OEE Prediction Final Rep
IBM Hackathon & ML Course

YUCHUN WANG
12/09/2025

1)Introduction

The background of this project is to analyze production line data and establish the correlation between data and Overall Equipment Efficiency (OEE) by leveraging the IBM Watsonx platform.

The core objective is to achieve accurate prediction of production line performance based on real-time data collected by production line sensors.

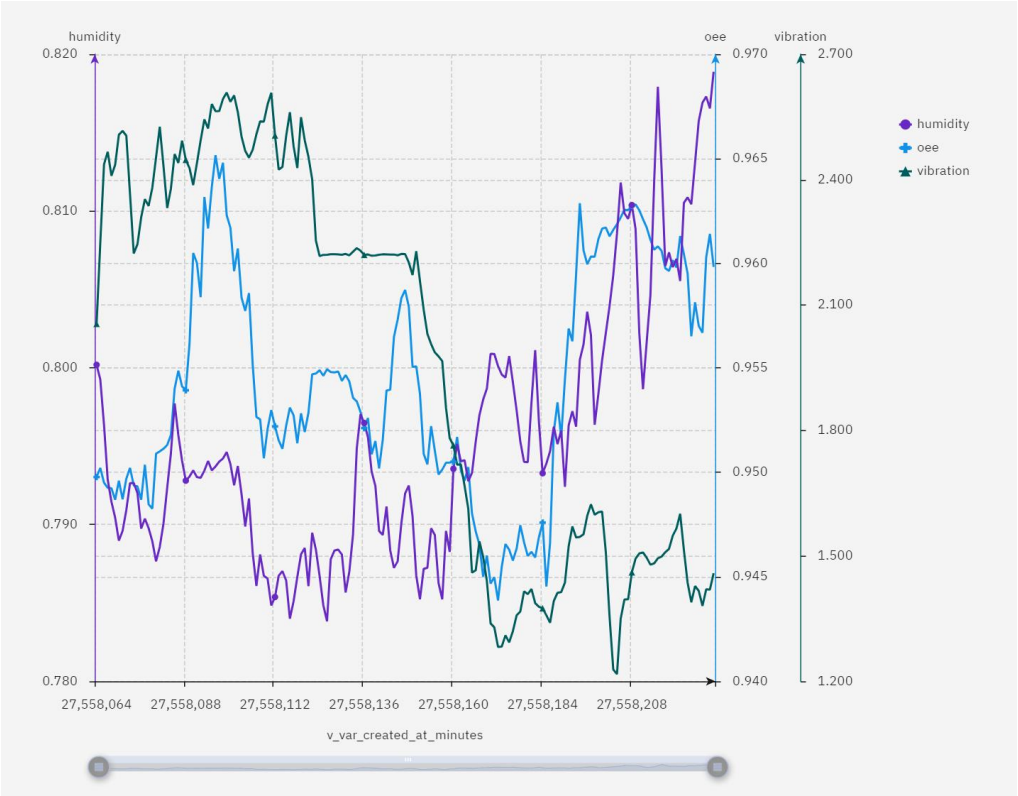
From a machine learning perspective, with production line sensor data (vibration and humidity) as input features, and production line performance indicators (linked to OEE) for the corresponding as output labels, the task is defined as a supervised learning task. By building a model to learn the mapping relationship between inputs and outputs, performance prediction is realized.

2)Key steps

- **Data import and preparation**

The purpose is to make the data structured to facilitate subsequent analysis and exploration. In Watsonx, we use the Data Refinery tool to upload the original scenario_1 data, aggregate it by time, and merge the data in second units into minute units.

- **Exploration and visualisation:**



This graph shows the changes in humidity, OEE (Overall Equipment Efficiency), and vibration over time: In the later stage, as humidity rises, OEE increases sharply synchronously (their positive correlation is obvious); the vibration, which was at a high level in the early stage, decreases in the later stage and its influence on OEE weakens, reflecting that the impacts of vibration and humidity on production efficiency (OEE) exhibit stage - specific differences over time.

