MACHINE LEARNING FINAL PROJECT REPORT

Topic: Remaining Useful Life (RUL) Prediction Using NASA CMAPSS Turbofan Engine Dataset

Date: 05.12.2025

Prepared by:

Adeola KUMAPAYI

Mullainathan VENKATCHALAPATHY HEMAMALINI

Rohith RAMACHANDRAN JAYA SANKAR

1. Introduction

Predicting the Remaining Useful Life (RUL) of turbofan jet engines is a key problem in Prognostics and Health Management (PHM). The NASA CMAPSS dataset provides multivariate time-series data from multiple simulated engines operating until failure.

This project develops a complete machine learning pipeline including:

- Exploratory Data Analysis (EDA)
- Data preprocessing and feature engineering
- Baseline models
- Hyperparameter tuning
- Feature selection
- Ensemble learning
- Deep learning (LSTM) model for temporal degradation modeling

The dataset versions FD001–FD004 are analyzed to understand sensor correlations and degradation progression.

2. Dataset Description

The CMAPSS dataset contains multivariate time-series data for engines operating under different conditions:

Subset	Operating Conditions	Fault Modes
FD001	1	1
FD002	6	1
FD003	1	2
FD004	6	2

Each row represents an engine cycle and includes:

- 3 operational settings
- 21 sensor measurements
- Engine ID and cycle number

The goal is to predict the Remaining Useful Life (RUL) at each cycle.

3. Step 1 — Exploratory Data Analysis

3.1 Data Quality and Structure

All four subsets were loaded, cleaned, and standardized. No missing values were present, but sensor distributions varied significantly across subsets with multiple operating conditions (FD002, FD004).

Basic checks confirmed:

- No negative cycle values
- No inconsistent engine IDs
- Sensors with constant values detected (e.g., some sensors in FD001 show limited variability)

3.2 Correlation Analysis

Strong correlations were found among several temperature and pressure sensors, indicating redundant information and potential feature grouping.

