This approach streamlines the creation of complex dataflows. Users can view a module’s structure directly within the module node, ensuring transparency and preventing the model from becoming a “black box” [30].

As shown in Figure 4, each dataflow consists of an input layer, an AI layer, and an output layer. These layers are saved as JSON files on the server, with module nodes storing their paths. This enables dataflows to be replaced with module nodes, whose structures can be retrieved and displayed on the frontend to ensure transparency.

Algorithm 1 Comprehensive prediction algorithm

Require: User input data, AI node $model$, Function $predict(model, inputlist)$, Function $getdata(model)$

```

1: Initialize model execution list:  $model\_run\_list \rightarrow \{\}$ 
2: User inputs data for each input node  $data$ 
3: Add all AI nodes to  $model\_run\_list$ 
4: while  $model\_run\_list$  is not empty do
5:   for each model  $model_i$  in  $model\_run\_list$  do
6:     Set completion flag  $ready$  to 1
7:     for each input node  $J$  in  $model_i.inputlist$  do
8:       if  $J.data$  is null then
9:         Set  $ready$  to 0 and break
10:      end if
11:    end for
12:    if  $ready$  is 1 then
13:      Update the model's  $data$  using the prediction function:  $model_i.data \rightarrow$ 
         $predict(model_i, getdata(model_i))$ 
14:      Remove  $model_i$  from  $model\_run\_list$ 
15:    end if
16:  end for
17: end while
  
```

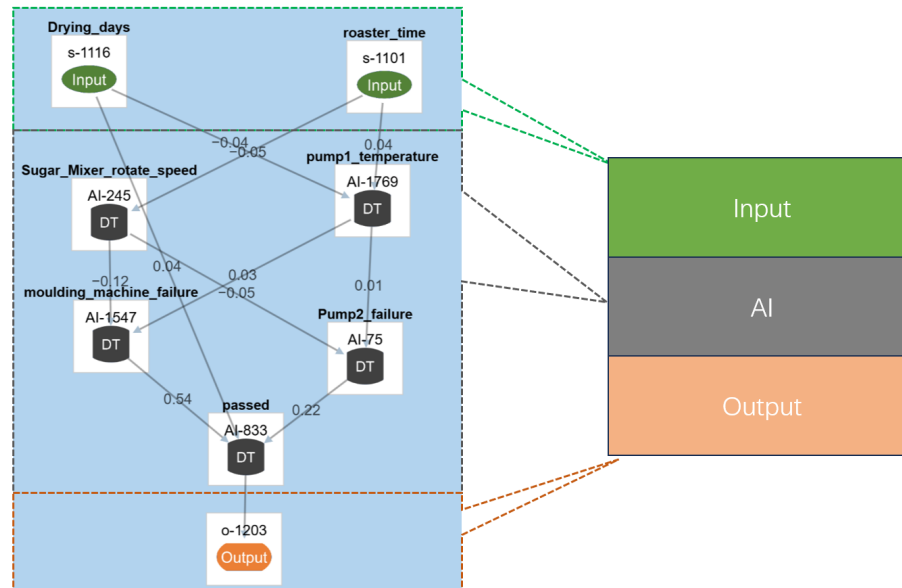


Figure 4. Three-layer structure of dataflow: input, AI, and output.

3.6. Ensemble Learning in SynergyAI

To improve the accuracy of prediction results and further reduce the risk of overfitting, we developed an ensemble learning [31] feature based on dataflows. As shown in Figure 5, the stacking functionality uses a new input node to store a dataset. AI nodes or modular nodes act as base learners, and different datasets are used to train these base learners. Finally, the outputs of all base learners are used as inputs for a meta-learner [32], which is trained to generate the final prediction result.

In addition to stacking, we also developed a bagging functionality. Bagging combines the outputs of various AI nodes or modular nodes by taking a majority vote or calculating the average to produce the final output.

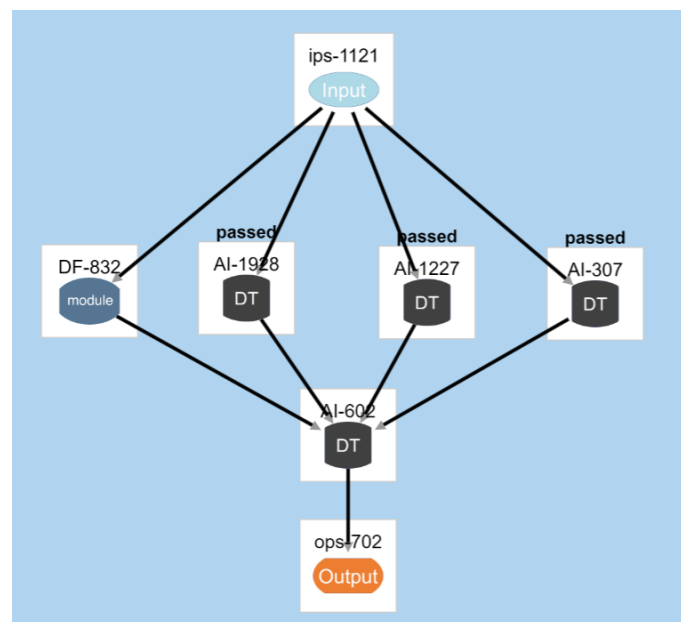


Figure 5. Dataflow representation of a stacking-based ensemble learning model.