- **Feature engineering in Jupyter**

Time lag essentially refers to using equipment data from a specific point in the past as new feature variables. Adding time lag features enables

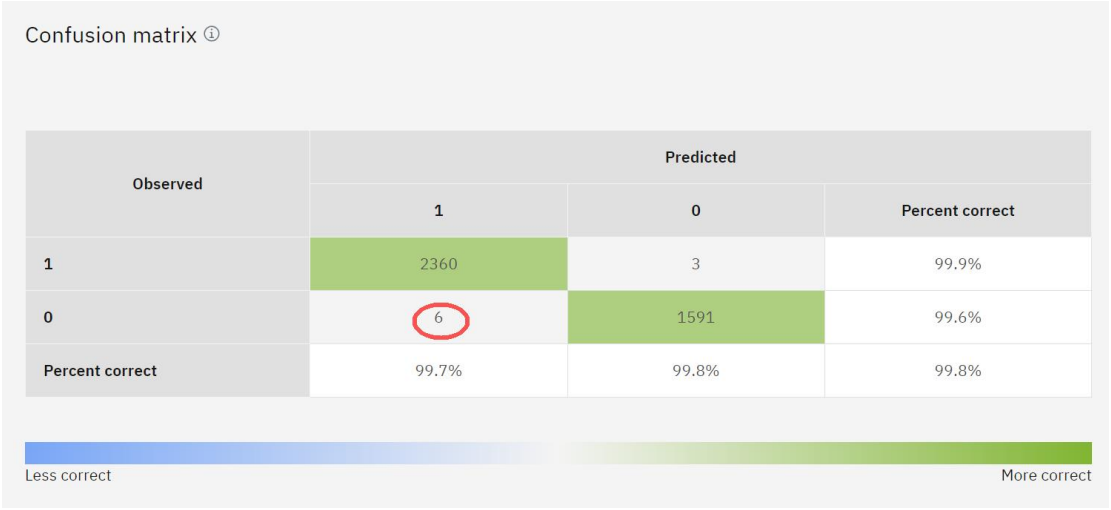
the model to capture the correlation patterns between historical states and future OEE, thereby improving predictive accuracy.

In Jupyter, key1_1m and key2_1m were configured as time lag features. Meanwhile, OEE was categorized into different classes.

- **AutoAI modelling**

AutoAI first limits the algorithm scope based on the "binary classification" task type, then takes the F1 score as the optimization metric. It automatically evaluates and selects the top-performing algorithms from the selected candidate algorithms (such as decision trees, random forests, etc.), generates multiple sets of model pipelines based on the selected top algorithm, and finally selects the model with the highest F1 score on the validation set.

The core principle of the Snap Decision Tree Classifier is to recursively partition the feature space, gradually split the dataset into subsets with higher homogeneity, and finally output categories (e.g., "OEE normal/abnormal") at the leaf nodes. It is suitable for industrial sensor data (such as vibration, humidity), can capture non-linear correlations between features, and features efficient training/inference and strong interpretability.



