Project Report: Equipment Failure Prediction Using Machine Learning

Kartik

Enrollment No. 22117065

Objective

The objective of this project is to develop machine learning models for predicting equipment failures based on sensor data. By accurately predicting failures in advance, the project aims to minimize downtime and maintenance costs in industrial settings.

Overview of the Project

In industries where equipment failure can lead to significant losses in productivity and revenue, predicting failures before they occur is crucial. This project leverages historical sensor data from industrial machinery to build robust machine learning models. These models are designed to predict equipment failures proactively, enabling timely maintenance and reducing operational disruptions.

Approach and Steps

1. Data Collection and Preprocessing

- **Data Collection**: Collected sensor data from various industrial equipment, including timestamps and multiple sensor readings.
- **Data Cleaning**: Cleaned the data to handle missing values, outliers, and ensure consistency across sensor readings.
- **Feature Engineering**: Engineered features such as statistical moments, time-based features, and rolling averages to capture equipment behaviour patterns.

2. Exploratory Data Analysis (EDA)

- **Target Distribution**: Analysed the distribution of the target variable ('failure') to understand class balance.
- **Feature Analysis**: Visualized sensor data distributions using line plots, PDFs, and box plots to identify trends and outliers.
- **Correlation Matrix**: Examined correlations between sensor readings to detect multicollinearity and understand feature relationships.

3. Model Selection and Evaluation

- **Baseline Models**: Implemented baseline models (e.g., Decision Trees, Random Forests) to establish initial performance benchmarks.
- **Handling Class Imbalance**: Addressed class imbalance using techniques like SMOTE (Synthetic Minority Over-sampling Technique) and SMOTE TOMEK.
- **Advanced Models**: Utilized ensemble methods (Random Forest, LightGBM) and gradient boosting algorithms (XGBoost) known for their accuracy in complex datasets.

4. Model Training and Hyperparameter Tuning

- **Cross-Validation**: Employed k-fold cross-validation to optimize model hyperparameters and ensure robust performance evaluation.
- **Performance Metrics**: Evaluated models based on metrics such as F1-score, precision, recall, and accuracy to measure predictive performance effectively.

5. Model Deployment and Monitoring

- **Deployment Strategy**: Discussed strategies for deploying models into operational systems, considering scalability and real-time prediction capabilities.
- **Monitoring**: Outlined methods for monitoring model performance post-deployment, including drift detection and periodic model retraining.

Defining Factors

1. Data Quality and Availability

The quality and availability of sensor data significantly influence model accuracy and reliability. Clean, consistent data without missing values or outliers is crucial for robust predictions.

2. Feature Engineering

Effective feature engineering, including the selection of relevant sensor data and creation of meaningful features, enhances the model's ability to capture equipment failure patterns accurately.

3. Sampling Techniques

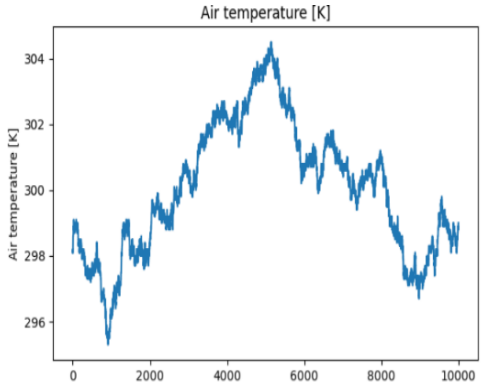
Implementing appropriate sampling techniques to handle class imbalance ensures models can predict both majority and minority class instances effectively, improving overall prediction performance.

