📘 Technical Documentation: Hydraulic Condition Monitoring Dashboard

Contents

🏭 Industry Context & Motivation

This project simulates predictive maintenance in **industrial hydraulics**, a common subsystem in manufacturing, automotive, and heavy machinery. Unplanned hydraulic failures can cost companies significant downtime and repair expenses.

By monitoring sensor data from multiple system components — pressure, temperature, vibration, and flow — this project enables early detection of abnormal conditions, such as cooler or valve failure.

The ultimate goal: **help maintenance teams act before critical failures**, increasing uptime and safety.

🛠️ Project Scope

This end-to-end machine learning project includes:

- Loading 20+ sensor `.txt` files and the label file `profile.txt`
- Exploratory Data Analysis (EDA) of sensor behavior and label distributions
- Feature engineering: computing statistical descriptors per cycle

- Training 5 models (regression/classification) with XGBoost

- Interpreting model predictions with SHAP

- Deploying a **Streamlit dashboard** for real-time batch predictions

GitHub Repository: [Streamlit-Hydraulic-Dashboard](https://github.com/ArturMelnyk-analyst/Streamlit-Hydraulic-Dashboard)

🔄 Phase-by-Phase Process

📂 Phase 1: Data Loading

Implemented smart Google Drive mounting in Colab to enable local and cloud support.

- Loaded and renamed 20+ `.txt` files into organized pandas DataFrames

- Grouped sensors logically (e.g., pressure, motor_power)

- Used `sensor_data` dictionary to simplify reuse in later phases

📁 See: [`01_load_data.ipynb`](https://github.com/ArturMelnyk-analyst/Streamlit-Hydraulic-Dashboard/blob/main/notebooks/01_load_data.ipynb)

📊 Phase 2: Exploratory Data Analysis

- Verified no missing values in sensor streams

- Reviewed target distributions from `profile.txt`

- Plotted example channels (e.g., `pressure_t1`, `temperature_t1`)

- Built correlation heatmaps to check sensor redundancy

EDA showed relatively clean input data, suitable for modeling.

📁 See: [`02_eda.ipynb`](https://github.com/ArturMelnyk-analyst/Streamlit-Hydraulic-Dashboard/blob/main/notebooks/02_eda.ipynb)

🧪 Phase 3: Feature Engineering

- Applied 7 statistics (mean, std, min, max, median, skew, kurtosis) per group
- Created ~49 features across all sensor groups
- Verified dataset shape: (2205 samples × 49 features)

📁 See: [`03_feature_engineering.ipynb`](https://github.com/ArturMelnyk-analyst/Streamlit-Hydraulic-Dashboard/blob/main/notebooks/03_feature_engineering.ipynb)

🤖 Phase 4: Model Training

- Trained 5 models (2 regression, 3 classification) with XGBoost
- Encoded labels for multi-class tasks
- Used SHAP for interpretability

Best results:

- Accumulator Pressure RMSE: 3.17
- Cooler Condition RMSE: 4.94
- Valve Condition Accuracy: 98.4%
- Pump Leakage Accuracy: 99.7%
- Stable Flag Accuracy: 96.8%

📁 See: [`04_modeling.ipynb`](https://github.com/ArturMelnyk-analyst/Streamlit-Hydraulic-Dashboard/blob/main/notebooks/04_modeling.ipynb)

📂 Models saved: `/models/*.joblib`

📌 Phase 5: Technical Documentation & Lessons Learned

- 💡 Smart Drive Mount + `get_base_path()` made data access portable
- 🔁 Using dictionaries (`sensor_data`, `trained_models`) improved structure
- 🛠️ Feature extraction was scalable and reused across sensor groups
- 🤯 SHAP added explainability and confidence in predictions
- 💻 Streamlit was easy to prototype, but styling could be enhanced

📁 See: [`05_reflection.ipynb`](https://github.com/ArturMelnyk-analyst/Streamlit-Hydraulic-Dashboard/blob/main/technical_docs/technical_documntation.pdf)

🌐 Phase 6: Streamlit Dashboard

Created a clean, responsive dashboard:

- Upload `.csv` with engineered features
- Get model predictions in browser
- Download results as `.csv`

📁 See: [`streamlit_dashboard_app.py`](https://github.com/ArturMelnyk-analyst/Streamlit-Hydraulic-Dashboard/blob/main/streamlit_dashboard_app.py)

⚠️ Would improve with better UI (e.g., tabs, visuals).

🎯 Final Thoughts

This was a complete **production-style machine learning pipeline**.

