Week 3 – Model Development and Training

Introduction:

The SmartFactory.AI project's third week is devoted to creating and assessing machine learning models for failure prediction, after the establishment of preprocessed, feature-rich datasets in Week 2. The objective is to use supervised learning to transform designed traits into intelligence that can be put to use. This week, prediction performance is benchmarked and important contributing elements to machine failures are interpreted using both linear and non-linear techniques. The results will be used as early maintenance indications to help predict malfunctions before they happen.

Objective:

The third week's goals are:

- To anticipate machine breakdown, create and train reliable categorization models.
- Compare the results of two baseline models: Random Forest (robust, non-linear) and Logistic Regression (interpretable, linear).
- Use ROC curves, precision, recall, F1-score, and confusion matrices to quantitatively assess models.
- See the Random Forest model's top contributing features.
- Store test data and trained models for further validation and deployment.

Methodology:

A full machine learning pipeline from training to assessment is part of the procedure:

- **Data Preparation and Loading:**
    - The script loads processed_sensor_data.json first, then transforms it into a pandas DataFrame.
    - The non-feature columns failure, machine_id, and timestamp are dropped.
    - For binary classification, the goal variable failure is utilized (1 = failure, 0 = normal).
- **Train-Test Division:**
    - To guarantee strong generalization:
    - Train_test_split is used to stratify the dataset by failure to preserve class balance, dividing it into 80% training and 20% testing.

  - This guarantees that during the training and testing stages, the model experiences a representative distribution of failures.

- **Model Pipelines:**
  To simplify preprocessing and training, two machine learning models are benchmarked and each is encapsulated in a scikit-learn pipeline:
  - **Logistic Regression (LR):**
    - L2 regularization is used in logistic regression (LR), which provides interpretable insights into feature contribution.
    - Mean imputation for missing values is part of the pipeline.
  - **Random Forest (RF):**
    - An ensemble-based technique for capturing non-linear interactions is Random Forest (RF).
    - Strong baseline since it can handle a variety of feature types and is resistant to overfitting.

  .fit() function is used to train both pipelines on the training data (X_train, y_train).

- **Model Evaluation Function:**
  A common evaluate_model function is used to:
  - Show a classification report that includes the F1-score, accuracy, recall, and support.
  - To compare the actual and anticipated results, plot a confusion matrix.
  - To evaluate the model's performance across several thresholds, create a ROC curve.
  - Determine the overall model quality using the ROC-AUC score.

  By doing this, LR and RF may be compared side by side.
- **Random Forest Feature Importance:**
  To extract interpretability from the Random Forest:
  - It imputes the training set on its own.
  - To display the top ten most significant predictors of machine failure, feature importances are extracted and shown.
  - This identifies the model's most informative properties and can help domain experts decide on an operational emphasis.
- **Artefact preservation:**
  To facilitate deployment and reproducibility:
  - Joblib.dump() is used to store both trained pipelines as.pkl models.
  - For usage in Week 4 and beyond, the test datasets (X_test and y_test) are stored in JSON format.

Results & Insights:

- **Performance of the Model:**
  - Logistic regression: Offers a solid baseline and performs well in terms of interpretability. Ideal for uses where openness is essential, including risk explanations and audits.
  - Because Random Forest can capture intricate relationships between constructed features, it performs better than logistic regression in terms of ROC-AUC and F1-score.
- **Confusion Matrix Observations:**
  - While RF has superior recall and fewer false positives, both models detect the majority of failure instances.
  - In manufacturing, where failing to detect a breakdown (false negative) can lead to expensive downtime, this is crucial.
- **ROC Curve and AUC Score:**
  - Both models' ROC curves show good discriminatory abilities, however Random Forest's higher AUC indicates a better level of model confidence across thresholds.
- **Top Predictive Features:**
  According to the Random Forest, the following factors were the main contributors:
  - Mechanical instability is reflected in vibration_rolling_std.
  - temperature_rolling_mean: Shows the risk of chronic overheating.
  - Abrupt fluctuations in speed, a symptom of stress or malfunction, are captured by rpm_roc.
  - Pressure_rolling_std and humidity_rolling_std: Draw attention to how equipment is impacted by environmental volatility.

  These characteristics provide useful information for plant operators and validate the importance of Week 2's temporal engineering.

Summary:

The main AI-driven failure prediction module of SmartFactory.AI is successfully launched in Week 3. Highlights consist of:

- Two interpretable classification pipelines have been deployed.
- Thorough analysis utilizing classification reports, ROC curves, and confusion matrices.

- Analysis of feature significance for domain knowledge and model reliability.
- Deployment readiness with all artifacts (test data, models) preserved.

Conclusion:

By creating and comparing two fundamental prediction models, week three of SmartFactory.AI represents a significant turning point. Random Forest is a great contender for early deployment due to its exceptional performance and extensive interpretability.

URL To Week 3 – Model Development and Training.ipynb