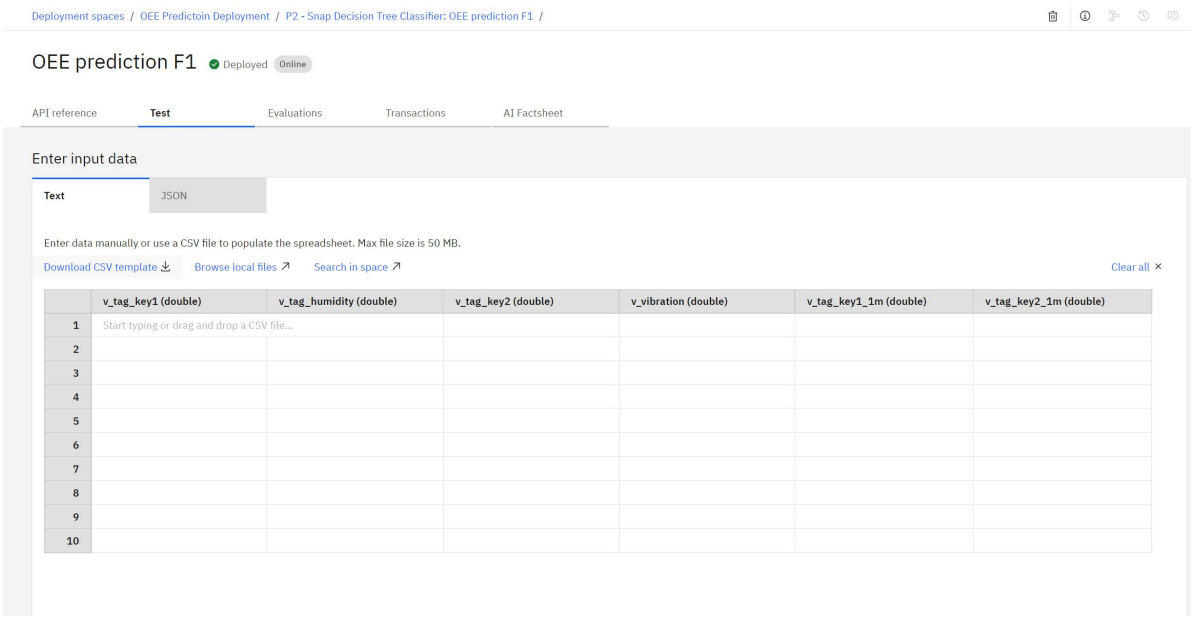
The model achieves an extremely high overall classification accuracy (nearly 99.8%), but there are still a small number of errors — with 6 false positives (FP) and 3 false negatives (FN).

3) Deployment and monitoring:

- **Deployment**

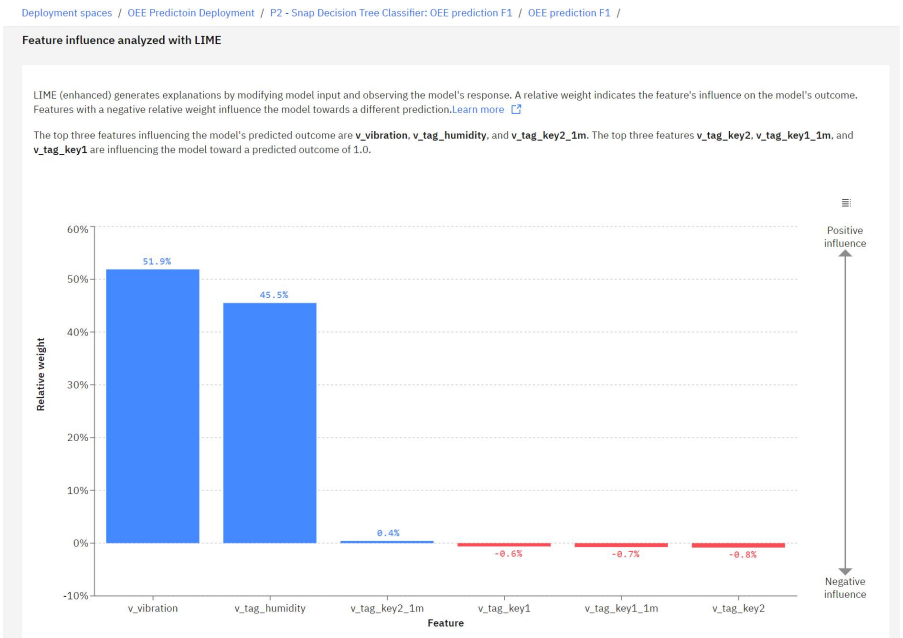
The core function of deployment is to **deploy the trained OEE binary classification model from the development environment to real-world application scenarios in production lines**. This enables the model to receive real-time sensor data and automatically output prediction results of "OEE up to standard/not up to standard", providing a decision-making basis for operation and maintenance personnel to adjust equipment in a timely manner and optimize production, thus truly transforming the model's capabilities into practical value in industrial

scenarios.



