□ COMPREHENSIVE ANALYSIS BASED ON YOUR OUTPUT

1. DATA DESCRIPTION & PRE-PROCESSING

a) Datasets Used:

```
10 datasets with ~200M+ total samples:
- 0D-4D.csv: ~26M samples each (likely different operating conditions)
- 0E-4E.csv: ~6.9M samples each (likely different test scenarios)
```

Characteristics: Massive-scale vibration data from physical shaft system

b) Why We Chose Them:

- **Scale:** 200M+ samples provide excellent statistical power
- **Diversity:** Multiple files suggest different operating conditions
- **Real-world data:** Actual sensor measurements from physical system
- **Comprehensive coverage:** Various vibration scenarios

c) Data Cleaning:

Implicit cleaning through feature extraction:

```
% Statistical features are naturally robust:
features.rms = rms(vibration);        % Robust to outliers
features.crest_factor = peak/rms;     % Normalized measure
features.kurtosis = kurtosis(vibration); % Detects outlier distribution
```

No explicit cleaning - relies on statistical robustness

d) Train/Test Splits:

```
% PROBLEM IDENTIFIED: Only 8 training, 2 test samples!
□ Data split: 8 training, 2 test samples
```

Issue: Split is per **dataset** not per **data point** - this is wrong!

e) Data Augmentations:

None implemented - Using raw extracted features directly

2. SYSTEM/MODEL ARCHITECTURE

a) Model Used:

```
Regression Trees (fitrtree) for each parameter:
- Spring_model, Damper_model, Inertia_model
```

b) Model Type:

Regression Decision Trees - Interpretable, non-parametric

c) Number of Layers:

- **Single decision tree** per parameter
- **Automatically determined depth** during training

d) Important Hyperparameters:

```
'MinLeafSize': 1 (from optimization)
% This creates very deep trees that memorize data
```

e) Input Processing:

```
12-dimensional feature vector:
1. RMS vibration
2. Peak vibration
3. Crest Factor
4. Kurtosis
5. Skewness
6. Dominant Frequency
7. Low Frequency Energy
8. Medium Frequency Energy
9. High Frequency Energy
10. Mean RPM
11. RPM Variation
12. Mean Voltage
```

f) Why This Architecture is Suitable:

- **Interpretable** - Engineers understand decision paths
- **Handles non-linear relationships** in vibration data
- **No feature scaling required** - Robust to different units
- **Fast inference** - Critical for real-time calibration

3. EXPERIMENTAL SETUP

a) Batch Size: N/A (Decision trees process all data at once)

b) Learning Rate: N/A (No gradient descent - uses CART algorithm)

c) Optimizer: CART (Classification and Regression Trees) algorithm

d) Epochs: Single training pass (no iterative training)

e) Loss Function: Mean Squared Error (default for regression trees)

f) Metrics:

```
- RMSE (Root Mean Square Error)
- MAE (Mean Absolute Error)
- R² (Coefficient of Determination)
- Cross-validation RMSE
```

g) Training Method:

```
% Recursive binary partitioning
fitrtree(features, targets, 'MinLeafSize', 1, 'CrossVal', 'on')
```

h) Early Stopping: Via `MinLeafSize` parameter

i) Validation Split:

```
80/20 split (PROBLEM: Only 10 datasets → 8 train, 2 test)
5-fold cross-validation
```

j) Checkpoints: Model saving after training completion

4. EVALUATION METRICS - ANALYSIS OF YOUR RESULTS

Your Output Shows Critical Issues:

```
Spring - Test RMSE: 0.00, CV RMSE: 0.00, R²: NaN, MAE: 0.00
Damper - Test RMSE: 0.00, CV RMSE: 0.00, R²: -Inf, MAE: 0.00
Inertia - Test RMSE: 0.019, CV RMSE: 0.012, R²: -0.562, MAE: 0.015
```

What These Metrics Mean:

- **RMSE = 0.00:** Perfect prediction (overfitting - memorized data)
- **R² = NaN/ -Inf:** Mathematical error - zero variance in predictions
- **MAE = 0.00:** Zero error (impossible in real world)

Proper Evaluation Metrics Should Show:

- **RMSE > 0:** Some prediction error (realistic)
- **R² between 0-1:** Model explains variance in data
- **Reasonable MAE:** Average error magnitude

CONCLUSION

Your current approach has a fundamental flaw: You're treating each entire dataset (26M samples) as a single data point, when you should be extracting features from smaller time windows within each dataset.

The perfect metrics (RMSE=0.00) indicate severe overfitting because each "dataset" only produces one feature vector, and with `MinLeafSize=1`, the trees memorize these 10 data points perfectly.

Next step: Implement sample-level feature extraction to create thousands of training examples from your 200M+ samples!