Hybrid Predictive Maintenance using NASA CMAPSS (FD001)

Developed with Streamlit and Machine Learning

Introduction & Problem Statement

Predictive Maintenance (PdM) is a technique that uses data analysis tools to detect anomalies in operations and possible defects in equipment and processes. The ability to predict equipment failure before it occurs is a major advantage in industries like aviation, manufacturing, and energy. This project leverages the NASA CMAPSS (FD001) dataset to build a hybrid machine learning system for PdM that predicts equipment degradation stages and estimates Remaining Useful Life (RUL).

The CMAPSS dataset contains time-series sensor data for aircraft engines. Each engine runs until failure, and sensor readings capture operational conditions over time. The main goals of this project are:

- Classify degradation stages using clustering and classification models.
- Predict RUL using regression models.
- Display results and alerts through an interactive Streamlit dashboard.

This system offers a cost-effective and scalable way to ensure machinery remains operational and failures are minimized through timely intervention.

Dataset Overview & Preprocessing

The NASA CMAPSS dataset used in this project contains multi-variate time-series data for turbofan engine degradation. The train_FD001.txt file contains the following:

- 100 different engines (unit numbers)
- Operational cycles until failure for each engine
- 3 operational settings

- 21 sensor measurements

Preprocessing Steps:

- Drop columns with only whitespace (columns 26 and 27)
- Assign proper column names for easier readability
- Use only sensor readings for modeling

This results in a cleaned dataset suitable for clustering and ML modeling. Additional features like RUL (Remaining Useful Life) are engineered later during the regression phase.

Furthermore, by analyzing trends in operational and sensor data, we better understand the degradation patterns, helping refine feature selection and improve model performance.

Methodology / Code Structure

The implementation is done using the following steps:

1. **Data Loading**: Reads train_FD001.txt, assigns column names, and drops null columns.
2. **Clustering**: Applies KMeans clustering (5 clusters) to identify degradation stages based on sensor features.
3. **Dashboard Sidebar**: Allows users to select a specific engine ID to visualize degradation over cycles.
4. **Stage Classification**:
    a. Features: All sensor readings.
    b. Target: Cluster labels.
    c. Classifier: RandomForestClassifier.
5. **RUL Prediction**:
    a. Calculates RUL = Max cycle - Current cycle.
    b. Regressor: RandomForestRegressor.
6. **Risk Alert System**: Combines classification and regression outputs to flag high-risk engines.
7. **Confusion Matrix**: Plots performance of classifier using Seaborn heatmap.

Technologies used:

- **Streamlit** for dashboard

- **Scikit-learn** for machine learning models
- **Matplotlib & Seaborn** for visualizations

Visual Outputs & User Interaction

The dashboard includes interactive and visual tools for users to explore the dataset and monitor engine health.

- **Line Chart**: Displays sensor readings over cycles for a selected engine, allowing visual inspection of degradation trends.
- **Classification Report**:
  - Displays precision, recall, f1-score for each of the five degradation stages.
  - Helps assess model performance at distinguishing engine conditions.
- **Regression Metrics**:
  - RMSE (Root Mean Squared Error): Indicates how accurate RUL predictions are.
  - R^2 Score: Shows how well the model explains the variance in RUL.
- **Alerts Table**:
  - Flags engines with severe predicted stages, low RUL, and high probability of failure.
- **Confusion Matrix**:
  - Heatmap shows classification accuracy for each stage, useful for understanding model errors.

These outputs are presented in real-time, making the system useful for decision support in real maintenance operations.

The layout and interactivity enhance user experience, providing both engineers and managers with insights into engine performance and potential maintenance needs.

Results and Observations

Clustering:

- KMeans identified five distinct degradation stages.
- Stages align with engine wear and tear over time.

Classification:

- RandomForestClassifier performed well with balanced precision and recall.
- Sensor fusion helped improve accuracy over using individual sensor data.

Regression:

- RMSE: A lower value indicates reliable RUL predictions.
- R^2 Score: A value closer to 1 shows strong correlation between actual and predicted RUL.

Risk Alerts:

- System accurately raised flags for engines likely to fail soon.
- Combined probability and RUL predictions to assess criticality.

Overall, the hybrid approach improved fault detection accuracy and helped in pre-empting breakdowns by highlighting potential high-risk conditions.

Evaluation & Model Interpretability

Model evaluation goes beyond accuracy metrics. In real-world applications, understanding model behavior and reliability is equally critical.

Additional Evaluation Points:

- **Cross-validation** was applied to avoid overfitting and validate model generalizability.
- **Feature Importance** (in Random Forests) helps identify which sensors contribute most to predictions.

Interpretability:

- Adding tools like SHAP (SHapley Additive exPlanations) in future can help explain individual predictions.
- Understanding sensor influence helps refine monitoring systems and sensor placement strategies.

These aspects make the system not only functional but also transparent, which builds trust in predictive models for industrial users.

Conclusion

This project showcases a complete pipeline from raw sensor data to real-time maintenance alerts using machine learning. The combination of unsupervised learning (clustering), supervised learning (classification and regression), and interactive dashboards makes the system versatile and informative.

Advantages of this approach:

- Adaptable to other industrial machines.
- Scalable and can be deployed with live sensor feeds.
- Reduces maintenance cost and prevents unexpected breakdowns.

By integrating data visualization and machine learning, this project delivers a powerful tool for predictive maintenance that can be extended to smart manufacturing and Industry 4.0 applications.

References, Limitations & Future Work

References:

- NASA CMAPSS Data Repository
- scikit-learn: Machine Learning in Python
- Streamlit Documentation
- Sensor analytics research papers

Limitations:

- Fixed number of clusters may not generalize across other datasets.
- Random Forest lacks temporal modeling of sequences.
- Dataset assumes failure occurs at final cycle which may not represent real-world gradual failure.
- No domain expert labels in training data — clusters are assumed stages.

Future Enhancements:

- Use LSTM or GRU models for time-series RUL prediction.
- Integrate anomaly detection for unseen failure types.
- Add real-time streaming data support.
- Feature importance analysis using SHAP.
- Enhance UI for maintenance scheduling integration.

This project lays the groundwork for smart maintenance systems that are proactive, data-driven, and industry-ready.