After deployment is completed, you can use your model with ease.

- **Monitoring**



During monitoring, LIME will generate intuitive explanation reports for

each OEE binary classification prediction result. For example, when the model determines that "OEE is not up to standard", it will clearly mark key sensor features such as excessive vibration data and abnormal temperature, as well as their impact weights on this classification result, enabling operation and maintenance personnel to quickly identify the core factors affecting equipment efficiency.

4)Conclusion

- **Summarise the entire pipeline.**

The entire process revolves around production line performance prediction, in sequence: import and prepare data using IBM Watsonx, conduct exploratory visualization and feature engineering in Jupyter, perform modeling via AutoAI, and finally complete model deployment and monitoring.

The time proportion of each stage is as follows: data import and preparation (15%), exploratory visualization and feature engineering (25%), AutoAI modeling (25%), and deployment and monitoring (35%).

- **the choice of algorithms selected for each stage.**

In the current production line OEE prediction scenario, choosing the **binary classification** mode simplifies OEE status judgment , which

better aligns with the intuitive management needs for equipment efficiency in industrial scenarios. **Optimizing the F1 score balances** precision and recall in classification tasks, effectively avoiding prediction biases caused by data imbalance and ensuring the model's ability to identify critical states. The selection of the **LIME** tool allows for clear unpacking of the model's prediction logic, enabling operation and maintenance personnel to understand the impact of each sensor data on the OEE classification result. Overall, these choices are highly aligned with the core requirements for model interpretability in industrial scenarios and perfectly match the scenario needs.

- **limitations of the platform**

First, I believe the platform's monitoring dimensions are relatively limited—it focuses heavily on model interpretability but **lacks adequate dynamic adaptability to real-time data fluctuations, failing to integrate deeply with industrial production scenarios.**

Second, more importantly from my personal experience, IBM has a large number of sub-platforms that are not interconnected; many of its derived platforms are **easily confused**. At the initial stage of my project, I even used the wrong platform, which prevented me from completing many tasks. When I attempted to migrate midway, the process was extremely complicated—even the exported project was not recognized

by IBM and could not be parsed, forcing me to start over.

Meanwhile, the algorithm selection in the AutoAI phase is not very intelligent: it automatically chooses to optimize accuracy when binary classification is mentioned, **without analyzing my data requirements, and requires manual adjustments.**