4. AI Assistant in SynergyAI

This chapter highlights the key functionalities of the AI assistant in SynergyAI, including visualized decision trees to address the black-box problem, decision tree optimization suggestions, data selection suggestions, and dataflow construction optimization suggestions. Finally, we integrate these functionalities into a human–AI collaboration model, demonstrating their combined effectiveness.

4.1. Explainability of AI Model

Representing models by dataflow enables visualization of the model's structure, including input, output, and dataflow direction. However, AI nodes remain black boxes, with opaque decision-making processes. To address this, we visualize decision tree models next to their AI nodes, as shown in Figure 6.

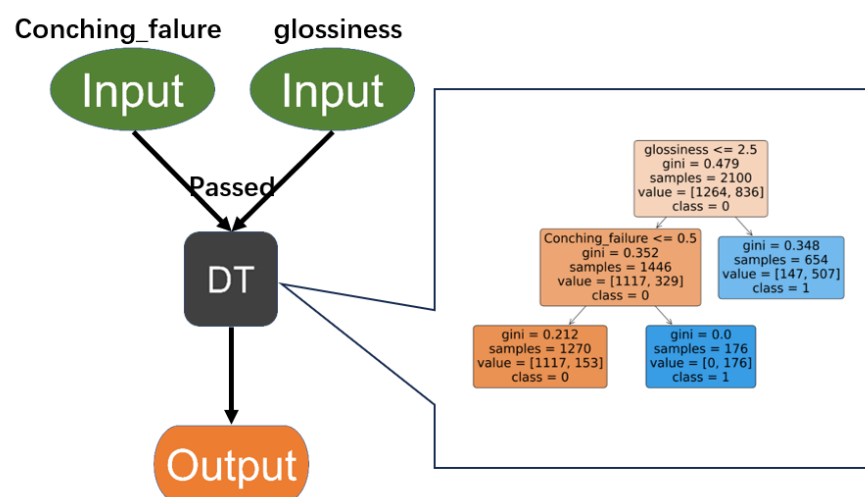


Figure 6. Dataflow with decision tree.

Figure 6 illustrates a dataflow using “Conching_failure” and “glossiness” as features to predict “passed” with a decision tree model. Displaying the decision tree next to the DT node clarifies the conditions for predicting “passed”. For example, when “glossiness” is greater than 2.5, or “glossiness” is less than 2.5 and “Conching_failure” is greater than 0.5, the output is 1; otherwise, it is 0. The decision tree also provides insights into sample splits, Gini coefficients, and sample counts, which help evaluate reliability and overfitting risks.

Using decision trees from the Scikit-learn library, this approach transforms the dataflow into a white-box model. To ensure explainability, the AI models used in this study are primarily decision tree models, and some AI assistant suggestions are tailored for decision trees. Details of the decision tree optimization function will be discussed in Section 4.2, Paragraph C.

4.2. Dataflow with AI Advice

We developed three AI advice functionalities to enhance model construction and training efficiency. These include data selection for individual AI nodes, optimization of the dataflow structure, and parameter tuning for decision trees within AI nodes. By leveraging these functionalities, users can minimize trial and error, streamline data analysis, uncover correlations, and rapidly develop highly accurate models.

(A) Advice 1: Data Selection

To assist in selecting input data for an AI node, the AI assistant ranks columns of the loaded CSV data by the absolute value of their correlation coefficients. After data selection, the AI assistant generates a Scatterplot Matrix of the training features and target labels. The frontend sends selected node data to the backend to generate the Scatterplot Matrix. The backend uses Seaborn’s pairplot method to create the visualization, labeling axes starting with ‘AI’ in red. The matrix is then encoded as a Base64 string and sent back to the frontend for display. The Scatterplot Matrix reveals linear and nonlinear relationships between data, while the diagonal line highlights data distribution, providing users with valuable insights for model construction and training.

Since the correlation coefficient only captures linear relationships, it may underestimate significant relationships in data with non-monotonic patterns. For example, when $y = \sin(x)$, the correlation coefficient between x and y is low, yet there is a clear relationship. A Scatterplot Matrix allows direct visualization of such relationships, reducing the risk of overlooking important patterns.