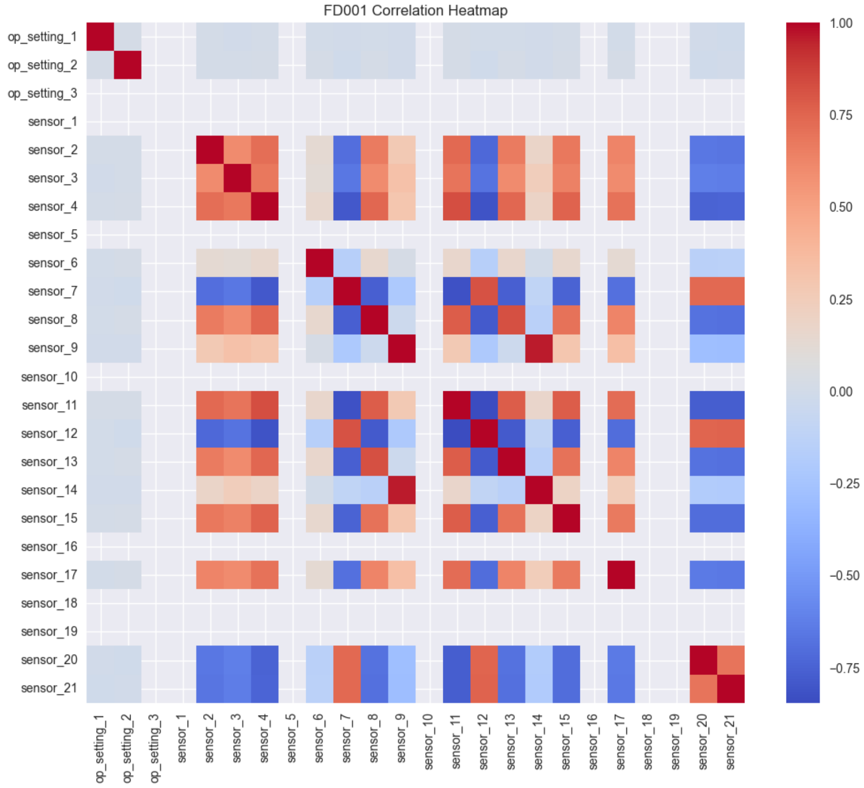

Figure 1: *Correlation Heatmap — FD001*

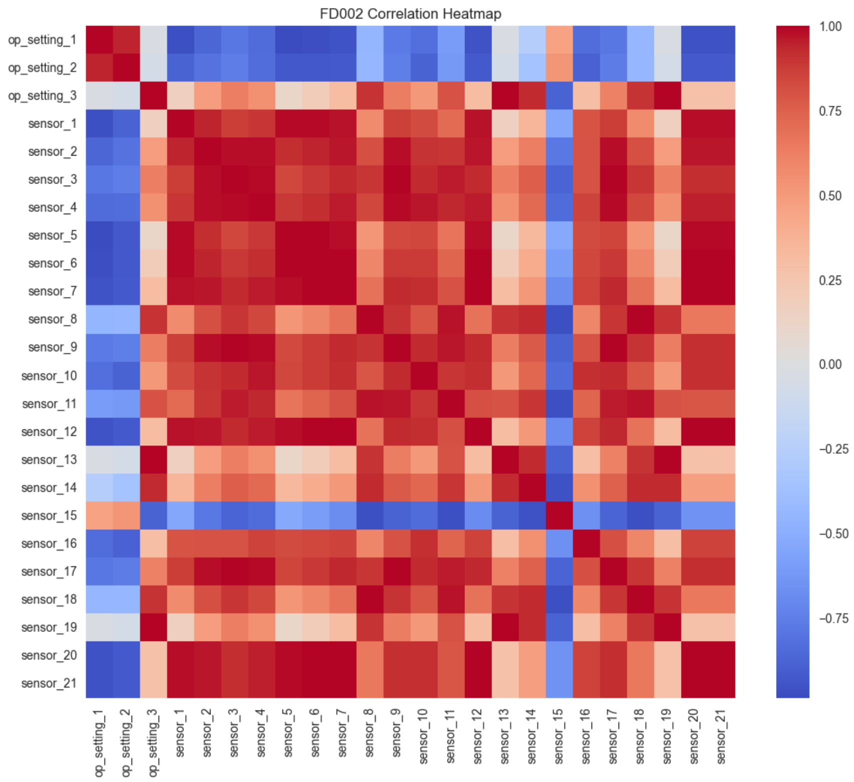

Figure 2: *Correlation Heatmap — FD002*

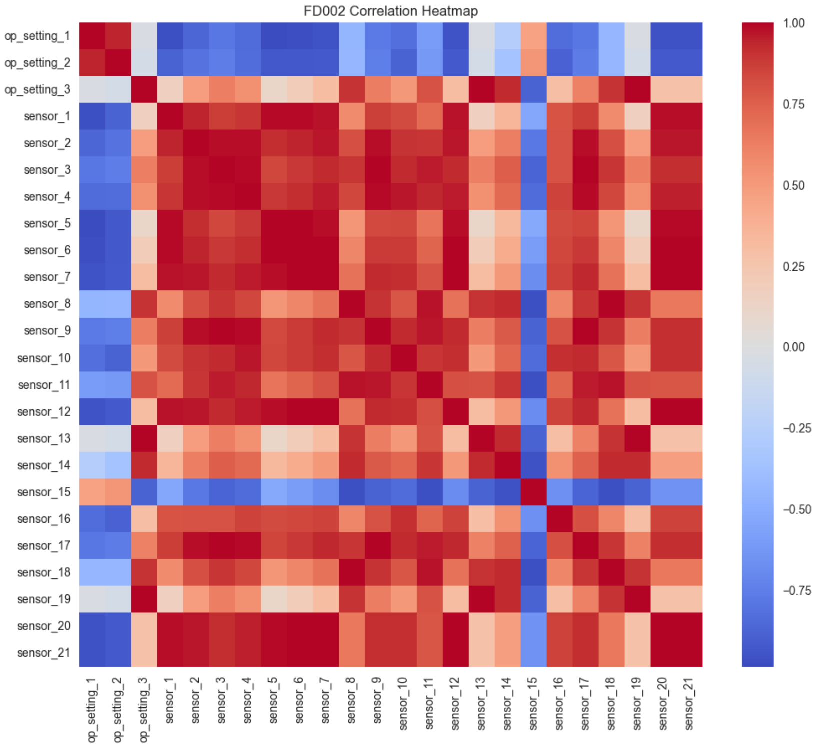

Figure 3: *Correlation Heatmap — FD003*

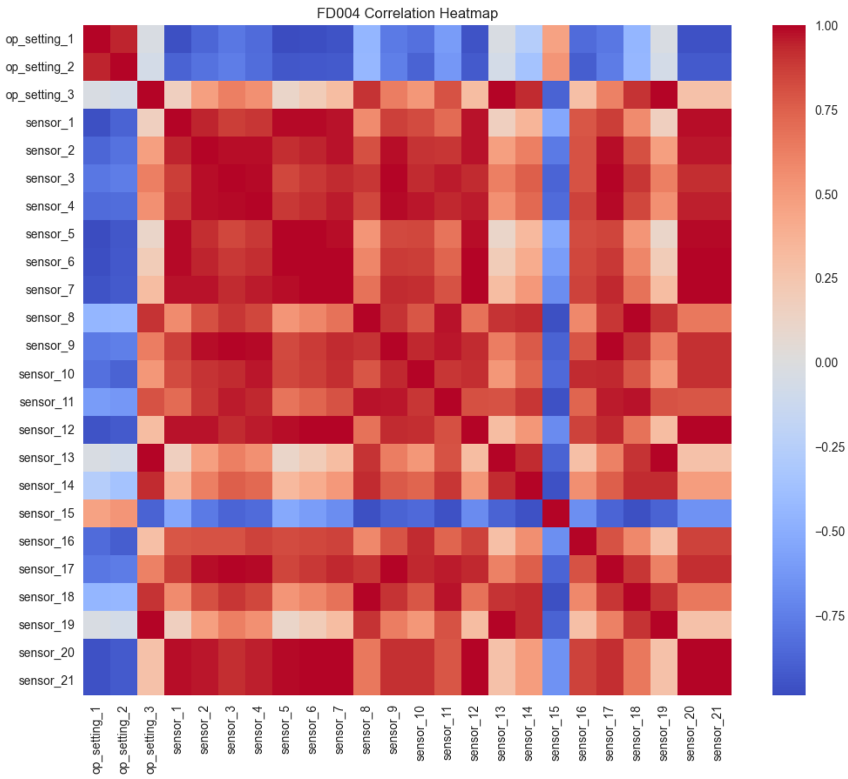

Figure 4: *Correlation Heatmap — FD004*

Insights:

- FD001 and FD003 show cleaner structure due to single operating condition.
- FD002 and FD004 display cluster blocks, reflecting multiple operating regimes.
- Sensors 2, 3, 4, 7, and 8 demonstrate particularly high mutual correlation.