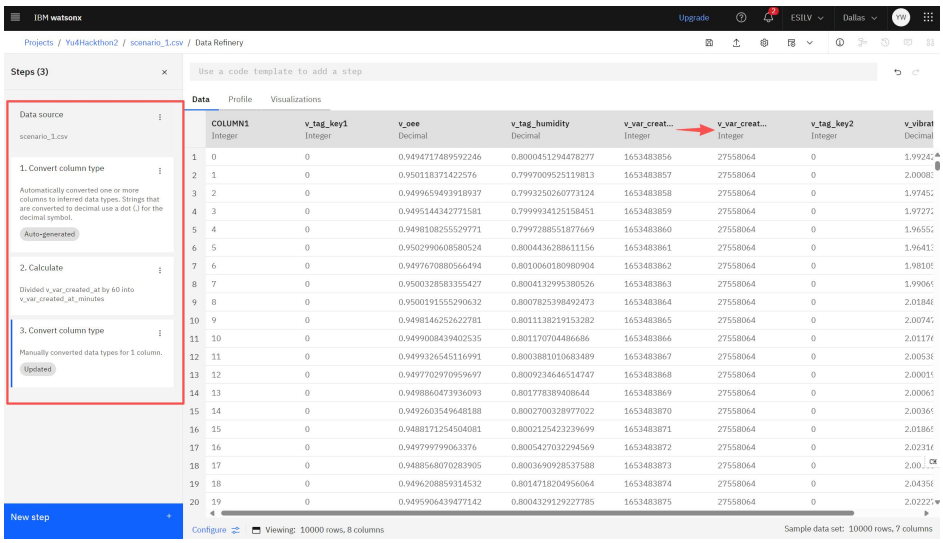
- **I've learned**

The selection of binary classification combined with the F1 score needs to be aligned with the scenario, as it can better meet the needs of OEE status management in industrial settings. For AutoAI modeling, manual parameter intervention is required to avoid insufficient adaptability of default settings to special noise in data. In industrial scenarios, model interpretability must be prioritized; tools such as LIME can help operation and maintenance personnel understand the prediction logic. Tools need to be complementary—for instance, using Jupyter to assist in feature engineering can improve process efficiency.

Appendix I

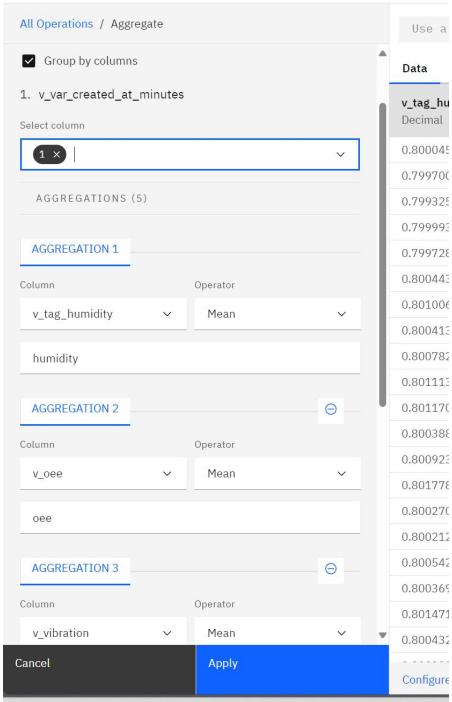
The answer of Lab guide

1.First, you need to access IBM Data Refinery. The first step is to perform calculations on v_var_created_at: divide it by 60 to convert it into minutes. (At this step, I first considered whether I could directly use Data Refinery's tools to extract minutes; unfortunately, this tool can only be used for timestamps.) Then, since the requirement is to include only integers, the second step is to convert the extracted data from decimal to integer type.



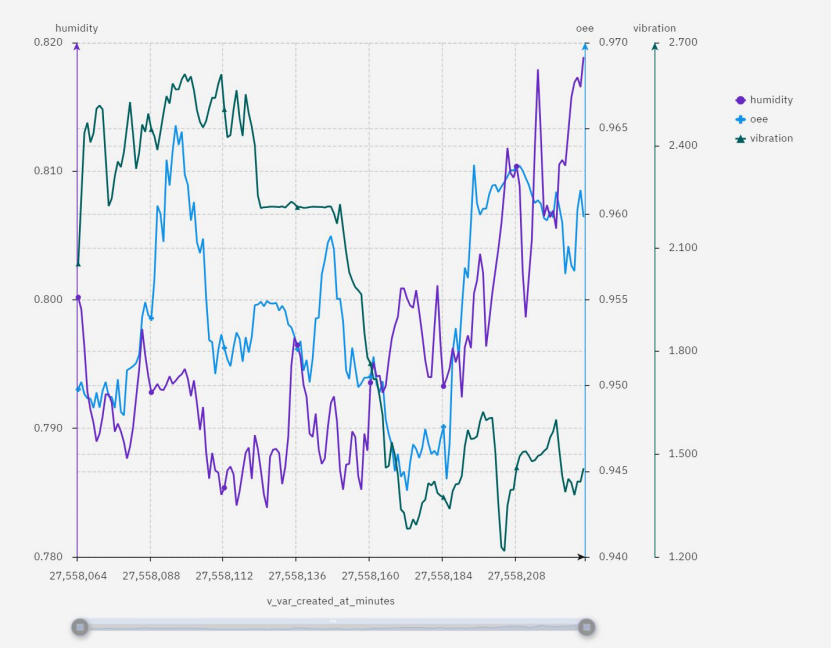
2.