(B) Advice 2: Optimize the Structure of Dataflow

Dataflow construction modification advice is applied after completing the dataflow to optimize the input–output relationships of the existing data. This function is divided into frontend and backend parts. On the frontend, users input two thresholds for high and low correlations, preset to 0.5 and 0.3, respectively, but adjustable as needed. These thresholds determine whether edges should be added or deleted between nodes based on correlation coefficients. The “csv_data” and “csv_column” items of all nodes in the dataflow are sent to the backend. Additionally, because this advice applies only to AI nodes, the frontend marks the “csv_data” items of AI nodes as “selected_data” and sends them to the backend as well. On the backend, the process begins by parsing the received data into a Pandas DataFrame and calculating the correlation matrix. A heatmap is then generated to visualize the correlations and encoded as a Base64 image. Based on the user-defined thresholds, pairs of nodes with correlation coefficients above the high correlation threshold are identified as candidates for edge addition, while pairs below the low correlation threshold are flagged for edge deletion.

Finally, the backend packages the heatmap image, high-correlation pairs, and low-correlation pairs into a JSON format and returns it to the frontend. The frontend dynamically displays this information in a popup, as illustrated in Figure 7. The upper part of the figure shows the heatmap, while the lower part lists the recommended edges to add or delete. Users can either manually adjust the dataflow according to these suggestions or click the “yes” button to let the AI automatically modify the dataflow structure.

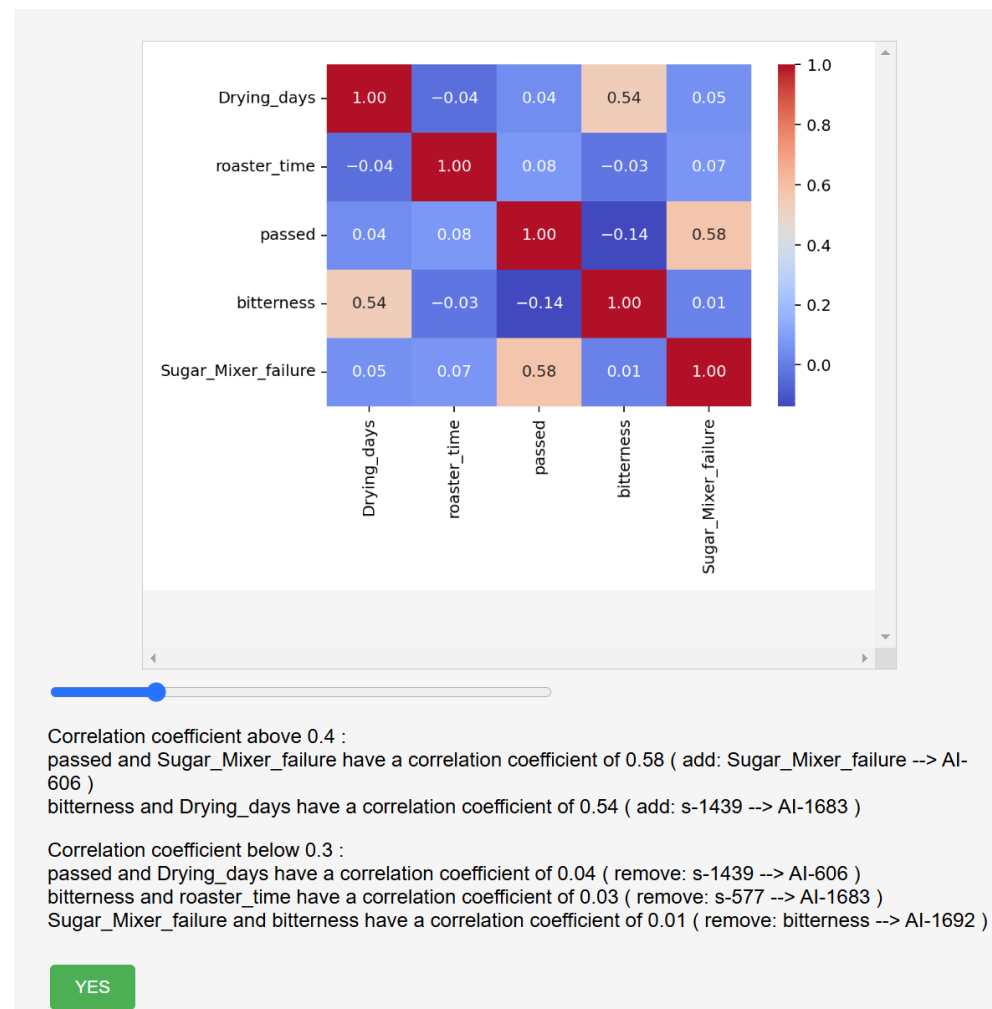


Figure 7. Correlation-based AI recommendations for dataflow structure optimization.

(C) Advice 3: Optimize the Decision Tree Model

The combination of decision trees and dataflow can address the black-box problem. However, decision tree models may overfit, resulting in complex models that perform well on test data but poorly in real-world applications. These issues arise from excessive tree depth, an overabundance of leaf nodes, or irrelevant branches. To mitigate this, we dynamically adjust decision tree parameters based on input and target data.

