FSM Online Internship Completion Report
on

Equipment Failure Prediction for Predictive Maintenance

In

Machine Learning

Submitted by

Nancy
3rd Year, IGDTUW

Under Mentorship of
Mr. Dhairya

IITD-AIA Foundation for Smart Manufacturing

[1-June-2021 to 31-July-2021]

Equipment Failure Prediction for Predictive maintenance

Abstract

The emerging infrastructure presented by the Internet of Things and data science have been a revolutionary factor in the manufacturing industry. In modern manufacturing system, RUL prediction has been increasingly important in machine health monitoring. Predictive maintenance is a technique that aims to prevent equipment failures by using data analysis and machine learning methods. The challenge of the project also, was to predict the Remaining Useful Life of the engine by using the given sensor's data and operational conditions. But I tried to simplify that by converting it to a Classification Problem, where the class labels will be of 3 types, i.e., Good Condition represented by 0, Moderate Condition represented by 1 and Warning Condition represented by 2. So, in this paper, a methodology to predict the time of equipment failure is proposed so that the machine can be repaired to reduce the risk of failure. The NASA turbofan engine dataset is used to implement the methodology. The machine learning and Deep Learning models such as Logistic Regression, Random Forest, Gradient Boosting, KNN, Decision Tree, Extreme Gradient Boosting, Light GBM, Gaussian Naive Bayes, ANN, LSTM are implemented. But the real-world datasets are mostly imbalanced and the imbalanced class distributions pose several issues in the performance of classifiers. Consequently, the classifiers show low Accuracy, Precision value, Recall value with a high degree of misclassification, etc. So, it is required to handle the imbalance dataset. SMOTE, SMTOEENN, SMOTE-Tomek balancing techniques are used to balance the imbalance dataset. The empirical analysis of the traditional ML models, along with comparative analysis of all the implemented hybrid models is done. The evaluation is done based on accuracy metrics, precision value, recall value and F-1 measure score.

Keywords: Predictive maintenance, Random Forest, Gradient Boosting, Imbalanced dataset, ANN, LSTM

Table of Content

1. Introduction

In modern manufacturing system, RUL prediction has been increasingly important in machine health monitoring. Compared with traditional physics-based models, data-driven models gain more attention due to the significant development of sensors, sensor networks and computing systems. Predictive maintenance is a proactive maintenance strategy that uses condition monitoring tools and predictive analytics to determine when maintenance is necessary and what maintenance needs to be performed. One of the challenges of predictive maintenance is to estimate the time-to-failure of equipment, which is the remaining useful life (RUL) of the equipment before it fails. It can reduce the likelihood of failures, avoid costly downtime, and lower maintenance costs. Predictive maintenance can be implemented by using sensor technology and computerized maintenance management system (CMMS) software. With the advancement of technology, industries leverage AI, machine learning, and deep learning to analyse massive real time data from sensors and devices, and generate insights and recommendations for maintenance actions. I have worked on the NASA turbofan engine dataset. Just like most of real-world dataset, the dataset is also imbalanced or we can also say that the class label, created are not balanced. The machine learning classification algorithms expect that the minority and majority classes are balanced, thus the imbalance class must be handled before fetching the dataset and training any machine learning model.

1.1. What is Equipment Failure Prediction for Predictive Maintenance?

Equipment failure prediction is a technique that aims to prevent or reduce the occurrence of equipment failures by using data analysis and machine learning methods. Equipment failures can cause significant losses in terms of productivity, quality, safety, and customer satisfaction. Therefore, it is important to detect and diagnose the signs of potential failures before they happen and take appropriate actions to avoid or mitigate them. Equipment failure prediction can be applied to various domains such as telecommunication, aviation, manufacturing, transportation, and healthcare. It can also be integrated with predictive maintenance, which is a strategy that optimizes the maintenance schedule and resources based on the condition and performance of the equipment. Equipment failure prediction for predictive maintenance is necessary because it can help:

- Reduced downtime: By predicting equipment failures in advance, organizations can schedule maintenance activities during planned downtime, minimizing unexpected disruptions and costly downtime periods.
- Increased lifespan of the equipment: Predictive maintenance can help extend the useful life of the equipment by preventing premature wear and tear, and reducing the risk of catastrophic failures.

- Lowered costs: By predicting equipment failures and optimizing maintenance schedules, organizations can save on labour, materials, and energy consumption associated with maintenance and repairs.
- Mitigate risks by avoiding safety hazards, environmental impacts, and customer dissatisfaction caused by equipment failures.
- Improved efficiency: Companies utilizing predictive maintenance solutions can optimize their operations and resources, improving efficiency and productivity.

1.2. Challenges in predicting equipment failure for predictive maintenance

- Data availability and quality: It is essential to collect data for training machine learning algorithms. This information must be readily accessible in sufficient numbers for the algorithms to learn properly. The data also must be accurate, consistent, and relevant to the equipment failure modes.
- Complexity of machine learning algorithms: Algorithms for machine learning can be complicated and challenging to implement, necessitating specific knowledge and skills. The algorithms also must be robust, scalable, and adaptable to different types of equipment and failure scenarios.
- Cost and time of implementation: Predictive maintenance requires investment in sensors, data storage, computing power, and software development. It also takes time to collect enough data, train the algorithms, validate the results, and deploy the solutions

1.3. Traditional method equipment failure prediction

- Vibration analysis: This technique monitors the vibration of equipment components to detect problems such as misalignment, imbalance, looseness, or wear.
- Oil analysis: This technique analyses the oil samples from lubricated equipment to detect contaminants, degradation, or wear particles that indicate the condition of the equipment.
- Thermography: This technique uses infrared cameras to detect high temperatures in equipment that indicate overheating, friction, or electrical faults.
- Acoustic analysis: This technique monitors the sound frequencies of equipment to detect problems in their technical performance, such as leaks, cavitation, or bearing defects1.
- Motor circuit analysis: This technique measures the electrical parameters of motor circuits to detect problems such as insulation breakdown, winding faults, or rotor bar defects.