Similarly, using IBM Data Refinery tool, create a new step to aggregate the data: select to group by v_var_created_at_minutes, and then calculate the average of all data.Similarly, using IBM Data Refinery tool, create a new step to aggregate the data: select to group by v_var_created_at_minutes, and then calculate the average of all data.Similarly, using IBM Data Refinery tool, create a new step to aggregate the data: select to group by v_var_created_at_minutes, and then calculate the average of all data.



3.We directly click on the processed shaped data, then select the visualization function in IBM. Next, choose the columns to be visualized, and finally select the most suitable chart for presenting the selected columns.

4.



The fluctuation trends of humidity and OEE show greater synchrony; however, the trend of vibration differs significantly from the first two, with a weak correlation in both rhythm and magnitude.

Early stage: Vibration remained at a relatively high level (close to 2.4–2.7), OEE was relatively low (around 0.945–0.955), and humidity was in a fluctuating state.

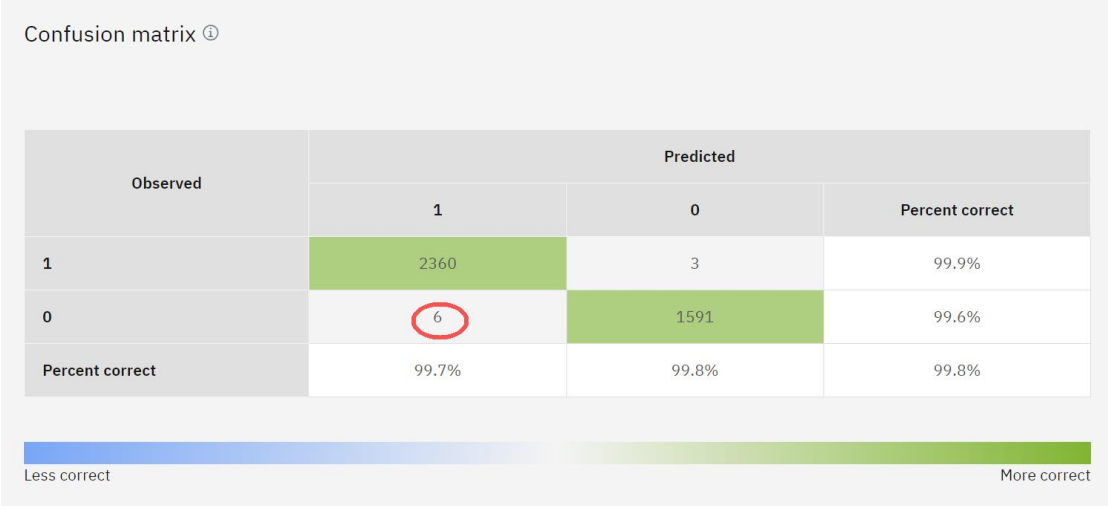
Mid-stage: Vibration decreased significantly (to the range of 1.2–1.5), while OEE and humidity entered their respective fluctuation adjustment phases.

Later stage: Humidity and OEE rose rapidly in synchronization to their peaks. Although vibration rebounded slightly, it did not return to the high range seen in the early stage.