We proposed and implemented a straightforward yet effective procedure to optimize the maximum number of leaf nodes (max_leaf_nodes) in a decision tree. Specifically, we take as input the range of candidate values for max_leaf_nodes , the feature set X , and the target labels y . For each candidate max_leaf_nodes value, we train and evaluate a decision tree using five-fold cross-validation to obtain the mean performance score. Finally, we select the max_leaf_nodes value that achieves the highest average score. This process helps balance model complexity and accuracy, reducing overfitting while maintaining strong predictive capability.

Figure 8 illustrates how the Cross-Validation Score varies with leaf node count, peaking before declining due to overfitting. The optimization algorithm resulted in a decision tree with 10 leaf nodes, improving explainability and achieving an accuracy of 78.7%. This optimization algorithm has been integrated into the AI assistant to provide solutions for individual models.

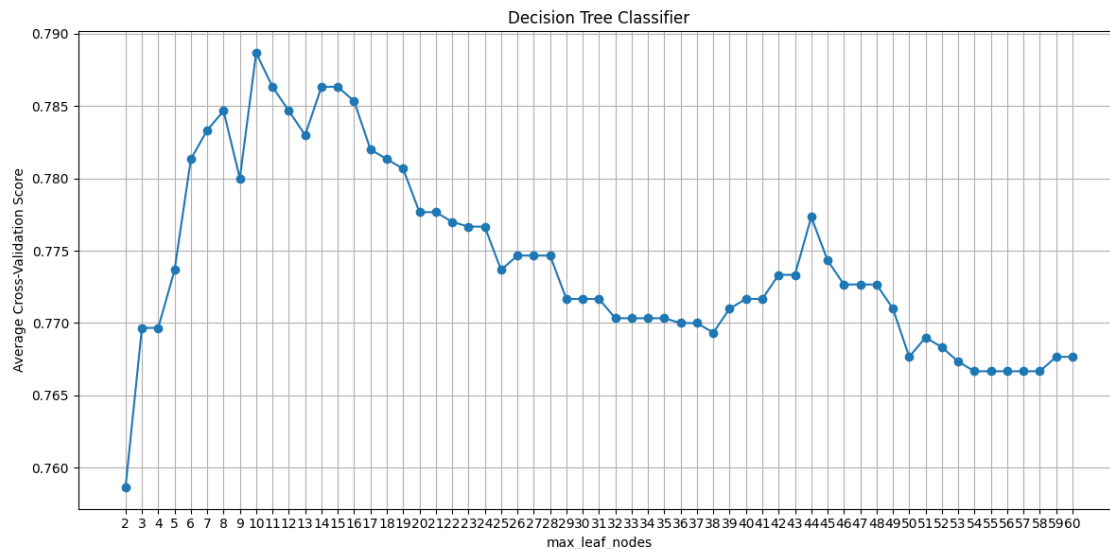


Figure 8. Impact of leaf node count on decision tree Cross-Validation Score.

5. Application Examples

5.1. Chocolate Factory Machinery Failure Prediction

In the experiment, we will use SynergyAI to create a factory machinery failure prediction model to identify the causes of failures. The benefits of predicting machinery failures in factories include improved production efficiency, reduced maintenance costs, minimized downtime, and extended equipment lifespan. By predicting failures, factories can schedule maintenance in advance, avoiding the production interruptions and losses caused by unexpected downtime. This process is closely related to the IoT, as IoT sensors can monitor equipment status in real time and collect vast amounts of data. By analyzing these data, AI models can identify potential signs of failure and enable predictive maintenance, leading to smarter and more efficient factory operations.

Moreover, identifying the causes of machinery failures and using them as a basis for prediction can significantly enhance prediction accuracy, optimize maintenance schedules, reduce downtime, extend equipment lifespan, and achieve optimal resource allocation and data-driven decision-making. This, in turn, further enhances the overall production efficiency and economic benefits of the factory.

Dataset Introduction

This experiment uses a dataset from a chocolate factory production line, as described in the research by Zhang et al. [10], with Table 3 showing a portion of the data. Due to the large number of columns in the dataset, only a subset of the data is displayed in the table. The dataset mainly includes three types of data:

- **Sensor Data:** data from sensors on factory machinery, including temperature, rotational speed, time, etc.
- **Quality Inspection Data:** data from the inspection of finished chocolates, including glossiness, sweetness, qualification status, etc.

- Machinery Failure Data: records of actual mechanical inspections, indicating whether the machinery failed.

Table 3. Production data.

Pump3_Pressure	Conching_Time	Sweetness	Passed	Roaster_Failure	Sugar_Mixer_Failure
63	224	1	0	0	0
58	182	2	0	0	0
64	249	2	0	0	0
64	255	3	0	0	0
61	192	3	0	0	0
72	359	3	1	0	0

Create Dataflow Model with AI Assistant

Figure 9 shows the chocolate factory production line, illustrating the relationships between various machines. In this experiment, we aim to predict the failure of the Sugar Mixer machine shown in the diagram.

