Using CRISP-DM to Detect and Prevent Machine Failures

Steven Chang

College of Engineering, San Jose State University

Business Understanding

The failure of a machine in a production or operational environment can have detrimental effects on a company. Such failures can lead to production downtime, resulting in significant revenue loss, increased maintenance costs, potential safety hazards, and negative impacts on company reputation. Therefore, predicting machine failures proactively becomes a critical task. Using the provided training and testing datasets, our aim is to develop a predictive model that can accurately determine whether a machine will fail or not, enabling the company to take preventive measures, optimize maintenance schedules, and ultimately save costs and ensure smooth operations.

Objective

Our primary objective is to achieve the highest accuracy possible while minimizing false positives and false negatives. The reason being, predicting a machine won't fail when it actually will can lead to severe consequences, and predicting a machine will fail when it won't can result in unnecessary maintenance costs. We aim to provide the company with a reliable tool that can help in making informed decisions regarding machinery maintenance.

Data Understanding

We are utilizing a dataset retrieved from Kaggle that details information of machines, their state, and if they have had a failure. The specific features of the dataset are described below:

id: An identifier for each record.
Product ID: Identifier for the product being produced.
Type: Type of the product.
Air temperature [K]: Temperature of the surrounding air.
Process temperature [K]: Temperature of the operational process.
Rotational speed [rpm]: Speed at which the machine is operating.
Torque [Nm]: Torque produced by the machine.
Tool wear [min]: Duration the tool has been in use without replacement.
Machine failure: Binary value indicating if the machine failed (1) or not (0).
TWF: Tool wear failure (binary: 1 if tool wear caused the failure, else 0).
HDF: Heat dissipation failure (binary: 1 if heat caused the failure, else 0).
PWF: Power failure (binary: 1 if power issues caused the failure, else 0).
OSF: Overstrain failure (binary: 1 if overstrain caused the failure, else 0).
RNF: Random failure (binary: 1 if the cause was random/unknown, else 0).

Data Visualizations

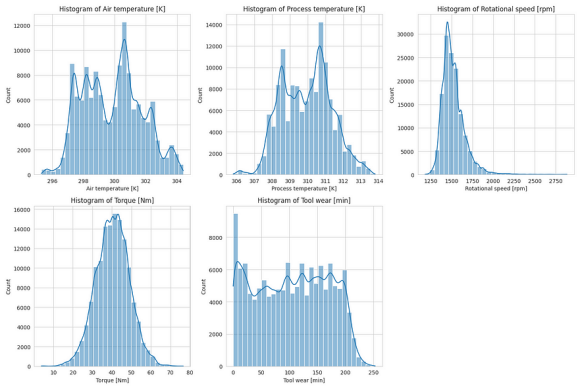

Figure 1. Histograms of Numerical Features

From these visualizations we perceive the following on the distribution and statistics of our features:

Air temperature [K]: The distribution appears to be fairly normal, centered around a temperature of approximately 300K.

Process temperature [K]: This also shows a fairly normal distribution, centered slightly above 310K.
Rotational speed [rpm]: The distribution is somewhat bimodal, indicating that there are two common operational speeds for the machine.
Torque [Nm]: The distribution is slightly right-skewed, with a mode around 30–40 Nm.
Tool wear [min]: The distribution is somewhat uniform with a slight increase as the wear time approaches around 200 minutes, indicating tools are possibly replaced or maintained after a certain duration.

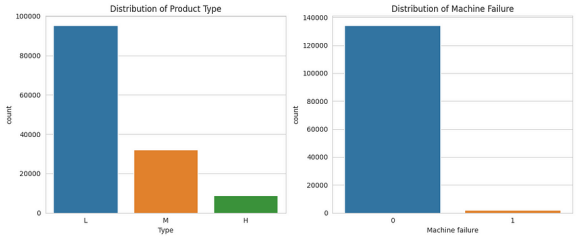

Figure 2. Bar graph of Categorical Features

From these visualization we perceive the following:

Distribution of Product Type: It appears that the majority of products in the dataset belong to type ‘M’, followed by type ‘L’ and a smaller portion of type ‘H’.

Distribution of Machine Failure: The dataset is highly imbalanced with respect to the target variable. A significant majority of the instances indicate no machine failure (Machine failure = 0), while a very small portion indicates machine failure (Machine failure = 1). This imbalance will need to be considered during the modeling phase, as it can influence the predictive performance of the model.

