AI Powered Predictive Maintenance

Eric Aschari, Chenle Chen, Chelsey Tao, Minghui Zhu

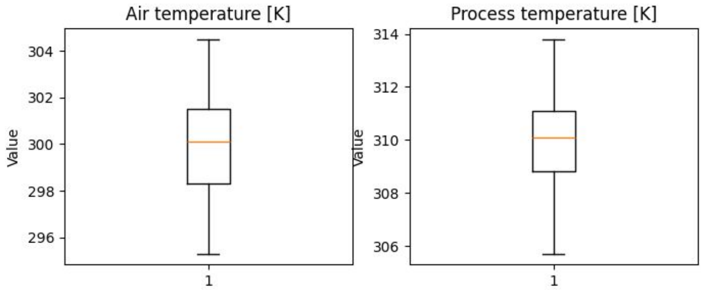
Fall 24

Task Description

- **Our goal:** build a multi-class classifier to predict machine states—either 'no failure' or one of five specific failure types.
- **Why** is this **important?** Predictive maintenance minimizes costly downtime in automated production lines, a critical challenge in Industry 4.0.
- **Dataset:** We’re using the AI4I 2020 Predictive Maintenance Dataset.
- Our **evaluation metrics**? Precision, recall, F1-score, and Matthews correlation coefficient.

Data Preparation

1. Exploratory Data Analysis



- Verify **no missing values**.
- Visualize data using boxplots (numeric features) and bar plots (categorical data).
- Analyzed feature distributions and key patterns, including correlations (using pandas).

2. Label Restructuring:

- Instances with simultaneous failures combined to **"Multiple failures"** (self-implemented).
- Seven classes: **No Failure + 5 failure types + Multiple Failures**.

3. Handling Class Imbalance:

- Addressed imbalance: 96.5% of data labeled as "No Failure")
  - **Stratified sampling** (using sklearn).
  - **Class-weighted loss functions** during training (using sklearn).

4. Preprocessing Steps

- **Feature scaling** applied to numeric features (using sklearn).
- **Binning** of continuous variables for decision trees (self-implemented).
- **One-hot encoding** used for categorical features (using sklearn).

Data Preparation

Numeric features

Air temperature	Process temperature	Rotational speed	Torque	Tool wear

Categorical features

Type (L, M, H)

Individual Failure Types

Tool Wear Failure	Heat Dissipation Failure	Power Failure	Overstrain Failure	Random Failures

Correlations

- 87.6% correlation between Air temperature and Process temperature
- -87.5% correlation between Torque and Rotational speed

→ No features excluded as correlation below 90%

Performance Metrics

Metric	Description
Accuracy	Proportion of correctly classified instances out of the total instances.
Precision	Proportion of correctly classified positives.
Recall	Proportion of correctly classified actual positives.
F1-Score	Harmonic mean of precision and recall.
Matthews Correlation Coefficient (MCC)	Balanced measure of the correlation between predicted and actual labels, considering all confusion matrix elements, even for imbalanced datasets.

→ Evaluation using Matthews Correlation Coefficient is crucial due to large label imbalance in dataset!

Naive Classifier - Baseline (self-implemented)

Classification type

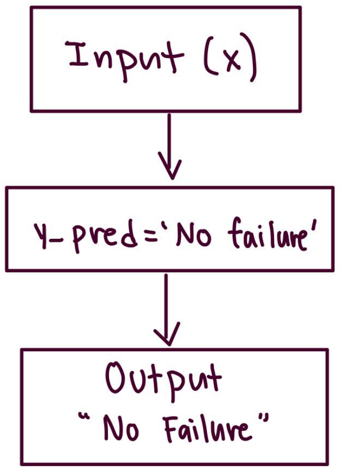
- Naive classifier → always predicts ‘No failure’

Trainable parameters

- None

Hyperparameters

- None



Results

Metric	Test Score
Accuracy	0.965
Precision	1.0
Recall	0.965
F1-Score	0.982
Matthews Correlation	0.0

Interpretation

- Naive classifier reaches 96.5% accuracy!
  → Strong accuracy baseline
- Naive classifier only classifies as ‘No failure’
  → 100% precision due to no false positives
- Matthews correlation of 0.0
  → Predictions by naive classifier are no better than random guessing

Decision Tree based Classifier (self-implemented)