1.4. Challenges in predicting Equipment failure using traditional techniques

- Limited scope and accuracy: Traditional techniques such as vibration analysis, oil analysis, or thermography can only detect certain types of failures and may not be able to capture the complex interactions and dependencies among different components or systems.
- High cost and complexity: Traditional techniques require specialized equipment, skilled personnel, and frequent inspections to collect and analyse the data. This can be expensive, time-consuming, and prone to human errors.
- Low reliability and scalability: Traditional techniques may not be able to handle the large volume, variety, and velocity of data generated by modern equipment. They may also suffer from data quality issues, such as noise, outliers, or missing values, that can affect the accuracy and reliability of the predictions.

1.5. Problem with Imbalanced Dataset

A classification dataset with uneven class proportions is called imbalanced dataset. If the dataset imbalance is not addressed properly, then it can cause following problems for machine learning algorithms, such as:

- The classification algorithms will show higher accuracy for the majority classes and poor accuracy for the minority classes.
- Bias towards the majority class and ignoring the minority class
- Poor classification of the minority class and high misclassification rate
- Misleading accuracy score and sub-optimal model performance

2. Problem Definition

The NASA turbofan jet engine data set is a synthetic data set that simulates the run-to-failure degradation of a fleet of turbofan engines. The problem definition is to predict the remaining useful life (RUL) of the engines based on their engine settings and sensor readings. RUL is equivalent of number of flights remained for the engine after the last datapoint in the test dataset. The dataset consists of multiple multivariate time series. Each data set is further divided into training and test subsets. Each time series is from a different engine i.e., the data can be from a fleet of engines of the same type. Each engine starts with different degrees of initial wear and manufacturing variation which is unknown to the user. This wear and variation are considered normal, i.e., it is not considered a fault condition. There are three operational settings that have a substantial effect on engine performance. These settings are also included in the data. The data is contaminated with sensor noise. It has 4 datasets, FD001, FD002, FD003, FD004 which contains time series of 21 sensors and 3 settings of 100 turbofan engines. Each engine works normally at the beginning of each time series and fails at the end of the time series. Each row is a snapshot of the data taken during a single operation cycle. The columns/features of the dataset are given below:

Engine, cycle, setting_1, setting_2, setting_3] // the engine, cycle and 3 settings
(Fan inlet temperature) (◦R) // 21 sensors
(LPC outlet temperature) (◦R)
(HPC outlet temperature) (◦R)
(LPT outlet temperature) (◦R)
(Fan inlet Pressure) (psia)
(Bypass-duct pressure) (psia)
(HPC outlet pressure) (psia)
(Physical fan speed) (rpm)
(Physical core speed) (rpm)
(Engine pressure ratio) (P50/P2)
(HPC outlet Static pressure) (psia)
(Ratio of fuel flow to Ps30) (pps/psia)
(Corrected fan speed) (rpm)
(Corrected core speed) (rpm)
(Bypass Ratio)
(Burner fuel-air ratio)
(Bleed Enthalpy)
(Required fan speed)
(Required fan conversion speed)

(High-pressure turbines Cool air flow)
(Low-pressure turbines Cool air flow)

A low-pressure compressor (LPC) and high-pressure compressor (HPC) supply compressed high temperature, high pressure gases to the combustor. Low pressure turbine (LPT) can decelerate and pressurize air to improve the chemical energy conversion efficiency of aviation kerosene. High pressure turbines (HPT) generate mechanical energy by using high temperature and high-pressure gas strike turbine blades. Low-pressure rotor (N1), high-pressure rotor (N2), and nozzle guarantee the combustion efficiency of the engine. The ratio of the air that goes around the engine to the air that goes through the core is called the bypass ratio. Each cycle of a turbo engine generates a certain amount of wear and tear on its components, and the performance of the engine may degrade over time due to various factors, such as aging, contamination, or damage. Therefore, monitoring the engine cycles and predicting its remaining useful life (RUL) is crucial for ensuring its reliability and safety.

The challenge of this problem was to predict the Remaining Useful Life of the engine by using the given sensor's data and operational conditions. But I tried to simplify that by converting it to a Classification Problem, where the class labels will be of 3 types, i.e., Good Condition, Moderate Condition and Warning Condition.

Labels corresponding to each condition:

- Good Condition – 0
- Moderate Condition – 1
- Warning Condition – 2

Warning condition means that the maintenance is required and must be done to reduce the risk of failure of engine. If we get 0 or 1 then it means the engine is working fine and no need of maintenance. To defining the class labels, I define the engine's condition with Life_Ratio (LR), which is the ratio between Current Cycle and the End cycle (end_of_life). If LR=0, that means the component has just started its degradation and LR=1 means, it is completely degraded.

- If LR <= 0.6 - Good Condition (0)
- if LR <= 0.8 - Moderate Condition (1)
- if 0.8 < LR - Warning Condition (2)

3. Existing Solution

Machine learning techniques, especially, the Deep learning (DL) techniques are regarded as a powerful solution due to its ability to provide a more agility to process data associated with highly nonlinear and complex feature abstraction through a cascade of multiple layers. DL provides the decision-makers new visibility into their operations, as well as real-time performance measures and costs. Recently, the latest DL-based, four main DL variants: AE, DBN, CNN and RNN are introduced. It has been observed that DL-based techniques were mainly used for fault diagnostics, and very limited studied applied DL-based techniques in RUL prediction until recent years. Also, an increasing interest and suggests a promising future of DL in RUL prediction. Besides, DL related RUL prediction approaches are purely data driven approaches. Deep Neural Networks (DNN) have delivered their most prominent achievements in the fields of Computer Vision (CV) and Natural Language Processing (NLP), recent research works have shown their effective use also for prognostics. However, one of the limitations of standard DNN models is that they do not provide an explicit quantification of the uncertainty associated with the predicted RUL.