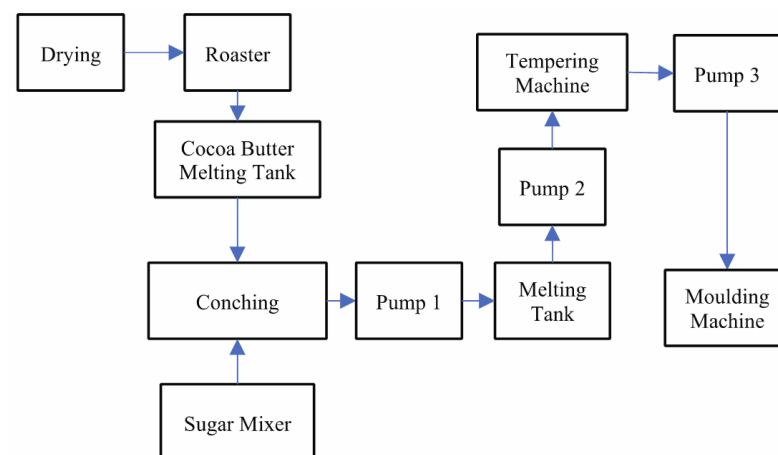


Figure 9. Chocolate production line diagram.

First, as illustrated in Figure 10, we used the AI assistant’s data selection suggestions for individual AI nodes to identify which data are correlated with “Sugar Mixer failure”. The results indicated that this machine’s failure is correlated with failures of other machines. However, since we are only predicting the Sugar Mixer failure in this experiment, we did not select other machines’ failure data.

After further analyzing the sensor data and chocolate quality data, we found that although the failure of the Sugar Mixer was not highly correlated with the sensor data, it did show a certain correlation with the chocolate quality data, particularly “passed” (qualification status) and “glossiness”. Therefore, we selected “passed” and “glossiness” as the second-layer data and further searched for data that had a strong correlation with them, from which we selected relevant sensor data.

The following principles are valid for selecting input data for the third layer.

1. Select only sensor data.
2. Preferentially select sensor data from the target machine and other machines that have strong operational dependencies on that machine.
3. Use the selected data as input for the second layer.

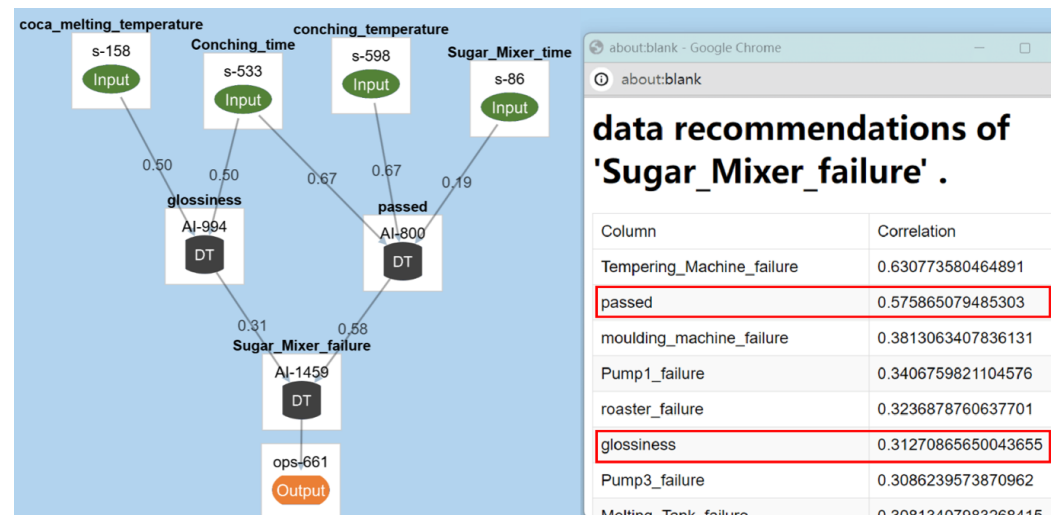


Figure 10. Create the dataflow based on Advice 1.

Following these principles, we constructed the dataflow shown on the left side of Figure 10.

Next, as shown in Figure 11, we used the AI assistant's recommendations to optimize the dataflow structure. The AI assistant provided multiple modification suggestions, as shown on the right side of Figure 11. Based on these suggestions, we observed that there was generally a high correlation between the second and third layers. We modified the dataflow based on the following principles.

1. Select the data with the highest correlation with the second-layer variables from the AI assistant's suggestions and establish connections accordingly.
2. Retain sensor data connections related to the target machine (Sugar Mixer in this case) since the goal is to predict its failure.
3. Train a decision tree model and analyze its structure. If certain input features are not utilized in decision-making, remove their corresponding connections to simplify the model.