It showcased:

- Data wrangling and analysis
- Reusable function design
- Multi-output modeling
- Explainability (SHAP)
- Frontend deployment (Streamlit)

💡 If time allowed, I'd explore:

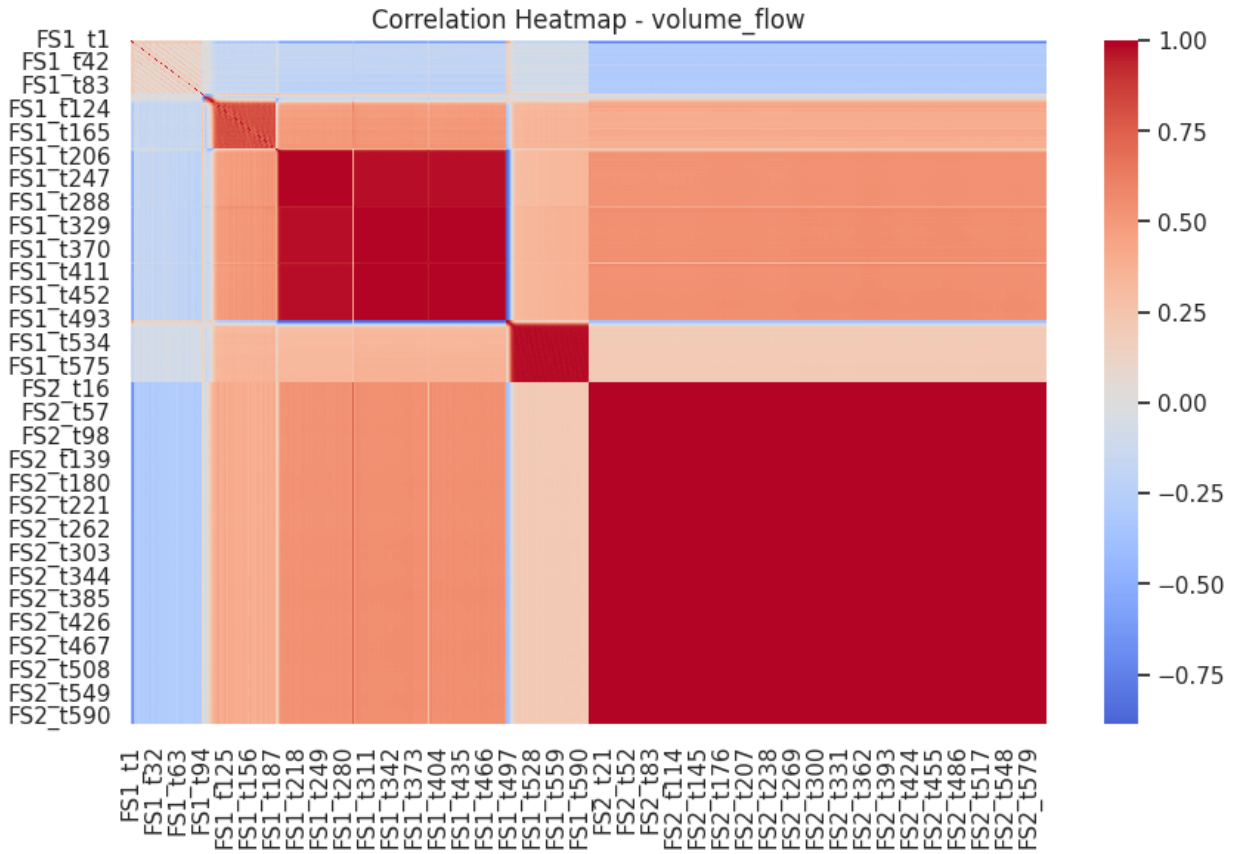
- Real-time API integration
- Concept drift simulation
- Data versioning with DVC