Biggio[1] in their work, proposes an uncertainty-aware remaining useful life (RUL) predictor that leverages a variational recurrent neural network (VRNN) to model the temporal dynamics of the degradation process and to estimate the aleatoric and epistemic uncertainties associated with the RUL prediction. They considered Deep Gaussian Processes (DGPs) as possible solutions for predicting RUL. They implemented various methods like Stochastic Variational Gaussian Processes (SVGPs), Deep Gaussian Processes (DGPs), Deep Sigma Point Processes (DSPPs) and Monte Carlo Dropout (MCD). Overall, their results demonstrate that the DSPP and MCD models are best performing.

Maschler [2] in their work, presented a continual learning-based algorithm for fault prediction of turbofan engines, allowing for distributed, cooperative learning by elastic weight consolidation (EWC). EWC is a method that enhances a conventional neural network by adding a regularization term to the loss function. They use deep learning architectures for the prediction of the remaining useful life. They chose an autoencoder and an LSTM for their proposed deep learning architecture. Their evaluation of this conventional deep learning architecture revealed a performance close to, but not as good as the best-performing published approaches.

Ellefsen[3] in their work proposed a novel concurrent semi-supervised model to estimate the remaining useful life (RUL) of the aero-engine. The model combines unsupervised learning with a concurrent structure that can improve the RUL prediction accuracy on multivariate time series data. Their paper claims that the proposed model outperforms several state-of-the-art methods on two public datasets.

Wu [4] in their work, concluded that random forests (RFs) outperform artificial neural networks (ANNs) and support vector regression (SVR) in terms of accuracy and training time for tool wear prediction in milling operations. They also, introduces a RFs-based prognostic method for tool wear prediction and evaluates it on an experimental dataset collected from 315 milling tests.

4. Proposed Development

In any model implementation firstly, the dataset is fetched and data preprocessing is done which is then followed by data visualization. After that the dataset is split into training and testing data. Then the model is trained on the training data and the trained model is then tested on test data. Finally, the evaluation of the implemented model is done. As mentioned in the previous section, the challenge of this problem was to predict the Remaining Useful Life of the engine by using the given sensor's data and operational conditions. But I tried to simplify that by converting it to a Classification Problem, where the class labels will be of 3 types, i.e., Good Condition, Moderate Condition and Warning Condition. So, after creating class labels, we got to know that dataset is imbalanced so using SMOTE and SMOTE-Tomek balancing technique the datasets are balanced and various ML and DL algorithms are applied.

1. Dataset Selection: Selecting or collecting the data of Liver patients for selecting meaningful records to obtain and analyze the productive knowledge by performing various data mining technique.
2. Data Preprocessing: Data preprocessing is an important step in the data mining process which involves cleaning and transforming raw data to make it suitable for analysis. Some

common steps in data preprocessing include, Data Cleaning, Data Transformation, Data Reduction, Handling Missing Data and Noisy Data.

3. Feature Selection: It involves choosing the independent variables which have a significant relationship with the dependent variable, and to remove irrelevant features. In simple terms, the limiting the number of input variables so that the machine learning models are trained faster. It reduces the computational complexity, improve the performance of models, and makes it easier to interpret.
4. Data Balancing: Handling the imbalance dataset using different balancing techniques. The two balancing techniques are used namely, SMOTE and SMOTE-Tomek.
   - SMOTE (Synthetic Minority Oversampling Technique):

The main use of this technique is to balance the data by synthesizing additional samples for the minority class. It randomly selects ‘A’ minority class instance and searches for its k nearest minority class neighbors. The synthetic instance is then created by choosing one of the k nearest neighbor ‘B’ at random and connecting ‘A’ and ‘B’ to form a line segment in the feature space. The convex combination of the two chosen instances A and B are generated synthetic instances. This method is effective because the synthetic data generated, are relatively close with the feature space on the minority class, resulting in adding new “information” on the data, unlike the original oversampling method.

Pseudo Code:

Input: minority class samples M; amount of synthetic sample N; number of nearest neighbors k

```
for i in range(N):
   x = random.sample(M) // generating random sample
   neighbors = k-nearest-neighbors(x)
   y = random.sample(neighbors)
   sample = x + (y - x) * random.uniform(0, 1)
   T.add(sample)
```

Output: T synthetic minority class samples

- SMOTETomek: Smote (Oversampler) + TomekLinks (Undersampler).

This method combines the SMOTE ability to generate synthetic data for minority class and Tomek Links ability to remove the data that are identified as Tomek links from the majority class i.e., the samples of data from the majority class that is closest with the minority class data. Tomek Links can be used as a Under Sampling techniques to remove the majority class samples to balance the data set. This approach can be used as data cleaning method, which can remove samples from both majority and minority classes if the below Tomek Link condition is satisfied.

TOMEK: Tomek links is defined as follows: given two samples which belongs to different classes, E_i and E_j. The distance between E_i and E_j is $d(E_i, E_j)$. This pair will be called a Tomek link if there is not an sample E_l, such as that $d(E_i, E_l)$ or $d(E_j, E_l)$ is less than $d(E_i, E_j)$.

Pseudo Code:

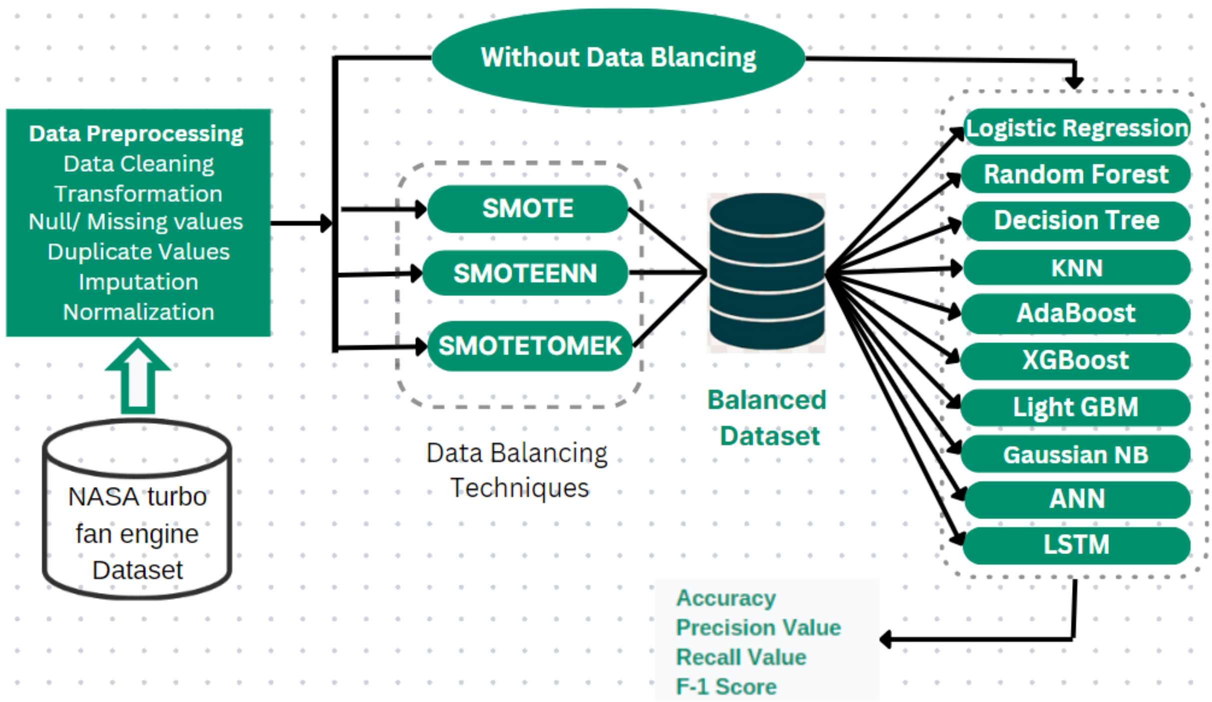
---

1. (Starting of SMOTE) From minority class, choose random data.
2. The distance between the choosen random data and its k nearest neighbors is calculated.
3. Select a random number between 0 to 1 and multiply it with the above calculated distance.
4. We get a synthetic sample. Add this synthetic sample to the minority class.

5. Repeat the step 2 to 4 to get the desired proportion of minority class. (End of SMOTE)
6. (Starting of Tomek Links) From the majority class, choose random data.
7. Find the nearest neighbor of the choosen random data. If it is the data from the minority class i.e. Tomek Link is created, then remove the Tomek Link. (End)

5. Model Building: Train the model based on Maching Learning Algorithms like: Logistic Regression, Random Forest, Gradient Boosting, KNN, Decision Tree, Extreme Gradient Boosting, Light GBM, Gaussian Naive Bayes, ANN, LSTM.

6. Model Building and Evaluation: Train the model based on ML algorithms mentioned above. And then evaluating the performance of the models using accuracy, precision, recall, F1-score using Confusion metrics. Because accuracy alone is not a good metrics for imbalanced datasets.



Workflow of the implementation used

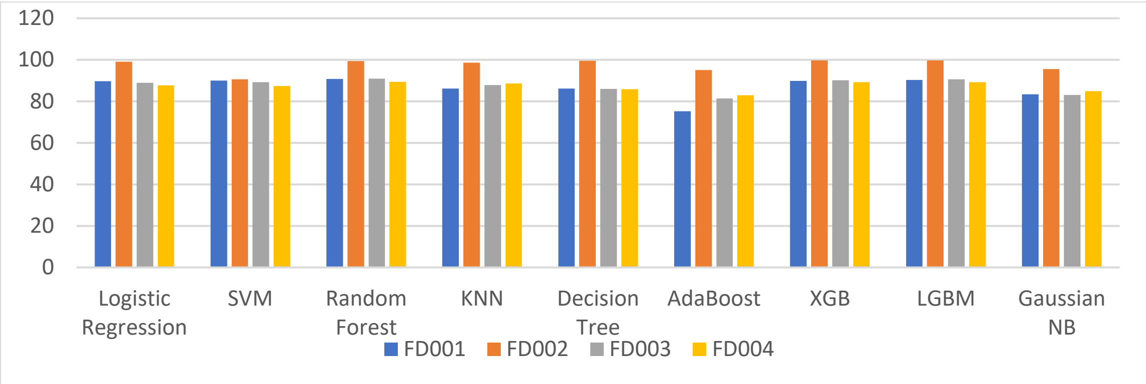
5. Functional Implementation

Using Traditional ML models before balancing the dataset: The below table shows the result we got after implementing traditional ML algorithms like Logistic Regression, SVM, Random Forest, KNN, decision Tree, AdaBoost, XGB, Light GBM, Gaussian Naïve Bayes on training and testing data.

Dataset→	**FD001**		**FD002**		**FD003**		**FD004**	
Model	**Train Acc**	**Test Acc**	**Train Acc**	**Test Acc**	**Train Acc**	**Test Acc**	**Train Acc**	**Test Accu**
Logistic Regression	89.24%	89.70%	99.14%	99.1 %	88.72%	88.95%	87.1%	87.72%
SVM	89.95%	90.06%	99.56%	90.6%	89.12%	89.27%	87.52%	87.46%
Random Forest	100%	90.74%	100%	99.4%	99.99%	90.89%	100%	89.33%

KNN	90.23%	86.14%	99.25%	98.56%	91.64%	87.86%	91.75%	88.67%
Decision Tree	100%	86.14%	100%	99.59%	100%	86%	100%	85.78%
AdaBoost	75.26%	75.18%	94.93%	95.10%	80.99%	81.37%	83.28%	82.88%
XGB	98.99%	89.82%	100%	99.70%	98.1%	90.19%	93.28%	89.22%
Light GBM	96.83%	90.28%	100%	99.70%	95.9%	90.57%	91.55%	89.3%
Gaussian NB	82.95%	83.42%	95.06%	95.56%	83.22%	83.11%	84.67%	84.86%

Table 1: Accuracy of models on training and testing data



Graph of the accuracy of implemented ML models on all the 4 datasets

Random Forest, XGB and Light GBM models shows 100% training accuracy. The reason for this is overfitting because ML models cannot have 100 training accuracy. Overfitting occurs when a ML model fits too closely to a particular dataset. In simple terms when a model is trained too much and the models starts memorizing the data instead of learning from it. Now, let us balance the data set to improve the result and accuracy of models.