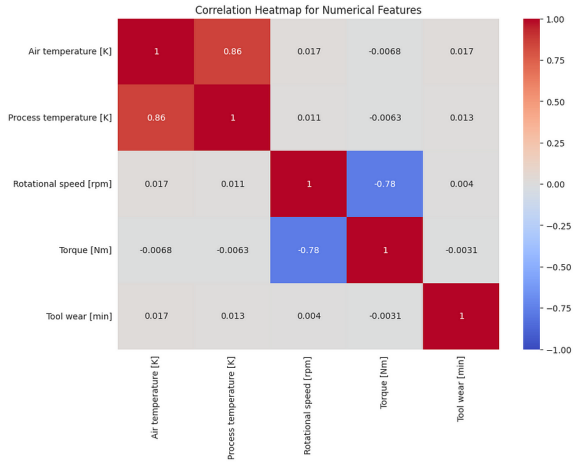

Figure 3. Correlation Heat-Map of Categorical Variables

The correlation heat-map reveals the following insights regarding the relationships among the numerical features:

Air temperature [K] and Process temperature [K]: These two features have a very high positive correlation (approximately 0.87). This suggests that as the air temperature rises, the process temperature also tends to increase, and vice versa.

Rotational speed [rpm] and Torque [Nm]: These features have a noticeable negative correlation (approximately -0.47). This indicates that as the rotational speed of the machine increases, the torque tends to decrease, and vice versa.

Other pairs of features have relatively low correlation values, indicating weaker linear relationships.

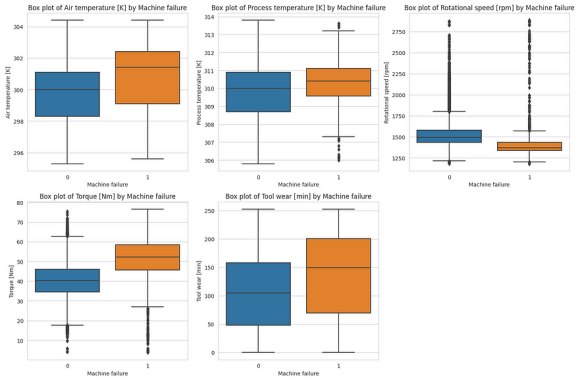

Figure 4. Box Plots Showing Outliers of Categorical Data

The box plots offer insights into the distribution of numerical features relative to machine failure:

Air temperature [K]: There seems to be a slightly higher median air temperature for instances where the machine failed.

Process temperature [K]: Similar to air temperature, the process temperature's median is marginally higher for machine failures.

Rotational speed [rpm]: The median rotational speed seems fairly consistent between both classes (failure and no failure). However, there's a wider range of values when the machine has failed.

Torque [Nm]: There's a noticeable difference in the torque values between the two classes. Machine failures tend to occur at higher torque values.

Tool wear [min]: Machine failures seem to be slightly more common after a longer tool wear duration, as indicated by the higher median for instances where the machine failed.

Summary

The training dataset consists of both numerical and categorical columns, capturing machine operational parameters, product details, and machine failure indicators. We observed the distributions of numerical columns using histograms and understood the categorical column (Type) and target variable (Machine failure) distributions using bar plots. There is a strong positive correlation between air and process temperatures and a notable negative correlation between rotational speed and torque.We visualized how numerical features vary for instances where machines failed versus when they didn't. Features like torque and tool wear duration exhibited differences between the two classes.

Data Preparation

In this stage we must prepare our dataset for use in training a model. To do so we made the following changes to the data:

Encoded Categorical Variables: Utilized one-hot encoding to convert categorical variables into a numerical format that is acceptable for training a model.

Feature Engineering: Given the columns TWF, HDF, PWF, OSF, and RNF that indicate specific reasons for machine failure, we can derive a new feature that represents the total number of specific failures a machine has experienced. This might capture any compounded effects of multiple failure types on the machine's overall failure likelihood.

Data Scaling: Scale numerical variables so they have similar scales, which can help with certain machine learning algorithms. The numerical columns we'll scale are: Air temperature [K], Process temperature [K], Rotational speed [rpm], Torque [Nm], and Tool wear [min]. We'll also scale the newly derived feature Total_Specific_Failures.

Addressing Data Imbalance: The target variable (Machine failure) is imbalanced. Depending on the chosen modeling approach, we might need to address this, e.g., through resampling techniques or by using specific evaluation metrics.

We can use oversampling or under-sampling to balance the dataset. With oversampling we can manually over-sample the minority class by randomly duplicating some of the instances. With under-sampling we can manually under-sample the majority class by randomly removing some of the instances. Given the significant imbalance, we chose to over-sample the minority class, as under-sampling could result in a significant loss of data.

Modeling

Now that we've prepared our data, the next step is to build predictive models. The goal is to find a model that can accurately predict machine failures.

Given the binary nature of our target variable (Machine failure), this is a classification problem. We used our data to build models with the following commonly used classification algorithms:

Logistic Regression: A simple linear classifier that works well as a baseline.

Random Forest: An ensemble learning method that can capture complex patterns. Random Forest is an ensemble learning method that combines multiple decision trees to produce a more accurate and robust prediction. Given its ability to capture complex relationships and its inherent feature importance estimation, it's a popular choice for many classification tasks.