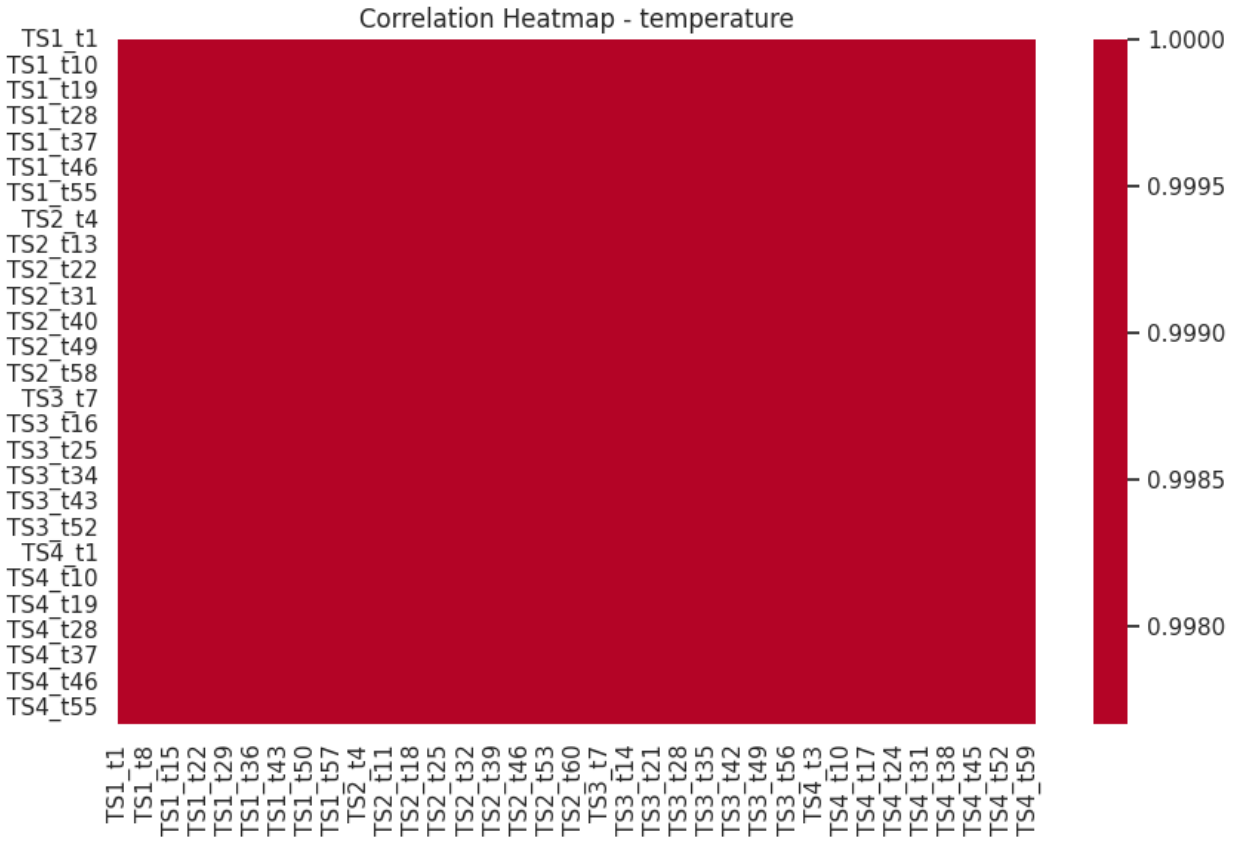
📓 Visual Appendix

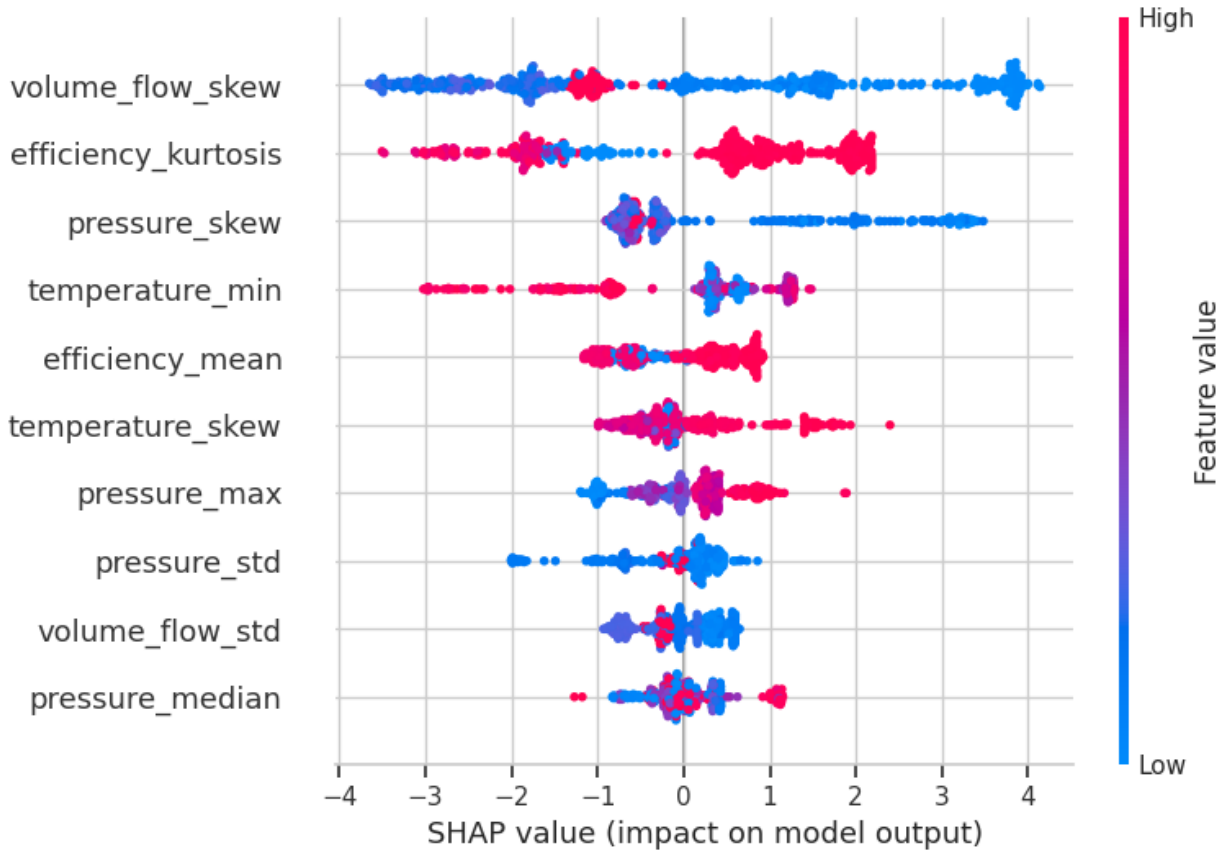
```
🔧 Training model for: Cooler_Condition
✅ RMSE: 4.94 | MAE: 0.48

🔧 Training model for: Valve_Condition
/usr/local/lib/python3.11/dist-packages/xgboost/core.py:158: UserWarning: [13:55:17] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
✅ Accuracy: 0.9841
              precision    recall  f1-score   support

           0       0.99      1.00      0.99        88
           1       1.00      0.97      0.98        67
           2       0.93      1.00      0.96        80
           3       1.00      0.98      0.99       206

    accuracy                           0.98       441
   macro avg       0.98      0.99      0.98       441
weighted avg       0.99      0.98      0.98       441
```

```
🔧 Training model for: Pump_Leakage
/usr/local/lib/python3.11/dist-packages/xgboost/core.py:158: UserWarning: [13:55:18] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
✅ Accuracy: 0.9977
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       228
           1       1.00      1.00      1.00       101
           2       1.00      0.99      1.00       112

    accuracy                           1.00       441
   macro avg       1.00      1.00      1.00       441
weighted avg       1.00      1.00      1.00       441
```

```
🔧 Training model for: Accumulator_Pressure
✅ RMSE: 3.17 | MAE: 1.78

🔧 Training model for: Stable_Flag
/usr/local/lib/python3.11/dist-packages/xgboost/core.py:158: UserWarning: [13:55:19] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
✅ Accuracy: 0.9683