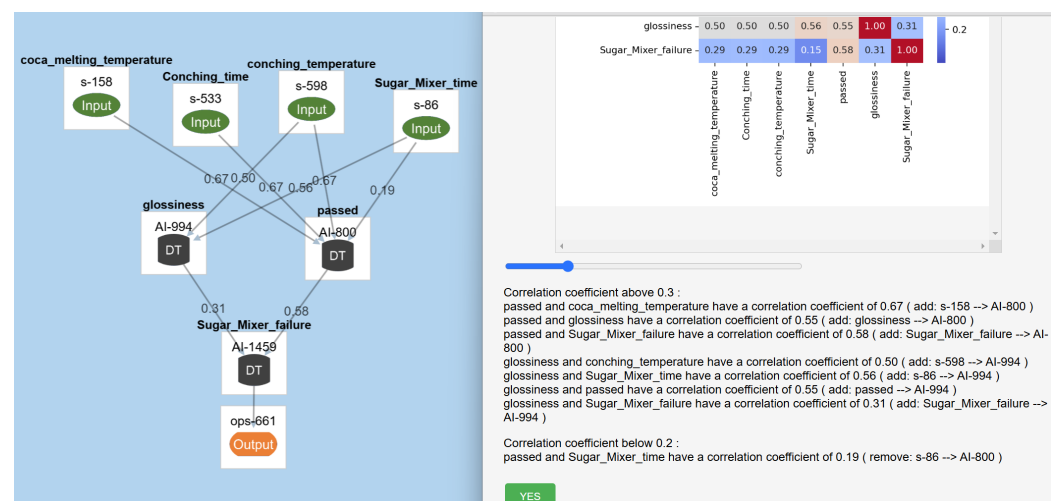


Figure 11. Modify the dataflow based on Advice 2 and user experience.

As a result, we obtained the dataflow shown on the left side of Figure 11.

Lastly, we applied the AI assistant's decision tree optimization suggestions to refine the decision tree structures of each AI node. As shown in Figure 12, after optimization, the

prediction accuracy for “Sugar Mixer failure” reached 85%, while the number of decision tree nodes was reduced from 11 to 9.

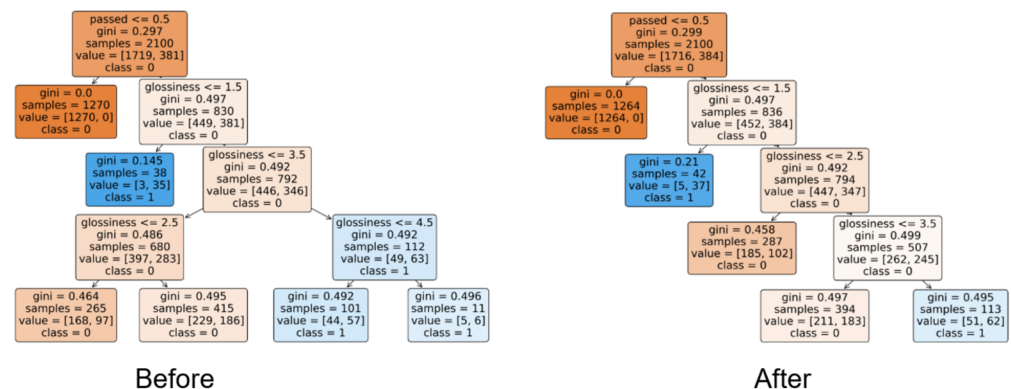


Figure 12. Optimize the decision tree based on Advice 3.

5.2. Evaluation

This experiment utilized various AI assistant functionalities of SynergyAI to create a white-box dataflow model, achieving 85% accuracy in predicting Sugar Mixer failure. This not only demonstrates SynergyAI’s capability in constructing multi-layer models through human–AI collaboration but also highlights the benefits of high model interpretability.

As shown in Figure 13, the model predicts chocolate quality using sensor data from the two machines adjacent to the Sugar Mixer—conching and cocoa butter melting tank—and then uses the chocolate quality to further predict Sugar Mixer failure. In the future, if anomalies are detected in the sensor data from these two machines, we can anticipate specific issues in the chocolate product, allowing for targeted inspections of the Sugar Mixer and early identification of potential failures.