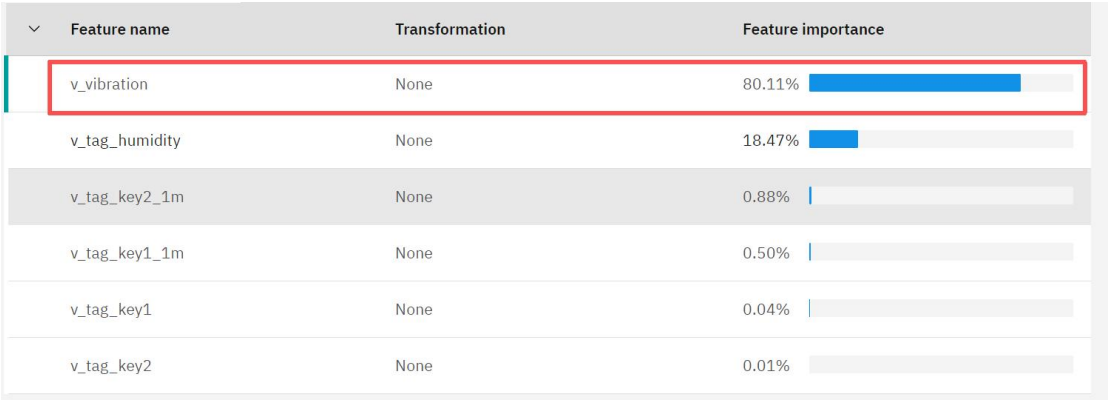
When OEE reached its peak, humidity was also close to its peak; however, vibration was not at its peak during these periods and was even in a relatively low range.

5. I believe that direct classification is more in line with the needs of practical applications. We only need to quickly determine whether intervention is required, rather than needing precise values. Meanwhile, dividing into two categories reduces model complexity. Compared to regression tasks that predict continuous values, using binary classification tasks directly is faster and allows for more intuitive evaluation, enabling rapid verification of the model's effectiveness.

6. we should select scenario_1_training_data.csv, the processed data from the previous step, as the training data. Choose v_oee_h+1_cat as the column to be predicted. **I have selected the F1 score as the metric to be optimized because in the current industrial scenario, my core goal is to avoid missed judgments and misjudgments. The F1 score can balance recall and precision, preventing errors from relying on a single metric.** Here, the platform prioritizes accuracy as the recommended metric; however, in industrial data scenarios, this can easily lead to missing all anomalies.



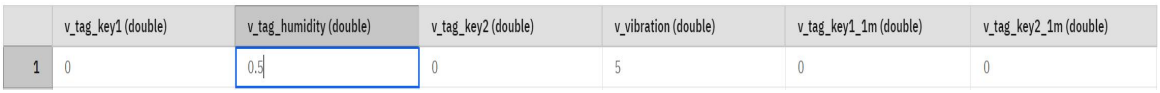
The number of false positives is 6.



Vibration has the greatest contribution.

7. It includes basic information of AI use cases, model metadata, training information, associations with development projects, model performance metrics (training vs. validation), and input features (partial listing).

8.The prediction result is 0, which means the OEE is below 0.9 with a 100% confidence level.
After inputting (the data), the prediction result changes to 1 with a 100% confidence level.



9.We should choose IBM Watson Machine Learning, and the algorithm is regression.

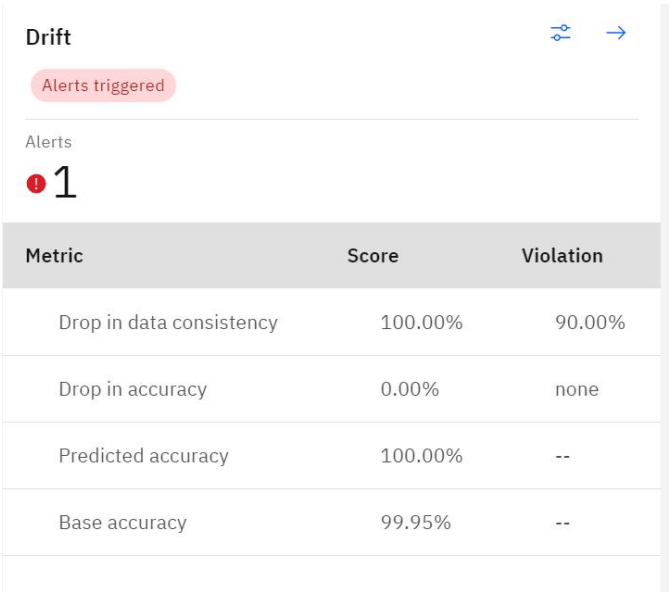
10.I believe LIME focuses more on the local level: it uses simple simulations to mimic the behavior of the black box and identify key features, and it is fast, but it lacks a global perspective.