              precision    recall  f1-score   support

           0       0.96      0.99      0.98       294
           1       0.99      0.92      0.95       147

    accuracy                           0.97       441
   macro avg       0.97      0.96      0.96       441
weighted avg       0.97      0.97      0.97       441
```




Correlation Heatmap - temperature
TS1_t1
TS1_t10
TS1_t19
TS1_t28
TS1_t37
TS1_t46
TS1_t55
TS2_t4
TS2_t13
TS2_t22
TS2_t31
TS2_t40
TS2_t49
TS2_t58
TS3_t7
TS3_t16
TS3_t25
TS3_t34
TS3_t43
TS3_t52
TS4_t1
TS4_t10
TS4_t19
TS4_t28
TS4_t37
TS4_t46
TS4_t55
TS1_t1
TS1_t8
TS1_t15
TS1_t22
TS1_t29
TS1_t36
TS1_t43
TS1_t50
TS1_t57
TS2_t4
TS2_t11
TS2_t18
TS2_t25
TS2_t32
TS2_t39
TS2_t46
TS2_t53
TS2_t60
TS3_t7
TS3_t14
TS3_t21
TS3_t28
TS3_t35
TS3_t42
TS3_t49
TS3_t56
TS4_t3
TS4_t10
TS4_t17
TS4_t24
TS4_t31
TS4_t38
TS4_t45
TS4_t52
TS4_t59
1.0000
0.9995
0.9990
0.9985
0.9980


volume_flow_skew
efficiency_kurtosis
pressure_skew
temperature_min
efficiency_mean
temperature_skew
pressure_max
pressure_std
volume_flow_std
pressure_median
−4
−3
−2
−1
0
1
2
3
4
SHAP value (impact on model output)
High
Feature value
Low

Repository structure:

```
hydraulic_dashboard/
 ├── data/                  # Raw sensor files (.txt)
 ├── models/                # Saved .joblib models
 ├── notebooks/
 |   ├── 01_load_data.ipynb
 |   ├── 02_eda.ipynb
 |   ├── 03_feature_engineering.ipynb
 |   ├── 04_modeling.ipynb
 ├── technical_docs/
 |   ├── technical_documentation.pdf
 |   └── architecture.pdf
 ├── streamlit_dashboard_app.py
 ├── .gitignore
 └── README.md
```

Thanks for reviewing!