Classification type

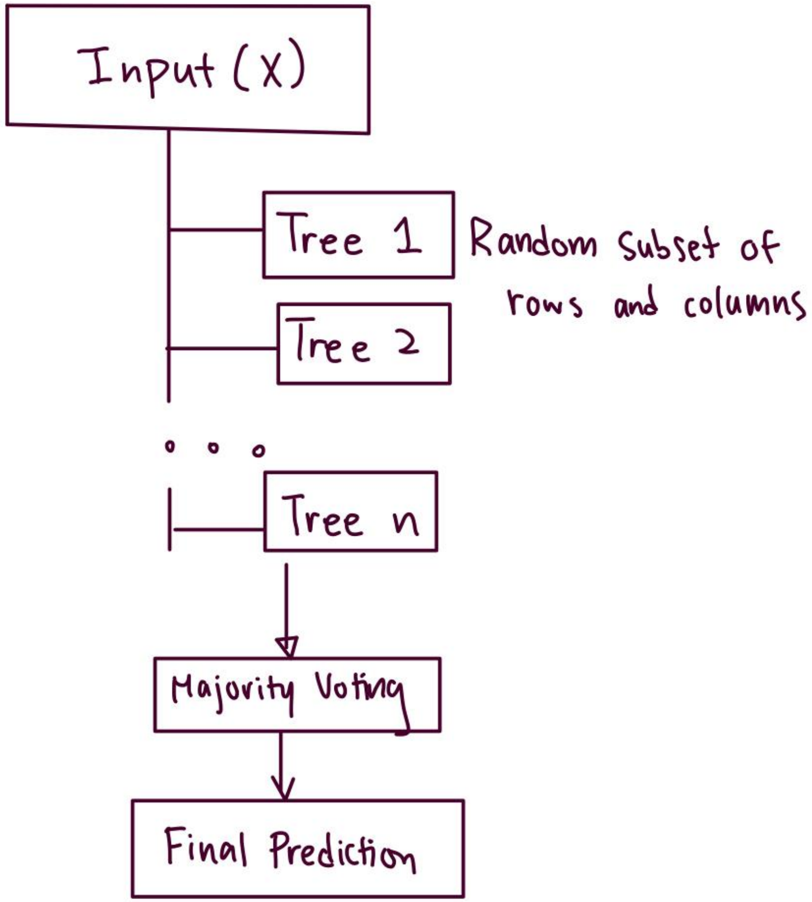
- Decision trees

Trainable parameters

- Tree structure

Hyperparameters

- Binning of continuous feature values
  → Equal width binning implemented
- Number of decision trees for Random Forest (RF)
  → 100 trees
- Range of rows and columns to use for RF
  → minimum 500 examples
  → minimum 4 attributes
- Method against overfitting
  → Pruning



Decision Tree based Classifier - Results

Decision Tree Results

Metric	Test Score
Accuracy	0.968
Precision	0.981
Recall	0.968
F1-Score	0.974
Matthews Correlation	0.441

Random Forest Results

Metric	Test Score
Accuracy	0.966
Precision	0.997
Recall	0.966
F1-Score	0.981
Matthews Correlation	0.191

Interpretation

- Decision tree reaches a F1-score of 97.4% while having a Matthews correlation of 0.441
  → No random guessing!
- Random Forest has a lower Matthews correlation coefficient than decision tree
  → Due to random sampling of examples
  → Should implement stratified sampling for RF

Comparison with Sklearn Random Forest

- Sklearn Random Forest reaches an F1-score of 98.9% with a Matthews correlation coefficient of 0.755
  → Decision boundaries for continuous variables is a treated as a trainable parameter instead of a hyperparameter
  → Includes methods for smart sampling of examples

K-Nearest Neighbors (using Sklearn)

Euclidean Distance: $d(x_i, x_j) = \sqrt{\sum_{k=1}^{n}(x_{ik} - x_{jk})^2}$

Classification type

- K-Nearest Neighbors - Multi-Class Classification, Instance-based learning

Trainable parameters

- None

Hyperparameters

- Number of Neighbors → k = 3
- Distance Metric → Euclidean Distance
- Weighting of neighbors (weights) → Default (uniform)
- Algorithm → “auto”, automatically selects optimal search algorithm

KNN Results

Metric	Test Score
Accuracy	0.969
Precision	0.984
Recall	0.969
F1-Score	0.975
Matthews Correlation	0.415

Interpretation

- High performance (Accuracy → 96.9%)
- High F1-score: balance between precision and recall
- Moderate Matthews Correlation (41.5%)

Feed Forward Neural Network (built using Torch)

Classification type

- Multi-Class Classification → Feed Forward Neural Network

Trainable parameters

- Weights and Biases in NN, hidden layer parameters + output later parameters

Hyperparameters

- Batch size → 32
- Epochs → 200
- Learning Rate → 0.001
- Class Weights → Balanced

Weighted Cross-Entropy Loss

$$p(c|\mathbf{x}) = \frac{e^{z_c}}{\sum_{j=1}^{C} e^{z_j}}.$$

$$\mathcal{L}(\mathbf{z}, y) = -\sum_{c=1}^{C} w_c \, \delta_{c,y} \, \log\big(p(c|\mathbf{x})\big),$$

where $\delta_{c,y} = 1$ if $c = y$ and 0 otherwise.