3.3 Engine Lifespan Distribution

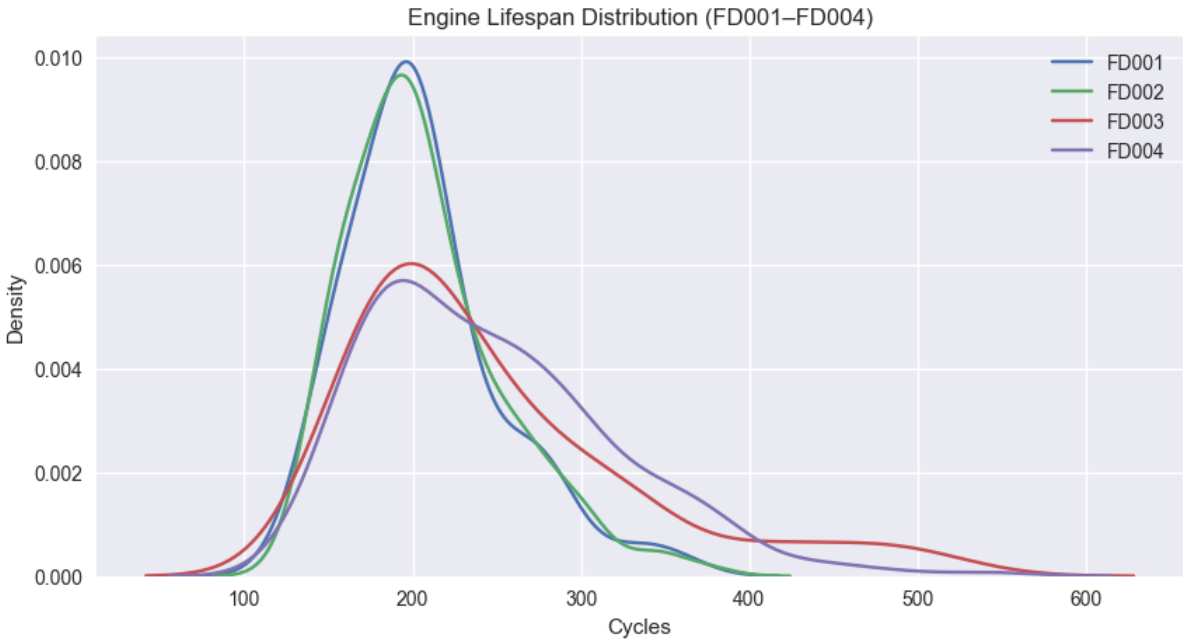

Figure 5: *Engine Lifespan Density Plot (FD001–FD004)*

- Lifespans range from ~130 to ~600 cycles.
- FD003 and FD004 generally show longer average runtimes.
- Wide variability indicates substantial modeling difficulty.

3.4 Feature Engineering

Four major transformations were applied:

1. RUL computation per engine
2. Min–Max Scaling
3. Delta features:
   - capturing intra-engine cycle-to-cycle changes
4. Rolling window features (window=5):
   - capturing smoothed degradation patterns

Resulting feature counts:

- 75 engineered features (sensors + operations + deltas + rolling means)

4. Step 2 — Baseline Models

Three classical ML models were trained using GroupKFold (engine-wise splitting):

Models Evaluated

- Decision Tree Regressor
- Random Forest Regressor
- Support Vector Regressor (RBF)

Performance Summary

Model	MAE	RMSE
Decision Tree	**32.30**	**45.97**
Random Forest	**29.47**	**42.00**
SVR (RBF)	**32.35**	**44.53**

Random Forest emerges as the strongest baseline.

5. Step 3 — Hyperparameter Tuning

A computationally efficient hyperparameter search was performed using RandomizedSearchCV with GroupKFold.

Tuned Results

Model	Best MAE	Best Hyperparameters
Random Forest	29.31	{n_estimators=120, max_depth=10, max_features='sqrt'}
Decision Tree	31.69	{max_depth=5}
SVR (RBF)	29.04	{C=10, ε=0.01, γ='scale'}

Both Random Forest and SVR outperform baseline versions.

6. Step 4 — Feature Selection

Two complementary methods were used:

- Random Forest Feature Importance
- Mutual Information Regression

Intersection of top-performing features yielded a reduced set of 26 features, significantly lowering feature dimensionality while preserving predictive power.