Handling Imbalanced dataset:

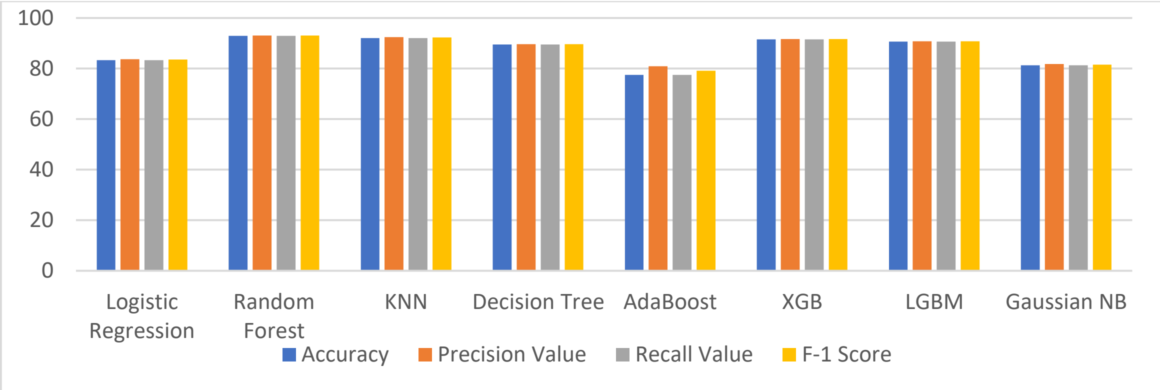
- Using Balancing Techniques SMOTE and SMOTE-Tomek
- Now splitting the balanced data into training and test data in 80%, 20 % ratio.
- Training the models on balanced data and test the implemented model on test data.
- The results we got for implemented classification techniques after balancing is shown in the below table:

For FD001 dataset:

- Using SMOTE:
  Original data: [0 1 2] [12338 4130 4163]
  After SMOTE: [0 1 2] [12338 12338 12338]

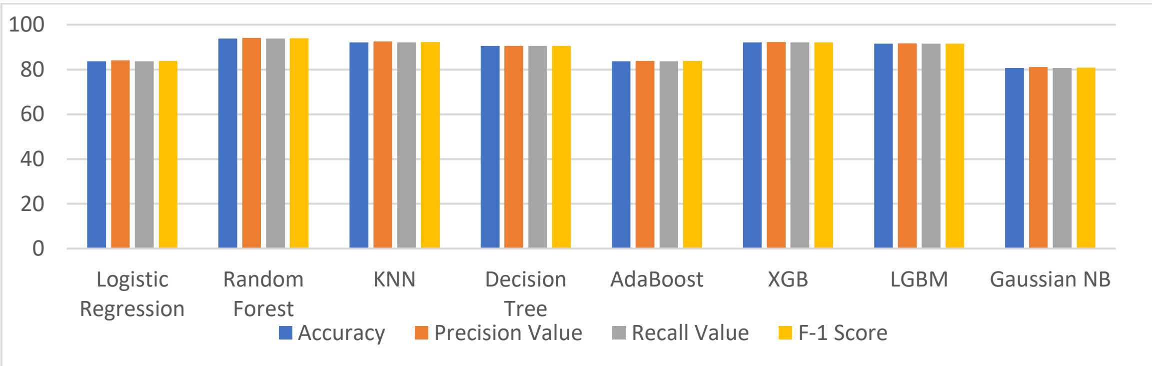
Models	**Accuracy**	**Precision Value**	**Recall Value**	**F-1 Score**
Logistic Regression	83.37%	83.71%	83.37%	83.54%
Random Forest	92.94%	93.08%	92.94%	93.01%
KNN	92.09%	92.46%	92.09%	92.28%
Decision Tree	89.54%	89.64%	89.54%	89.59%
AdaBoost	77.49%	80.90%	77.49%	79.16%
XGB	91.59%	91.69%	91.59%	91.64%
Light GBM	90.67%	90.77%	90.67%	90.72%

Gaussian Naïve Bayes	81.34%	81.81%	81.34%	81.57%



- Using SMOTE-Tomek
  Original data: [0 1 2] [12338 4130 4163]
  After SMOTE-Tomek: [0 1 2] [12202 12188 12320]

Models	**Accuracy**	**Precision Value**	**Recall Value**	**F-1 Score**
Logistic Regression	83.62%	84.06%	83.62%	83.84%
Random Forest	93.89%	94.06%	93.89%	93.97%
KNN	92.18%	92.48%	92.18%	92.33%
Decision Tree	90.52%	90.60%	90.52%	90.52%
AdaBoost	83.72%	83.88%	83.72%	83.80%
XGB	92.07%	92.23%	92.07%	92.15%
Light GBM	91.48%	91.65%	91.48%	91.57%
Gaussian Naïve Bayes	80.64%	81.10%	80.64%	80.87%