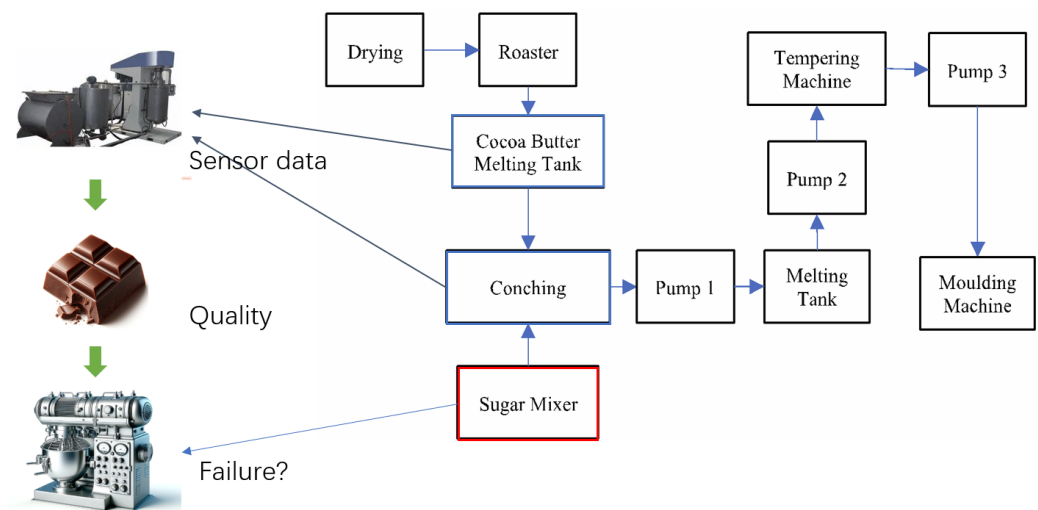


Figure 13. Prediction for chocolate production line

6. Conclusions

This study developed the human–AI collaborative model design tool SynergyAI to address the black-box problem. By using dataflow and visualized decision trees, the tool enhances model explainability and improves the efficiency of model building through human–AI collaboration. Human experts provide domain knowledge and data annotation, while AI handles complex calculations and pattern recognition, enabling effective collaboration.

The proposed algorithms optimize decision tree parameters to adapt to diverse datasets and scenarios, ensuring reliable predictions. Additionally, the ensemble learning mechanism combines predictions from multiple models to reduce errors and enhance overall performance, effectively mitigating overfitting issues.

SynergyAI's AI assistant provided valuable recommendations during data selection, dataflow construction, and model optimization in this experiment. This process reduced trial and error, effectively combining human intuition with AI precision to build a robust and reliable predictive model for chocolate factory machinery failure. The experiment demonstrated SynergyAI's versatility and practical application potential. The tool not only improved prediction accuracy, achieving 85%, but also provided explainable results, which are crucial for decision-making in real-world industrial applications.

By offering a transparent and intuitive platform, SynergyAI fosters further advancements in AI technology and promotes its adoption across various domains. With SynergyAI, users can develop more accurate and reliable models, driving better outcomes in health-care [33], manufacturing, and beyond.

Author Contributions: Conceptualization, L.J. and S.Y.; methodology, L.J.; software, L.J.; validation, L.J., S.Y. and M.A.B.A.; formal analysis, L.J.; investigation, L.J.; resources, L.J.; data curation, L.J.; writing—original draft preparation, L.J.; writing—review and editing, L.J. and S.Y.; visualization, L.J.; supervision, S.Y. and M.A.B.A.; project administration, S.Y. and M.A.B.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research is partially supported by Interface Corporation, Japan.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The experiment was conducted using a human–AI pair programming tool based on dataflow [34] and the experiment data are available in the Chocolate Factory Dataset [35].

Conflicts of Interest: Authors has received research grants from Interface Corporation. The funder had no role in study design, data collection, data analysis, data interpretation, or the writing of the report.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
IoT	Internet of Things
DT	Decision Tree
RF	Random Forest
DL	Deep Learning

References

1. Esteva, A.; Kuprel, B.; Novoa, R.; Ko, J.; Swetter, S.; Blau, H.; Thrun, S. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **2017**, *542*, 115–118. [[CrossRef](#)] [[PubMed](#)]
2. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2023**, arXiv:1706.03762.
3. Shin, C.; Moon, J. A Gradient Based Adversarial Example Method Using Attention Map Against Text Transformer Model. *J. Digit. Contents Soc.* **2021**, *22*, 2019–2026. [[CrossRef](#)]
4. Chen, C.; Shu, K. Combating misinformation in the age of LLMs: Opportunities and challenges. *AI Mag.* **2024**, *45*, 354–368. [[CrossRef](#)]
5. Goodall, N. Machine Ethics and Automated Vehicles. In *Machine Ethics and Automated Vehicles*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 93–102.

