Testing Plan: AI-Based Anomaly Detection in Spacecraft Telemetry Data

Objective

To Validate the AI-based anomaly detection system using Isolation Forest and LSTM autoencoder for spacecraft telemetry data, ensuring >90% precision, >95% recall, <1 second latency, and robustness across edge cases, as outlined in the solution plan by Jazmine Brown and Troy Nsofar.

Testing Goals

- Confirm the system detects anomalies with high precision (>90%) and recall (>95%).
- Ensure robustness against noisy or incomplete telemetry data.
- Verify real-time performance (<1 second latency per data window).
- Address interpretability by providing clear, actionable alerts.

Test Data

- **Sources**:
    - **NASA SMAP Dataset**:- 1 million data points 10 sensors, 10 Hz sampling 3 hours of telemetry 100 MB.
    - **Curiosity Dataset**: 500,000 data points 5 sensors, 5 Hz sampling, 5 hours 50 MB.
    - **Synthetic Data**: 100,000 data points with 20% injected anomalies.
- **Characteristics**:
    - Normal data: 80% of dataset (e.g., 10,000 60-second windows at 10Hz).
    - Anomalous data: 20% with labeled anomalies (e.g., 2,000 windows with spikes or drops).
    - Edge cases: 10% missing values, Gaussian noise (SNR=10), or extreme readings (±5σ).
- **Data Split**:
    - Training: 70% (normal data only for LSTM autoencoder; mixed for Isolation Forest).
    - Validation: 15% (mixed normal and anomalous for threshold tuning).
    - Testing: 15% (mixed normal and anomalous for final evaluation).

Testing Methods

1. **Unit Testing**:
    - **Preprocessing**: Verify data cleaning (no NaN values), normalization (values in [0, 1]), and windowing (correct 60-second segments).
    - **Isolation Forest**: Ensure model outputs anomaly scores (0 to 1) for static data.
    - **LSTM Autoencoder**: Confirm reconstruction errors are generated for time-series inputs.
    - **Alert System**: Test API delivers JSON alerts with correct format (e.g., timestamp, sensor, score).
2. **Integration Testing**:
    - Test end-to-end pipeline: Telemetry input → preprocessing → model (Isolation Forest or LSTM) → alert output.
    - Verify API integration with a mock mission control dashboard.
    - Evaluate edge cases missing data, noise, corrupted inputs with >85% recall target.
3. **Performance Testing**:
    - Measure latency: Process 1,000 data windows; target <1 second per window.
    - Test scalability: Run on 1 million data points to ensure no crashes.
4. **Robustness Testing**:
    - Test edge cases: Missing data (10% of points), noisy data (SNR=10), or corrupted inputs.
    - Verify model performance (recall >85%) under adverse conditions.

Evaluation Metrics

- **Primary Metrics**:
    - **Precision**: Percentage of flagged anomalies that are true (target: >90%).
    - **Recall**: Percentage of true anomalies detected (target: >95%).
    - **F1-Score**: Harmonic mean of precision and recall (target: >92%).
- **Secondary Metrics**:

    - **False Positive Rate**: Normal data flagged as anomalous target: <5%.
    - **Latency**: Processing time per window target: <1 second per 1,000 windows.
- **Threshold Tuning**:
    - Adjust Isolation Forest contamination (0.1 to 0.15) and LSTM reconstruction error threshold (0.05 to 0.1) to optimize metrics.
    - LSTM: Tune reconstruction error threshold to optimize F1-score.

Test Scenarios

Normal Operation:

- Input: Normal telemetry (e.g., stable temperature, voltage).
- Expected: No alerts; Isolation Forest scores <0.7; LSTM errors below threshold.

Known Anomaly:

- Input: Data with injected anomaly (e.g., 50°C temperature spike).
- Expected: Alert triggered with correct details (e.g., timestamp, sensor, score >0.7).

Edge Case:

- Input: Data with 10% missing values or added noise (SNR=10).
- Expected: Recall >85%; alerts remain accurate.

High Load:

- Input: Continuous 24-hour telemetry stream (e.g., 864,000 data points at 10Hz).
- Expected: System processes data in real time without crashes; latency <1 second.

Testing Process

1. **Setup**:
    - Use simulated NASA data SMAP, Curiosity, synthetic on AWS EC2, mimicking deployment environment.
2. **Execution**:

    - Run unit tests on preprocessing, models, and alerts.
    - Conduct integration tests on the full pipeline.
    - Perform performance and robustness tests with edge cases.
3. **Analysis**:
    - Calculate metrics precision, recall, F1-score, false positive rate for each scenario.
    - Log results in docs/test_results.md with tables and visualizations (e.g., ROC curves).
4. **Iteration**:
    - If metrics miss targets precision <90%, recall <95%
        1. Retraining: Adjust hyperparameters like Isolation Forest contamination to 0.15, LSTM learning rate to .0001.
        2. Fallback Model: Use Isolation Forest Alone if LSTM latency exceeds 1 second, leveraging its faster inference.
        3. Data Adjustment: Increase training data by 20% or augment with synthetic normal data to improve model robustness.
        4. Retest until targets are met.

Addressing Challenges

- **Data Quality**:
    - Test preprocessing robustness with noisy or incomplete data.
    - Use Isolation Forest as a fallback for low-quality data scenarios.
- **Real-Time Constraints**:
    - Optimize LSTM inference (e.g., model quantization) and leverage Isolation Forest's speed.
    - Measure latency in high-load tests to ensure <1 second processing.
- **Model Interpretability**:
    - Include visualizations (e.g., time-series plots with flagged anomalies) in alerts.
    - Provide confidence scores and sensor details to aid mission control.

Documentation

- Record results in docs/test_results.md:
  - Tables of metrics (precision, recall, F1-score) per scenario.
  - Visualizations (e.g., anomaly detection plots).
  - Notes on failures and mitigations.
- Update README.md: “See docs/testing_plan.md for the testing strategy and docs/test_results.md for results.”

Expected Outcomes

- System achieves >90% precision, >95% recall, and <5% false positives.
- Processes telemetry in real time (<1 second per 1,000 windows latency).
- Robust to edge case recall, maintaining >85% recall in noisy conditions.
- Provides clear, interpretable alerts for mission control.

References

- Hundman et al., “Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding,” KDD ’18: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018.