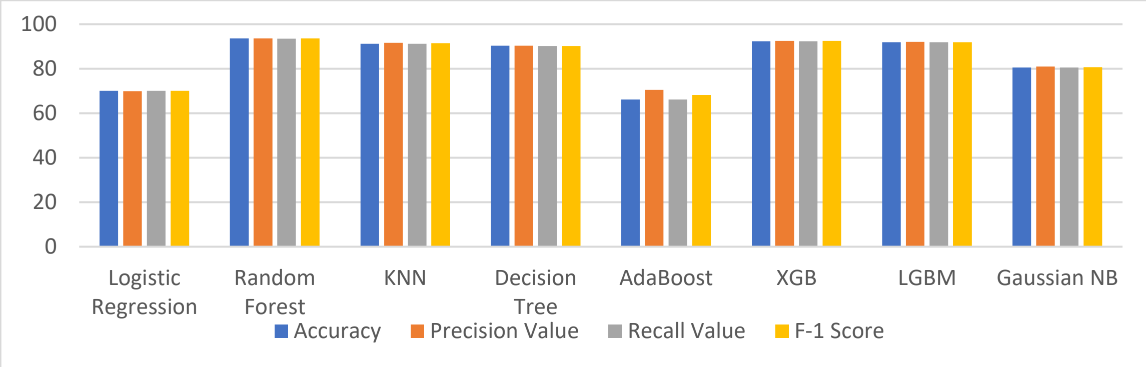
For FD003 dataset:

- Using SMOTE:

Original data: [0 1 2] [14793 4942 4985]
After SMOTE: [0 1 2] [14793 14793 14793]

Models	**Accuracy**	**Precision Value**	**Recall Value**	**F-1 Score**
Logistic Regression	70.12%	69.96%	70.12%	70.04%
Random Forest	93.65%	93.69%	93.56%	93.62%
KNN	91.2%	91.64%	91.20%	91.42%
Decision Tree	90.32%	90.26%	90.24%	90.25%
AdaBoost	66.13%	70.45%	66.15%	68.22%
XGB	92.37%	92.50%	92.37%	92.43%
Light GBM	91.86%	92%	91.86%	91.93%

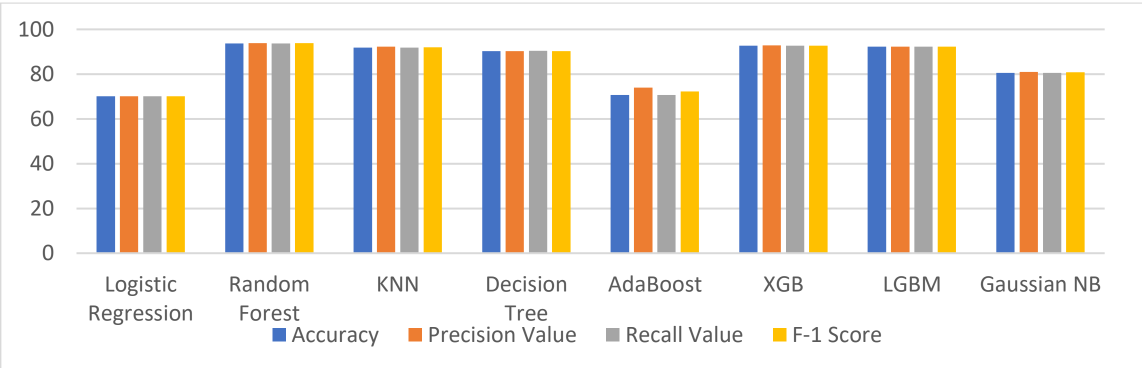
Gaussian Naïve Bayes	80.57%	81.02%	80.48%	80.75%



- Using SMOTE-Tomek:

Original data: [0 1 2] [14793 4942 4985]
After SMOTE-tomek: [0 1 2] [14632 14618 14771]

Models	**Accuracy**	**Precision Value**	**Recall Value**	**F-1 Score**
Logistic Regression	70.11%	70.13%	70.19%	70.12%
Random Forest	93.75%	93.83%	93.75%	93.79%
KNN	91.87%	92.23%	91.87%	92.05%
Decision Tree	90.32%	90.32%	90.37%	90.32%
AdaBoost	70.67%	73.95%	70.67%	72.27%
XGB	92.73%	92.80%	92.73%	92.77%
Light GBM	92.23%	92.33%	92.23%	92.28%
Gaussian Naïve Bayes	80.57%	80.99%	80.57%	80.78%



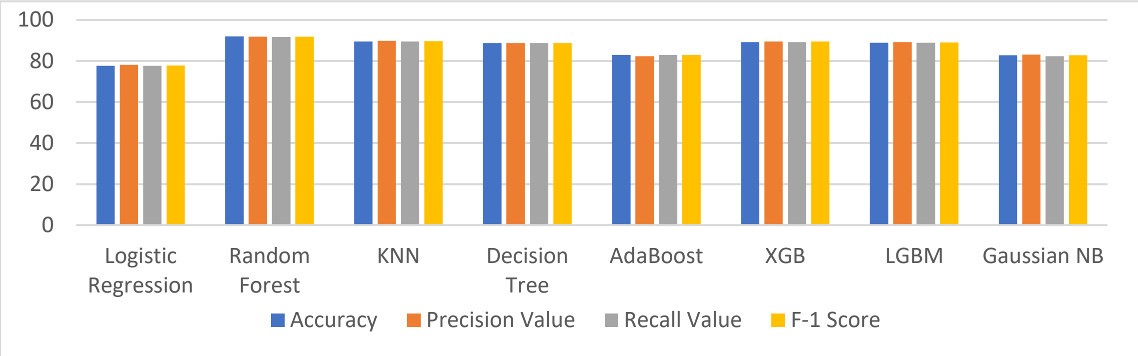
For FD004 dataset:

- Using SMOTE:

Original data: [0 1 2] [36655 12248 12346]
After SMOTE: [0 1 2] [36655 36655 36655]

Models	**Accuracy**	**Precision Value**	**Recall Value**	**F-1 Score**
Logistic Regression	77.65%	78.09%	77.65%	77.87%
Random Forest	91.96%	91.87%	91.69%	91.78%
KNN	89.43%	89.73%	89.44%	89.58%
Decision Tree	88.71%	88.78%	88.74%	88.74%
AdaBoost	83%	82.27%	83%	82.88%
XGB	89.13%	89.54%	89.13%	89.43%