6. Storey, V.; Lukyanenko, R.; Maass, W.; Parsons, J. Explainable AI. *Commun. ACM* **2022**, *65*, 27–29. [CrossRef]
7. Zhou, Z.-H.; Wu, J.; Tang, W. Ensembling neural networks: Many could be better than all. *Artif. Intell.* **2002**, *137*, 239–263. [CrossRef]
8. Molnar, C. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*; Lulu Press: Raleigh, NC, USA, 2022.
9. Liang, J.; Yang, C.; Myers, B. A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges. In Proceedings of the 46th IEEE/ACM International Conference on Software Engineering (ICSE), Ottawa, ON, Canada, 26 April–3 May 2023; pp. 616–628.
10. Zhang, D.; Bin Ahmadon, M.; Yamaguchi, S. Human-AI Pair Programming by Data Stream and Its Application Example. In Proceedings of the 2022 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Yeosu, Republic of Korea, 26–28 October 2022; pp. 1–4.
11. Kumar, A.; Cuccuru, G.; Grüning, B.; Backofen, R. An accessible infrastructure for artificial intelligence using a Docker-based JupyterLab in Galaxy. *GigaScience* **2023**, *12*, giad028. [CrossRef] [PubMed]
12. Krizhevsky, A.; Sutskever, I.; Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the 26th Annual Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012.
13. Lundberg, S.M.; Lee, S.I. A unified approach to interpreting model predictions. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 10, pp. 4765–4777.
14. Tang, X.; Li, X.; Ding, Y.; Song, M.; Bu, Y. The pace of artificial intelligence innovations: Speed, talent, and trial-and-error. *J. Inf.* **2020**, *14*, 101094. Available online: <https://www.sciencedirect.com/science/article/pii/S1751157720301991> (accessed on 23 February 2025). [CrossRef]
15. Liu, X.; Song, H.; Liu, A. Intelligent UAVs Trajectory Optimization from Space-Time for Data Collection in Social Networks. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 853–864. [CrossRef]
16. Scibilia, A.; Pedrocchi, N.; Fortuna, L. Modeling Nonlinear Dynamics in Human–Machine Interaction. *IEEE Access* **2023**, *11*, 58664–58678. [CrossRef]
17. Jiang, L.; Bin Ahmadon, M.; Zhang, D.; Fukada, T.; Yamaguchi, S. SynergyAI: An Human-AI Pair Programming Design Tool Based on Program Net. In Proceedings of the 2023 11th International Conference on Information and Education Technology (ICIET), Fujisawa, Japan, 18–20 March 2023; pp. 210–214.
18. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A system for large-scale machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016.
19. PyCaret. PyCaret: An Open-Source, Low-Code Machine Learning Library in Python. Available online: <https://github.com/pycaret/pycaret> (accessed on 16 July 2024).
20. National Instruments. LabVIEW: Graphical System Design Software. 2023. Available online: <https://www.ni.com/en-us/shop/labview.html> (accessed on 12 February 2025).
21. PTC Inc. ThingWorx: Industrial IoT Solutions. 2023. Available online: <https://www.ptc.com/en/products/thingworx> (accessed on 12 February 2025).
22. MathWorks. Simulink. 2023. Available online: <https://www.mathworks.com/products/simulink.html> (accessed on 12 February 2025).
23. Bin Ahmadon, M.A.; Yamaguchi, S.; Mahamad, A.K.; Saon, S. Physical Device Compatibility Support for Implementation of IoT Services with Design Once, Provide Anywhere Concept. *Information* **2021**, *12*, 30. [CrossRef]
24. Doshi-Velez, F.; Kim, B. Towards a rigorous science of interpretable machine learning. *arXiv* **2017**, arXiv:1702.08608
25. Topol, E. High-performance medicine: The convergence of human and artificial intelligence. *Nat. Med.* **2019**, *25*, 44–56. [CrossRef] [PubMed]
26. Davenport, T.; Ronanki, R. Artificial intelligence for the real world. *Harv. Bus. Rev.* **2018**, *96*, 108–116.
27. Luther, T.; Kimmerle, J.; Cress, U. Teaming Up with an AI: Exploring Human–AI Collaboration in a Writing Scenario with ChatGPT. *AI* **2024**, *5*, 1357–1376. [CrossRef]
28. Shannon, P.; Markiel, A.; Ozier, O.; Baliga, N.; Wang, J.; Ramage, D.; Amin, N.; Schwikowski, B.; Ideker, T. Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Res.* **2003**, *13*, 2498–2504. [CrossRef]
29. Franz, M.; Lopes, C.; Huck, G.; Dong, Y.; Sumer, O.; Bader, G. Cytoscape.js: A graph theory library for visualisation and analysis. *Bioinformatics* **2016**, *32*, 309–311. [CrossRef]
30. Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* **2019**, *1*, 206–215. [CrossRef]
31. Zhou, Z. *Ensemble Methods: Foundations and Algorithms*; Chapman: Orange, CA, USA, 2012.
32. Lee, Z.-J.; Cai, J.-X.; Wang, L.-H.; Yang, M.-R. Ensemble Algorithm Based on Gene Selection, Data Augmentation, and Boosting Approaches for Ovarian Cancer Classification. *Diagnostics* **2024**, *14*, 2772. [CrossRef]

33. Skaria, A.P. The economic and societal burden of Alzheimer disease: Managed care considerations. *Am. J. Manag. Care* **2022**, *28* (Suppl. S10), S188–S196. [[CrossRef](#)] [[PubMed](#)]
34. Yatusiro. *Human-AI Pair Programming Tool Based on Dataflow*. GitHub Repository. Available online: <https://github.com/yatusiro/SynergyAI241216> (accessed on 16 December 2024).
35. Yamaguchi. Chocolate Factory Dataset. (Kaggle, 2024). Available online: <https://www.kaggle.com/datasets/yamaguchi1/chocolate-factory-dataset> (accessed on 23 February 2025).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.