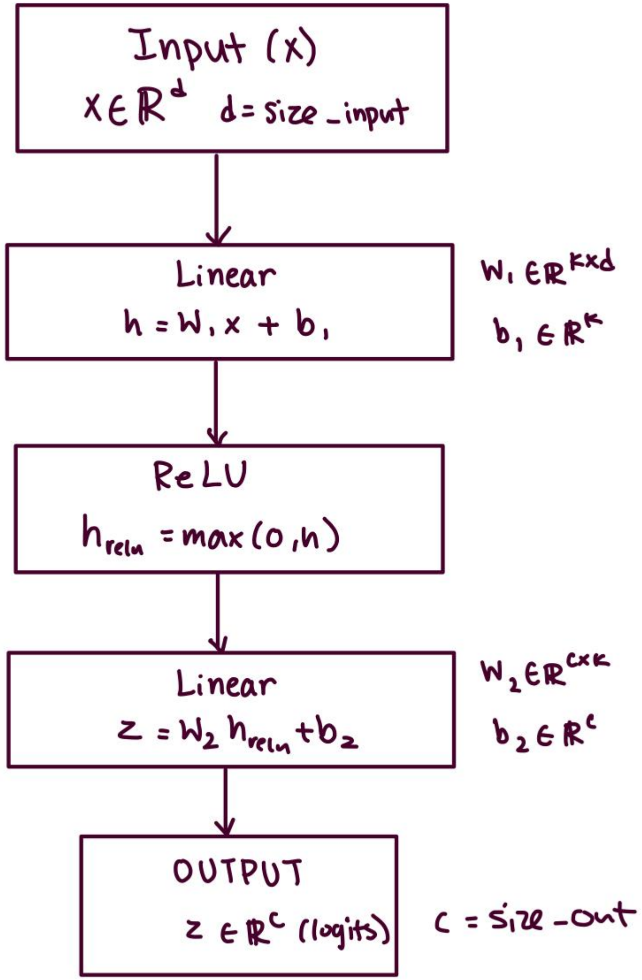
Adam Optimizer Equations

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla_\theta \mathcal{L}(\theta), \quad v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\nabla_\theta \mathcal{L}(\theta))^2.$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}.$$

$$\theta \leftarrow \theta - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

Feed Forward Neural Network



FFNN Results

Metric	Test Score
Accuracy	0.983
Precision	0.970
Recall	0.983
F1-Score	0.976
Matthews Correlation	0.715

Interpretation

- High performance (Accuracy → 98.3%)
- High F1-score: balance between precision and recall
- Higher Matthews Correlation Coefficient (71.5%)

Performance Differences

Comparing the models with only accuracy is not enough!

Model	Matthews Correlation Coefficient	Accuracy	Precision	Recall	F1-Score
Naive (Self-implemented)	0	0.965	1	0.965	0.982
Decision Tree (Self-implemented)	0.441	0.968	0.981	0.968	0.974
Random Forest (SKlearn)	0.755	0.985	0.993	0.985	0.989
Random Forest (Self-implemented)	0.191	0.966	0.997	0.966	0.981
K-NN (SKlearn)	0.414574	0.969	0.983541	0.969	0.975240
Neural Network (Built using Torch)	0.715	0.983	0.970	0.983	0.976

Conclusion

1. Class Imbalance
   - MCC metrics needed to failure mode identification efficiency
2. Model Complexity and Tuning
   - Traditional Models
     - E.g. decision trees and random forests
     - ❖ Effective without extensive tuning
   - Complex Models
     - E.g. decision trees and random forests
     - ❖ May need extensive hyperparameter tuning or more data

The cost of a missed failure is high

Identify rare failure states early and accurately

Neural Networks!

Future Work

1. Additional Balancing Techniques
   - Synthetic Minority Oversampling Technique
   - Class-weight adjustments
2. Neural Network Optimization
   - Use different network architectures (CNNs…)
   - Tuning of the hyperparameters (learning rates, layers…)
3. Fairness and Ethics
   - Need to be guided by principles of fairness and ethics.

Thank You!