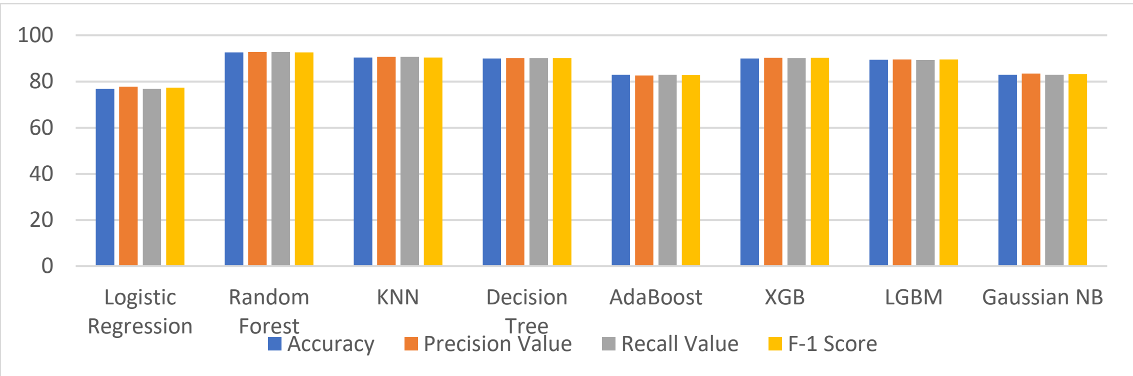
Light GBM	88.91%	89.15%	88.91%	89.03%
Gaussian Naïve Bayes	82.83%	83.10%	82.38%	82.74%



- Using SMOTE-Tomek

Original data: [0 1 2] [36655 12248 12346]
After SMOTE-tomek: [0 1 2] [35500 35246 36363]

Models	Accuracy	Precision Value	Recall Value	F-1 Score
Logistic Regression	76.78%	77.7%	76.78%	77.24%
Random Forest	92.53%	92.67%	92.67%	92.60%
KNN	90.39%	90.69%	90.66%	90.39%
Decision Tree	90%	90.05%	90.01%	90.03%
AdaBoost	82.87%	82.60%	82.87%	82.74%
XGB	90%	90.24%	90.08%	90.16%
Light GBM	89.38%	89.56%	89.3%	89.47%
Gaussian Naïve Bayes	82.83%	83.444%	82.87%	83.15%



SMOTE and SMOTE-Tomek shows similar performance. Both underperformed in some cases. SMOTE underperformed because it lacks flexibility and did over generalization. SMOTE-Tomek uses SMOTE to oversample the minority class and Tomek is used to remove majority class samples that are close to minority class samples.

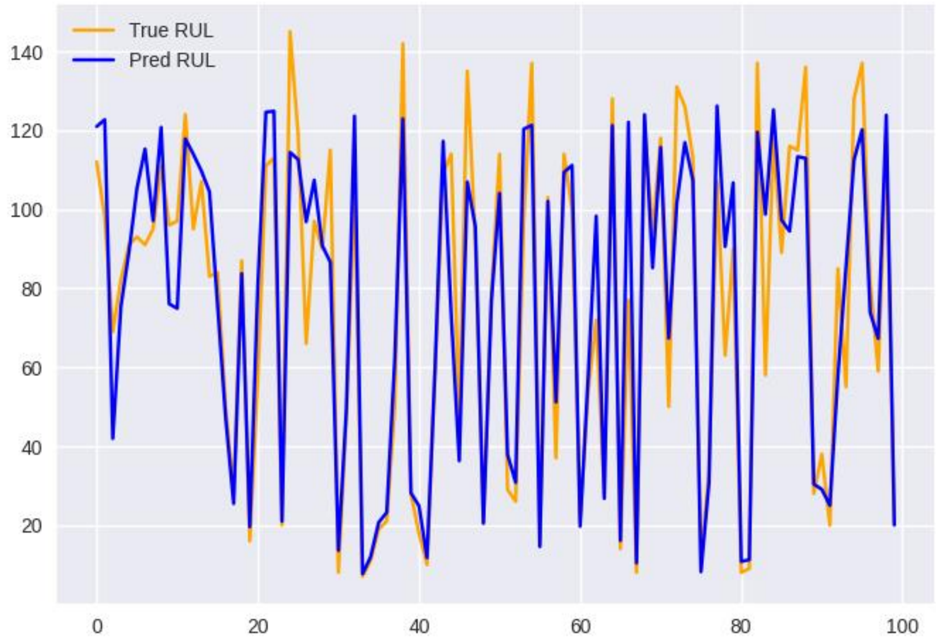
6. Final Deliverable

The results of the implemented techniques shows that ML techniques are useful in predicting Equipment Failure. It can be concluded that Random Forest, Extreme Gradient Boosting, Light GBM predict with good accuracy of more than 90 %. These models have precision, recall and f-1 score of more than 90 %. The ANN models show following results:

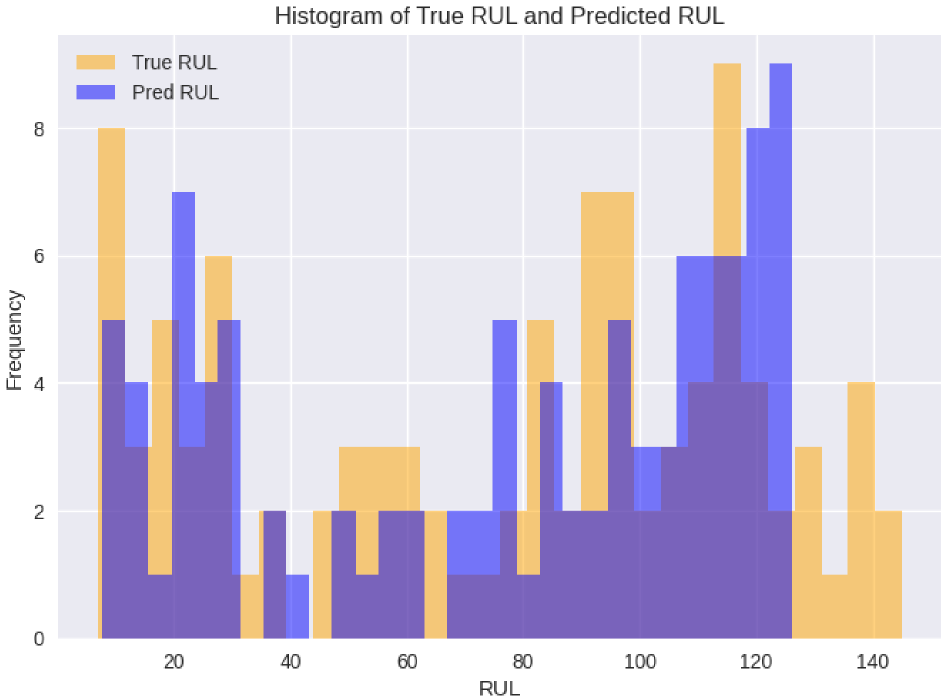
- For FD001:
  Root Mean Squared Error (RMSE)= 0.2636
  R2 score = 0.8894
- For FD002:
  Root Mean Squared Error (RMSE)= 0.1036
  R2 Score = 0.9832
- For FD003:
  Root Mean Squared Error (RMSE)= 0.2775
  R2 score = 0.8778
- For FD004:
  Root Mean Squared Error (RMSE)= 0.2928
  R2 score = 0.8674

The LSTM model shows

- Mean Absolute Error (MAE): 11.336
- Mean Squared Error (MSE): 239.498
- Root Mean Squared Error (RMSE): 15.475
- R^2 Score: 0.86131



This is the plot of True (actual) RUL value to the predicted RUL using LSTM model. The below figure shows the histogram of true RUL vs The predicted RUL.



7. Innovation in Implementation

The work done in this domain is very advanced. In past few years researchers have used multiple machine learning and deep learning techniques to find the solution to the problem of predicting RUL precisely. The Deep Neural Networks, CNNs, LSTM and their different variants are introduced. But as a beginner and given, the limited time, I only got to explore and about the Machine learning techniques. I also tried to implement CNN, ANN, and LSTM but the result I got are not much satisfactory and the implemented models have much scope for improvement. I tried to implemented my learning and tried to formulate a similar problem i.e., a simplified version of this problem, by converting it to a Classification Problem, where the class labels will be of 3 types, i.e., Good Condition represented by 0, Moderate Condition represented by 1 and Warning Condition represented by 2. Warning condition means that the maintenance is required and must be done to reduce the risk of failure of engine. If we get 0 or 1 then it means the engine is working fine and no need of maintenance. To defining the class labels, I define the engine's condition with Life_Ratio (LR), which is the ratio between Current Cycle and the End cycle (end_of_life). If LR=0, that means the component has just started its degradation and LR=1 means, it is completely degraded.

- If LR <= 0.6 - Good Condition (0)
- if LR <= 0.8 - Moderate Condition (1)
- if 0.8 < LR - Warning Condition (2)

In this way, I think the problem is simplified and we also got good results.

8. Scalability to solve industrial Problem

By implementing ML models to predict equipment failure for predictive maintenance, one technique is to use historical, available data to foresee when equipment failure is likely to occur and proactively address it with maintenance. We can use a classification or regression techniques to predict the possibility of failure in the next steps or the time left before a system fails based on various types of data such as vibration, temperature, etc. To solve industrial problems using equipment failure prediction, one approach is to use machine learning techniques such as deep neural networks, support vector machines, random forests, etc. that can perform fault detection and prognosis based on various types of data such as vibration, acoustic emission, temperature, etc. Another approach is to use a transfer learning algorithm based on time windows that can extract richer features from time series data. We can also use automated machine learning that can automatically select the best models and hyperparameters without manual effort and time, for predicting the number of days until the next failure or the probability of failure in a fixed number of days. Also, a framework should be used to formulate models and identify algorithms that can solve large-scale industrial planning problems with limited computation capacity and extend capabilities beyond solving mathematical programming models.

9. References

[1] Biggio, L., Wieland, A., Chao, M. A., Kastanis, I., & Fink, O. (2021). Uncertainty-aware remaining useful life predictor. arXiv preprint arXiv:2104.03613.

[2] Maschler, B., Vietz, H., Jazdi, N., & Weyrich, M. (2020, September). Continual learning of fault prediction for turbofan engines using deep learning with elastic weight consolidation. In 2020 25th IEEE international conference on emerging technologies and factory automation (ETFA) (Vol. 1, pp. 959-966). IEEE.

[3] Ellefsen, A. L., Bjørlykhaug, E., Æsøy, V., Ushakov, S., & Zhang, H. (2019). Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. Reliability Engineering & System Safety, 183, 240-251.

[4] Wu, D., Jennings, C., Terpenny, J., Gao, R. X., & Kumara, S. (2017). A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests. Journal of Manufacturing Science and Engineering, 139(7), 071018.

[5] Shariff, A., Misriya, N., Banu, S., & Shalini, R. A. Prediction of Remaining Useful Lifetime (RUL) of Turbofan Engine using Machine Learning.

[6] Wang, C., Peng, Z., & Liu, R. (2022). Prediction of Aero-Engine Remaining Useful Life Combined with Fault Information. *Machines*, *10*(10), 927.

[7] Nacchia, M., Fruggiero, F., Lambiase, A., & Bruton, K. (2021). A systematic mapping ofthe advancing use of machine learning techniques for predictive maintenance in the manufacturing sector. Applied Sciences, 11(6), 2546.

[8] Hesser, D. F., & Markert, B. (2019). Tool wear monitoring of a retrofitted CNC milling machine using artificial neural networks. Manufacturing letters, 19, 1-4.

[9] Lubomski, J. F. (1980, January). Status of NASA full-scale engine aeroelasticity research. In *Struct., Structural Dyn., and Mater. Conf.* (No. NASA-TM-81500).