SHAP is more equitable — it distributes (feature importance) to each feature like dividing a cake, yet it requires significant computational effort.

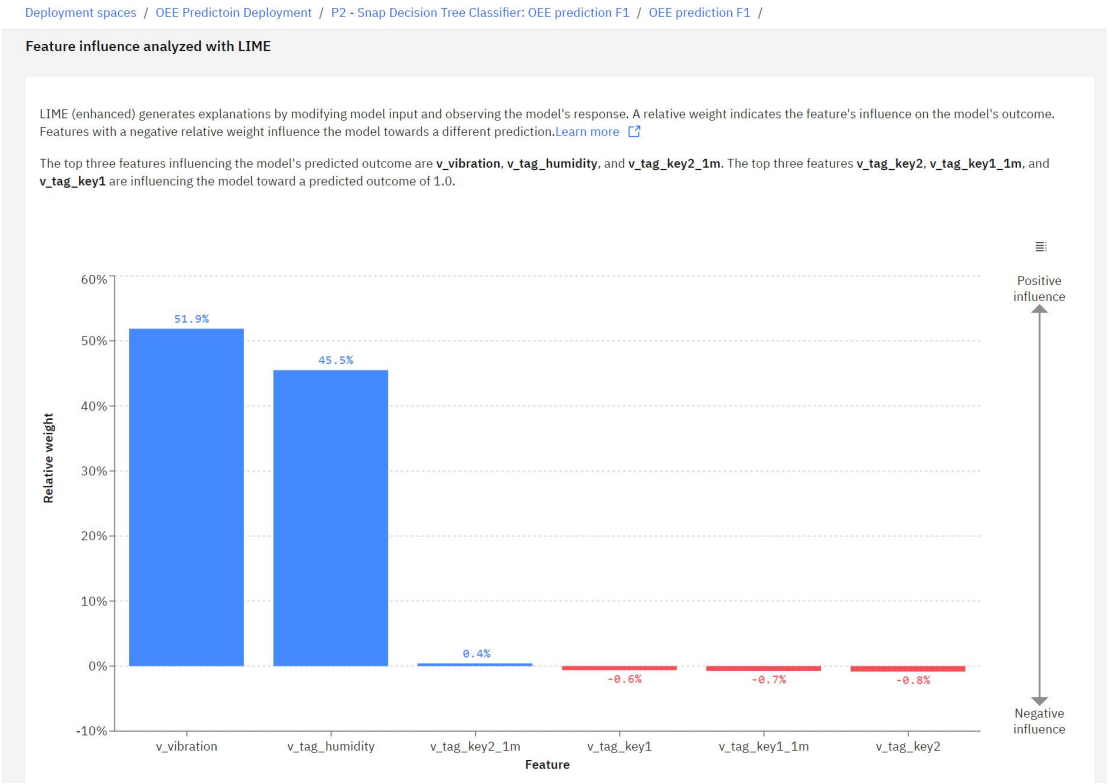
11.v_tag_key1 is the most core one. Alarms are direct signals that a device has experienced/continuously experiences anomalies — once an alarm is triggered, the device is highly likely to be in a state of "unplanned downtime" or "reduced-efficiency operation," and OEE will immediately and significantly decrease. It is the "most direct indicator" of OEE decline and has the highest discriminative power for model classification (high/low OEE).

12.

An alert for drift has been triggered (the interface displays "Alerts triggered" and the number of alerts is 1).
The type of drift that occurred is "Drop in data consistency" (data consistency decline).
The drift of the "Drop in data consistency" indicator is the most severe (its "Score" reaches 100.00%, and "Violation" is 90.00%), making it the only indicator that triggered a violation.



13.



The model's prediction result is 1.0; the variable that contributes the most to the prediction result is v_vibration (its relative weight reaches 51.9%, which has the highest proportion among all features).

Adjust the value of v_vibration from its original value (8.509743784479221) to at least 6.922.

Appendix II

Certificate