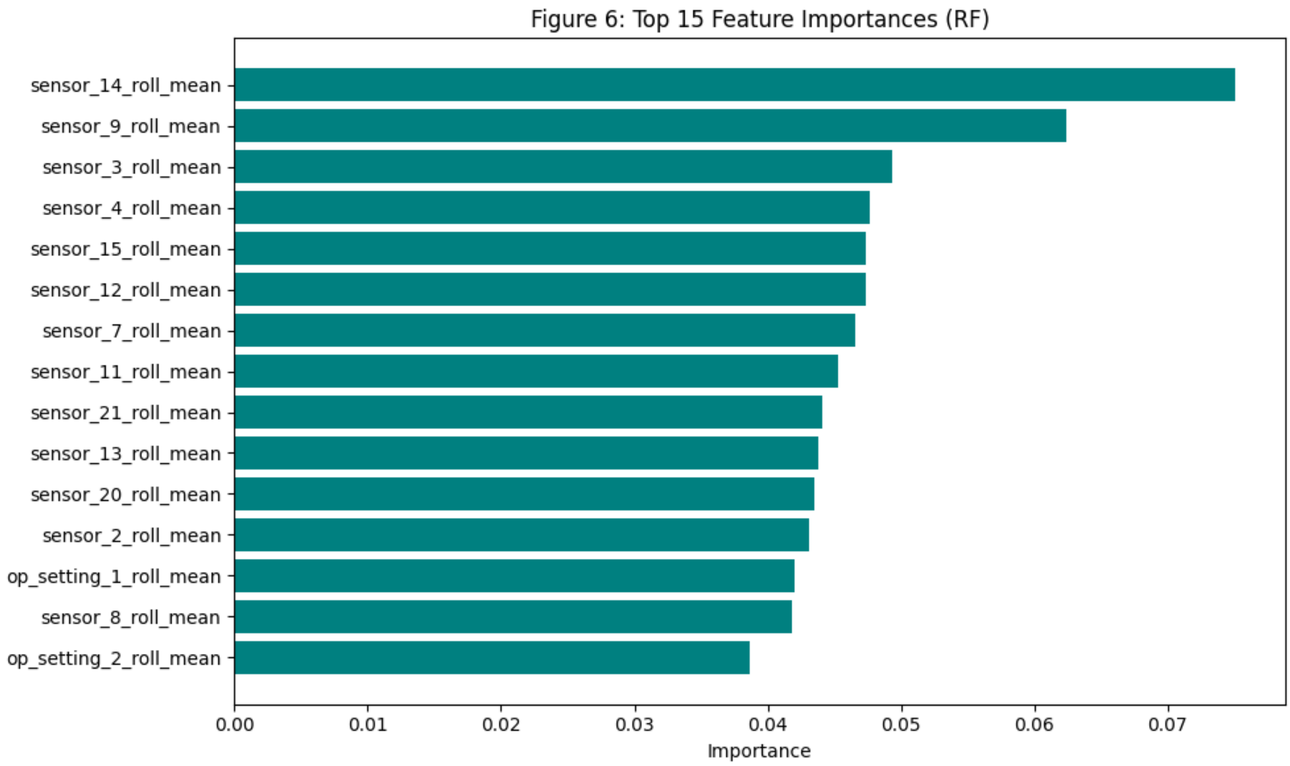

Figure 6: *Top Feature Importance Bar Plot*

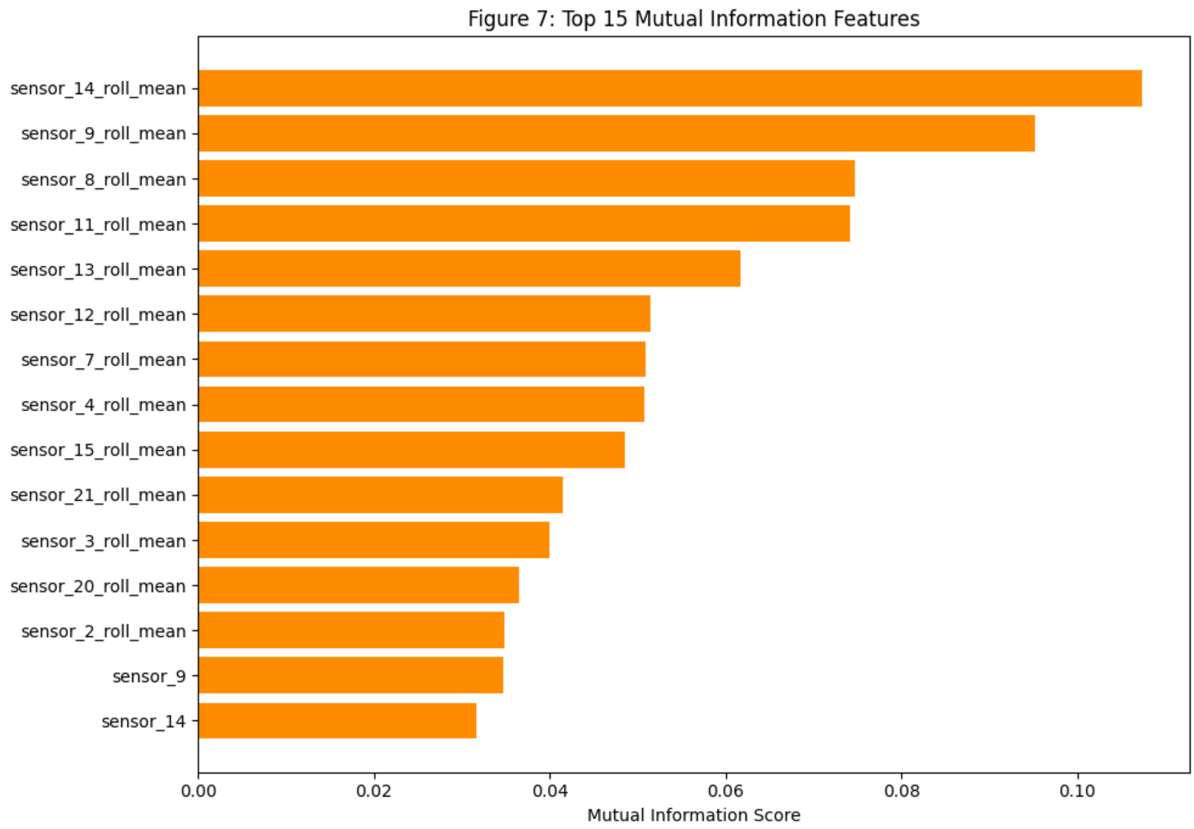

Figure 7: *Mutual Information Ranking Plot*

Result

- Original features: 75
- Selected: 26
- New dataset shape: *(20231 samples, 26 features)*

This reduced dataset was used for ensemble models and LSTM training.

7. Step 5 — Ensemble Learning

Three different ensemble methods were tested:

7.1 Weighted Ensemble

Uses optimized weights:

- 60% SVR
- 40% Random Forest

7.2 Voting Ensemble

Simple average of predictions.

7.3 Stacking Ensemble

Meta-learner: Linear Regression over RF + SVR outputs.

Cross-Validation Results (Reduced Dataset)

Ensemble	MAE (Mean)	Std Dev
Weighted Ensemble	**23.91**	**0.81**
Voting Ensemble	**28.93**	**1.06**
Stacking Ensemble	**30.73**	**1.49**

The Weighted Ensemble is the strongest model so far.

Huge improvement over Step 2 baselines:

- From MAE ≈ 29 → MAE ≈ 23.9

8. Step 6 — Deep Learning Approach (LSTM)

8.1 Motivation

Traditional machine learning models such as Random Forest, SVR, or Decision Tree treat each row of data as an **independent observation**. However, jet engine degradation is inherently **sequential**. The health state at cycle *t* is strongly influenced by sensor evolution during earlier cycles *(t–1, t–2, …)*.