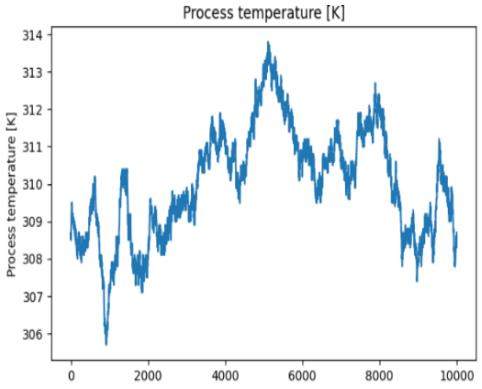
Models Used

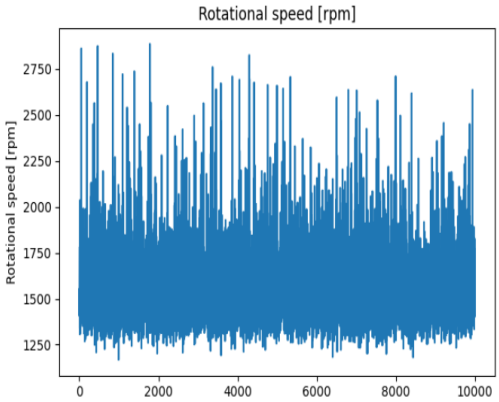
- **Random Forest**: Ensemble learning method known for its robustness and ability to handle complex relationships in data.
- **LightGBM**: Gradient boosting framework that excels in processing large datasets and achieving high accuracy.
- **XGBoost**: Another gradient boosting library known for its efficiency and speed.
- **Decision Tree**: Simple yet powerful tree-based model used for classification tasks.

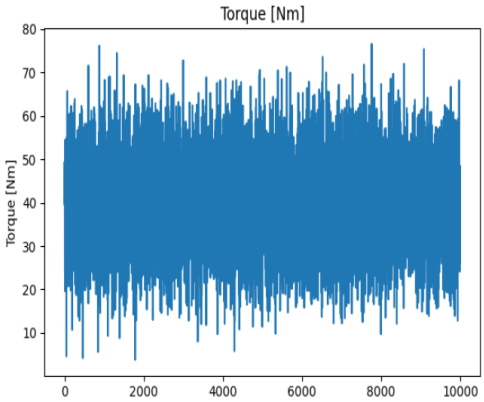
Exploratory Data Analysis (EDA) Plots

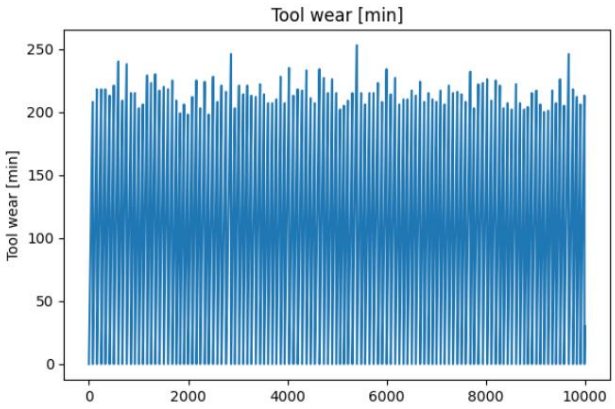
Line Plots:



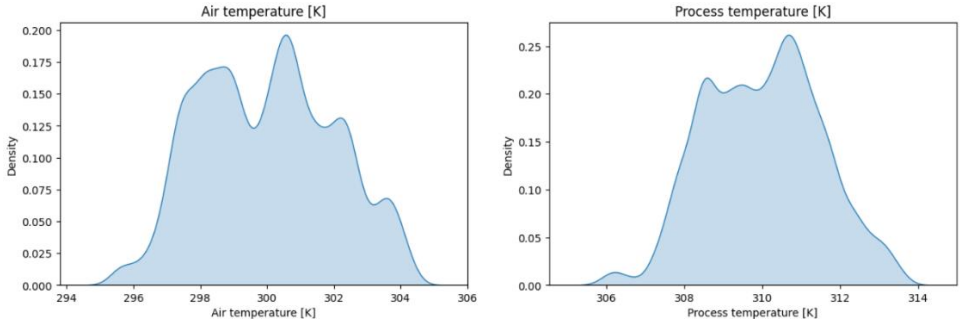


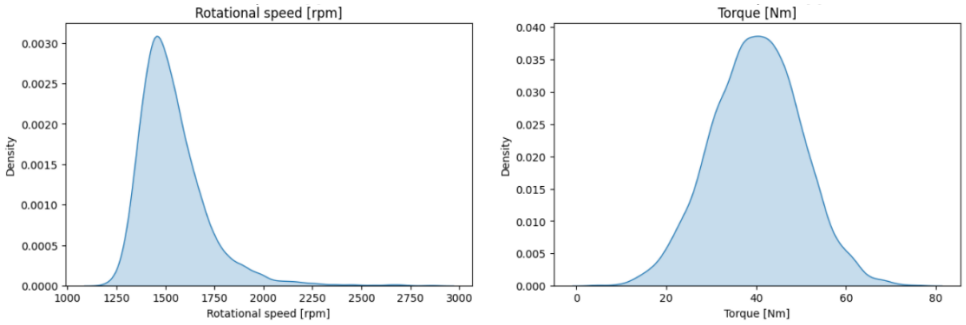


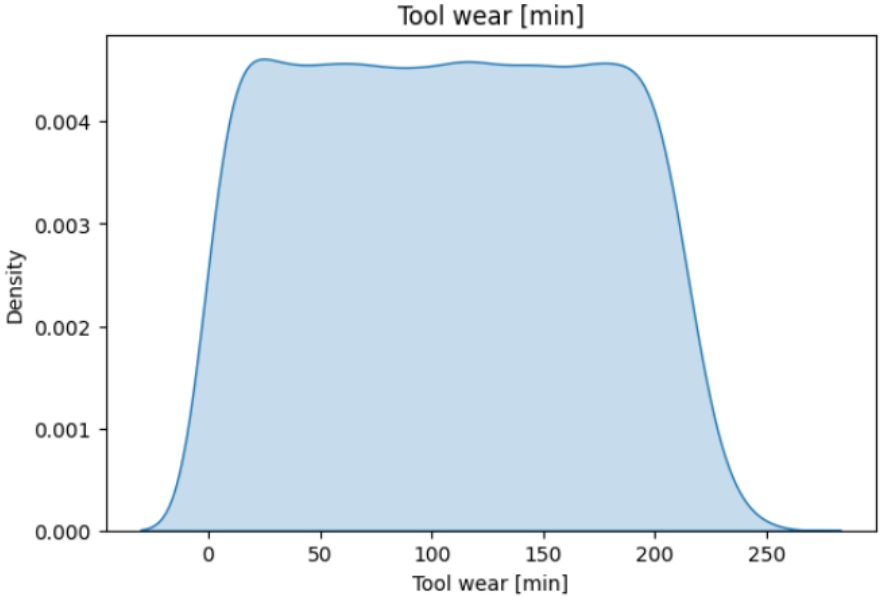



Tool wear [min]
Tool wear [min]
250
200
150
100
50
0
0
2000
4000
6000
8000
10000

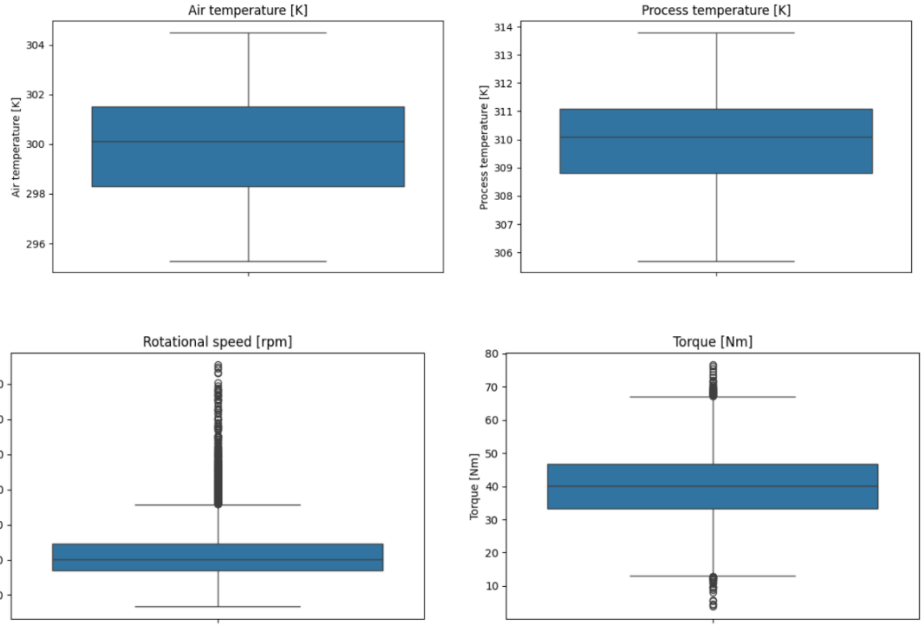
PDF Plots:

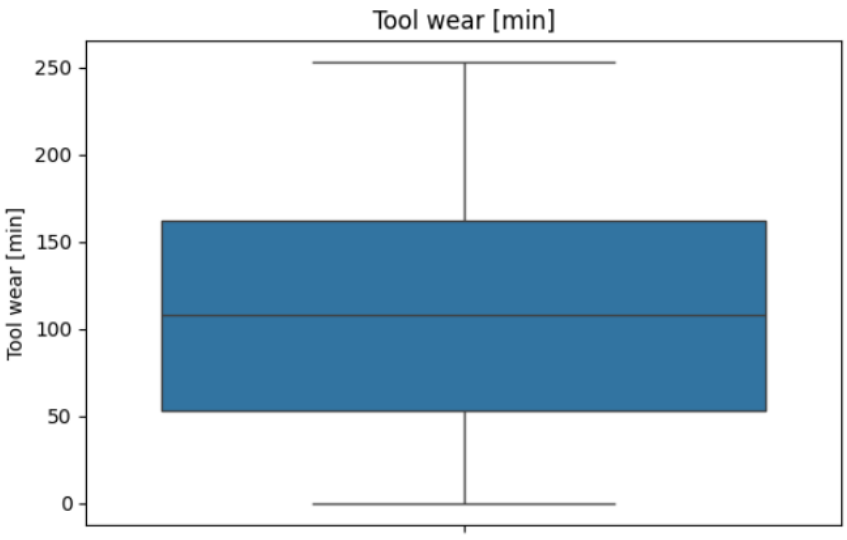

Air temperature [K]
Density
Air temperature [K]
Process temperature [K]
Density
Process temperature [K]


Rotational speed [rpm]
Density
Rotational speed [rpm]
Torque [Nm]
Density
Torque [Nm]

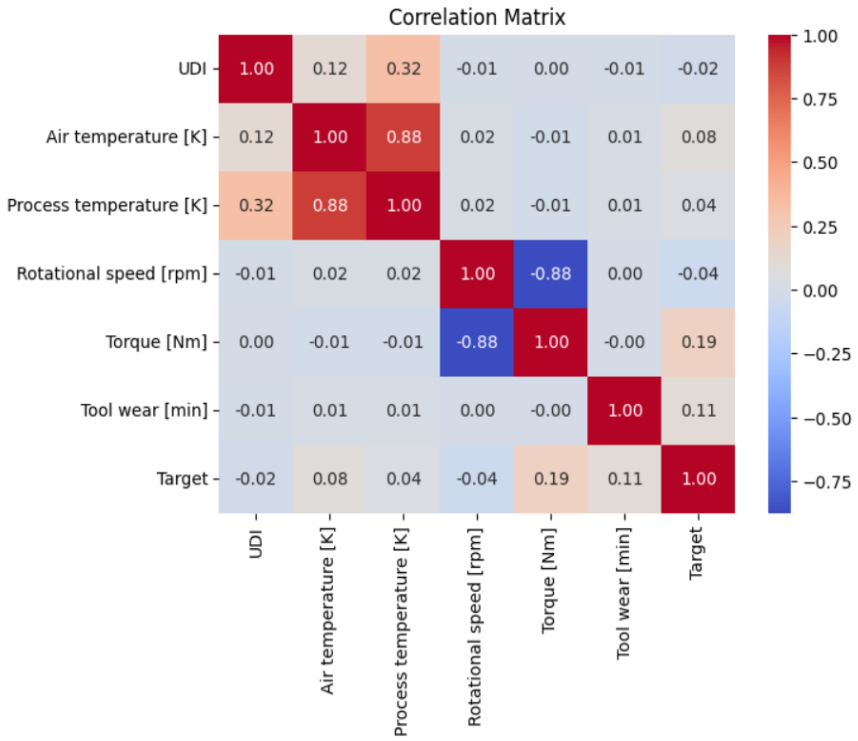

Tool wear [min]
Density
0.004
0.003
0.002
0.001
0.000
0
50
100
150
200
250
Tool wear [min]