Gradient Boosted Trees (e.g., XGBoost): A boosting algorithm known for its performance in a wide range of classification problems. XGBoost is an optimized distributed gradient boosting library that is efficient and known to provide better predictive accuracy than many other algorithms. It has become a popular choice for many Kaggle competitions and real-world applications.

Logistic Regression Model Results
Accuracy: Approximately 97.42%
F1-score: Approximately 50.32%

The high accuracy indicates that the model is generally classifying instances correctly. However, the moderate F1-score suggests that the model might be struggling to balance precision and recall, especially for the minority class (machine failures).

Random Forest Classifier Results
Accuracy: Approximately 99.48%
F1-score: Approximately 82.47%

The Random Forest model has outperformed the Logistic Regression model, especially in terms of the F1-score, which is quite high now. This indicates a better balance of precision and recall for the minority class (machine failures) compared to the Logistic Regression model.

XGBoost Classifier Results
Accuracy: Approximately 98.89%
F1-score: Approximately 69.6%

The XGBoost classifier has shown a high accuracy similar to the Random Forest model, but its F1-score is slightly lower than the Random Forest's F1-score of approximately 82.47%. This suggests that, in terms of balancing precision and recall for the minority class (machine failures), the Random Forest model is currently performing slightly better.

Evaluation

Model Performance

Logistic Regression: Accuracy: ~97.42% F1-score: ~50.32%
Random Forest: Accuracy: ~99.48% F1-score: ~82.47%
XGBoost: Accuracy: ~98.89% F1-score: ~69.6%

Results

The Random Forest classifier has achieved the highest F1-score among the models, making it the top performer in terms of balancing precision and recall. Given our business problem of predicting machine failures, where false negatives (failing to predict an impending failure) could be costly, a high F1-score is crucial.

XGBoost has shown competitive performance, especially in terms of accuracy, but its F1-score is lower than the Random Forest's. However, XGBoost offers more flexibility for tuning and might be improved further with hyper-parameter optimization

Logistic Regression, being a simpler linear model, has the lowest F1-score but served as a useful baseline.

Evaluation Summary

Considering the business implications of machine failures, the Random Forest model currently seems the most suitable, given its high F1-score. However, if computational resources and time permit, we could further optimize the XGBoost model with hyperparameter tuning to see if it can surpass Random Forest's performance.

Deployment

Deploying a model makes it available for practical, real-world use, either by integrating it into existing systems or making it accessible for on-demand predictions.

The following are common deployment strategies, that will be useful for our use case, and their considerations:

Batch Predictions

Instead of on-the-fly predictions, the model processes data in batches. This is useful if real-time predictions aren't necessary. Considering the importance of machine maintenance in preventing costly disruptions, deploying the predictive model using Batch Predictions is a practical approach for scenarios where real-time alerts are not crucial.

Batch Predictions would allow the company to process data from machines at regular intervals, such as at the end of a shift or daily. This method will generate a list of machines that are predicted to fail in the upcoming period. Technicians can start their day by reviewing this list, enabling them to prioritize their maintenance tasks and allocate resources more efficiently. Given that only technicians need access to the results, the predictions can be delivered via a simple interface, such as a daily report or a dedicated dashboard, listing machines at risk and their predicted failure probabilities.

Embedded Systems:

For devices or machinery, the model might be deployed directly onto the hardware. If the machines in question have onboard computing capabilities and can host the predictive model, an Embedded Systems deployment would offer distinct advantages.

The model can operate directly on the machine, eliminating the need to transmit data to a central server. This not only reduces data transmission costs but also addresses potential data privacy or security concerns.Predictions can be stored locally on the

machine and accessed by technicians as needed, perhaps via a connected device or interface. This ensures that only authorized personnel, such as technicians, can view the predictions. Since the model operates directly on the machine, it can be optimized for the specific hardware, ensuring efficient and fast predictions.

Feedback Loop:

After deploying, it's beneficial to have a system that collects feedback on the model's predictions. This real-world feedback can be used to further refine and retrain the model. Regardless of the primary deployment method, establishing a Feedback Loop is crucial for continuous model refinement.

After technicians perform their maintenance tasks, they can validate the model's predictions based on their findings. For instance, if a machine was predicted to fail but was found to be in good condition, this feedback can be recorded. This real-world feedback is invaluable for retraining and refining the predictive model, ensuring its accuracy and reliability improve over time. A simple interface, perhaps integrated into the technician's maintenance reporting tool, can facilitate the collection of this feedback. The process should be streamlined to minimize any additional workload for the technicians.

By adopting these recommendations, the company can ensure efficient maintenance planning, reduce machine downtime, and continuously enhance the predictive capabilities of the model, all while ensuring that only the relevant technicians have access to the predictions.