This means the dataset contains:

- **Temporal correlations** (patterns over time)
- **Long-term dependencies** (slow degradation, trends)
- **Short-term fluctuations** (noise from load variations)
- **Nonlinear relationships** between sensors

Classical ML cannot directly model these sequential structures, because they assume i.i.d. data (independent and identically distributed).

8.2 LSTM Overview

A **Long Short-Term Memory (LSTM)** network is a special type of Recurrent Neural Network (RNN) that solves the limitations of traditional RNNs by introducing **gates** and **cell states**, enabling it to:

- remember information over long time spans
- forget irrelevant information
- update internal memory selectively

LSTM contains three gates:

1. **Forget Gate**
    - Decides which past information should be discarded.
2. **Input Gate**
    - Determines what new information should be stored.
3. **Output Gate**
    - Controls what information from the internal memory (cell state) is used to produce the output.

Because of these mechanisms, LSTMs can learn long-term degradation patterns such as:

- gradual increase in temperature
- persistent vibrations
- pressure decay
- slow wear accumulated over cycles

8.3 Sequence Preparation

- Sequence length: 30
- Features: 26 selected features
- Shape:
    - X_seq = (17231, 30, 26)
    - y_seq = (17231,)

8.4 LSTM Architecture

- LSTM layers: 2
- Hidden size: 64
- Dropout: 0.2
- Learning rate: 0.001
- Epochs: 5
- Batch size: 128
- Loss: MSE

8.5 Training Output

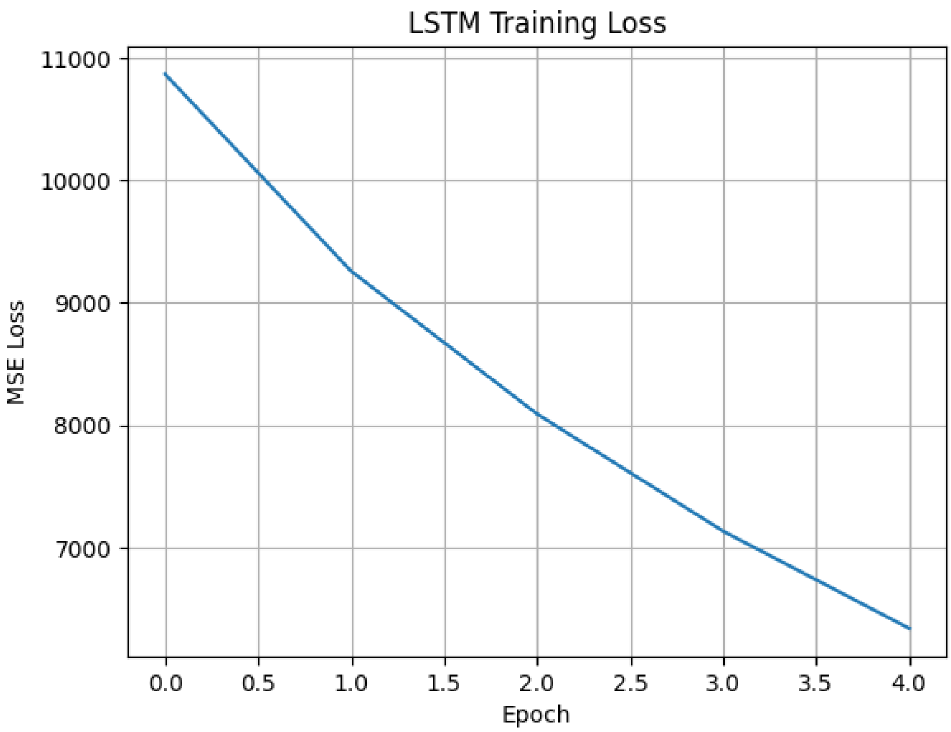

Figure 8: *LSTM Training Loss Curve*

Loss decreased steadily:

Epoch	Loss
1	10866.66
2	9257.15
3	8089.96
4	7132.55
5	6338.74

8.5 Model Performance

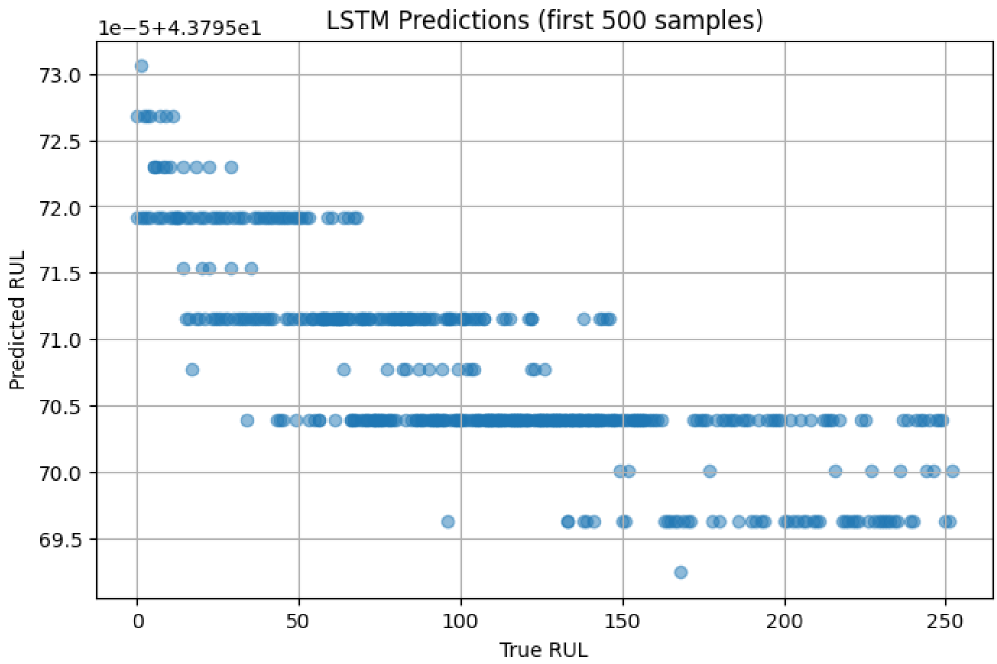

Figure 9: *LSTM Prediction Scatter Plot (first 500 samples)*

Metric	Value
MAE	59.41
RMSE	77.45

Interpretation:

- Predictions collapse into a narrow band → underfitting
- Shallow architecture + limited epochs + reduced features
- LSTM underperforms vs. ensemble models

9. Comparative Summary

Model Performance Ranking (Lower MAE = Better)

Rank	Model	MAE
1	Weighted Ensemble	23.91
2	Tuned SVR	29.04
3	Tuned Random Forest	29.31
4	Decision Tree	31.69
5	LSTM (5 epochs)	59.41

Conclusion:
➡ Traditional ML + engineered features outperform LSTM under current training settings.
➡ LSTM needs deeper architecture and longer training to compete.

10. Conclusion and Next Steps

This project built a complete RUL prediction pipeline from EDA to advanced ML:

- Strong EDA insights into sensor correlations and degradation
- Baselines established
- Tuned models improved accuracy
- Feature selection reduced dimensionality
- Ensemble techniques delivered the best performance
- LSTM shows potential but requires further optimization

Next Steps :

- Train deeper LSTM or GRU (128–256 units)
- Use 50–100 cycle sequence length
- Include all 75 engineered features
- Train for 30–50 epochs
- Explore CNN-LSTM or Transformer architectures
- Use FD002–FD004 for model generalization testing

GitHub Link: https://github.com/mullainathan29/turbofan-ml-project-scaffold