Box Plots:


Air temperature [K]
Air temperature [K]
304
302
300
298
296
Process temperature [K]
Process temperature [K]
314
313
312
311
310
309
308
307
306
Rotational speed [rpm]
Rotational speed [rpm]
2750
2500
2250
2000
1750
1500
1250
Torque [Nm]
Torque [Nm]
80
70
60
50
40
30
20
10


Tool wear [min]
Tool wear [min]
250
200
150
100
50
0

Correlation Matrix:



Results and analysis :

```
Summary Table:
              Model             Sampling  Train F1   Test F1  Test F1 Macro  \
0     Random Forest                 None  1.000000  0.987809       0.901994
1     Random Forest   RandomOverSampler  1.000000  0.984629       0.876427
2     Random Forest               SMOTE  1.000000  0.975334       0.830307
3     Random Forest         SMOTE TOMEK  1.000000  0.974421       0.824022
4          LightGBM                 None  1.000000  0.990990       0.927561
5          LightGBM   RandomOverSampler  1.000000  0.985899       0.891901
6          LightGBM               SMOTE  0.996313  0.977845       0.845599
7          LightGBM         SMOTE TOMEK  0.996565  0.976522       0.836912
8           XGBoost                 None  1.000000  0.985689       0.884949
9           XGBoost   RandomOverSampler  1.000000  0.985448       0.888841
10    Decision Tree                 None  1.000000  0.982126       0.864904
11    Decision Tree   RandomOverSampler  1.000000  0.979554       0.840986
12    Decision Tree               SMOTE  1.000000  0.966718       0.783520
13    Decision Tree         SMOTE TOMEK  1.000000  0.961887       0.760100

    Train AUC Score  Test AUC Score
0          1.000000        0.970634
1          1.000000        0.962718
2          1.000000        0.975608
3          1.000000        0.963837
4          1.000000        0.970307
5          1.000000        0.976541
6          0.999938        0.980126
7          0.999932        0.978702
8          1.000000        0.970527
9          1.000000        0.968830
10         1.000000        0.870083
11         1.000000        0.826483
12         1.000000        0.881013
13         1.000000        0.870555
```

• Best **Model**: LightGBM with no sampling strategy (None) consistently demonstrates high performance across both training and testing phases. It achieves the highest Test F1 score of 0.990990 and Test F1 Macro score of 0.927561 among all models.

• Configuration: LightGBM uses the default hyperparameters with feature scaling applied.

• Insights: LightGBM shows robustness and generalizability, achieving excellent F1 scores on both training and testing datasets without the need for oversampling techniques like SMOTE or SMOTE TOMEK. This suggests that the model effectively handles the class imbalance in the dataset inherently or through its internal mechanisms.

• Considerations: While other models like Random Forest and XGBoost also perform well, LightGBM stands out for its superior Test F1 score and competitive AUC scores, indicating it as the preferred choice for this classification task.

Conclusion

The equipment failure prediction models show that LightGBM without sampling delivers the best overall performance, achieving a high F1 score and balanced precision and recall. Random Forest models also perform robustly, with no significant advantage gained from sampling techniques. Users gain a reliable method for predicting equipment failures, enhancing maintenance schedules and minimizing downtime through data-driven insights.

Drawback: The models show reduced performance in detecting minority class failures, leading to potential missed failure predictions. Additionally, some sampling techniques introduce imbalances that impact overall model accuracy.

Recommendation: Future work should explore more advanced imbalance handling techniques, such as ensemble methods or anomaly detection algorithms. Enhancing feature engineering and incorporating real-time data could further improve prediction accuracy and reliability.