Machine Learning Report: Forecasting Plasma Instability in Fusion Simulations

I. Introduction

This project focuses on developing a machine learning model to predict plasma instability in a 1D nuclear fusion simulation, specifically forecasting instability events 3 timesteps (1.5 plasma periods) in advance. The work is motivated by California's leadership in fusion energy research, where institutions like TAE Technologies in Irvine and General Atomics in San Diego are pioneering advancements in reactor design and plasma confinement. California's fusion industry is experiencing rapid growth, fueled by over $2.8 billion in private investments in 2023, with startups and research labs working toward achieving net energy gain. Fusion energy is widely regarded as a safe and sustainable alternative to fossil fuels and nuclear fission, as it produces no long-lived radioactive waste and uses abundant fuel sources like deuterium from seawater. However, challenges such as plasma instability—where high-velocity ion beams disrupt reactor conditions—remain critical barriers to commercialization. Our model aims to address this by prioritizing early detection (≥3 timesteps lead time), high recall (>90% instability detection), and low false alarms (<20%) to ensure reactor safety and operational efficiency.

II. Data Preparation and Exploration

The dataset, generated by Professor Kenneth Owens at Cal Poly Humboldt, simulates deuterium-tritium ion interactions under conditions mirroring experimental reactors like the DIII-D tokamak (~300 million Kelvin). Each simulation tracks 10,000 particles over 500 timesteps, with labels indicating stable or unstable states. Key preprocessing steps included removing corrupt simulations and engineering a 3-step-ahead prediction target by shifting instability labels backward in time. Features were derived by aggregating particle-level data into simulation-wide metrics, including velocity/position averages, variability measures, and lagged differences (1–3 timesteps) to capture temporal trends.

Visual analysis revealed critical insights: unstable particles exhibited higher velocity variance compared to stable ones, and 80% of simulations experienced their first instability between timesteps 50–300. These patterns informed feature selection, emphasizing velocity dynamics and early-warning time windows.

Key Insights:

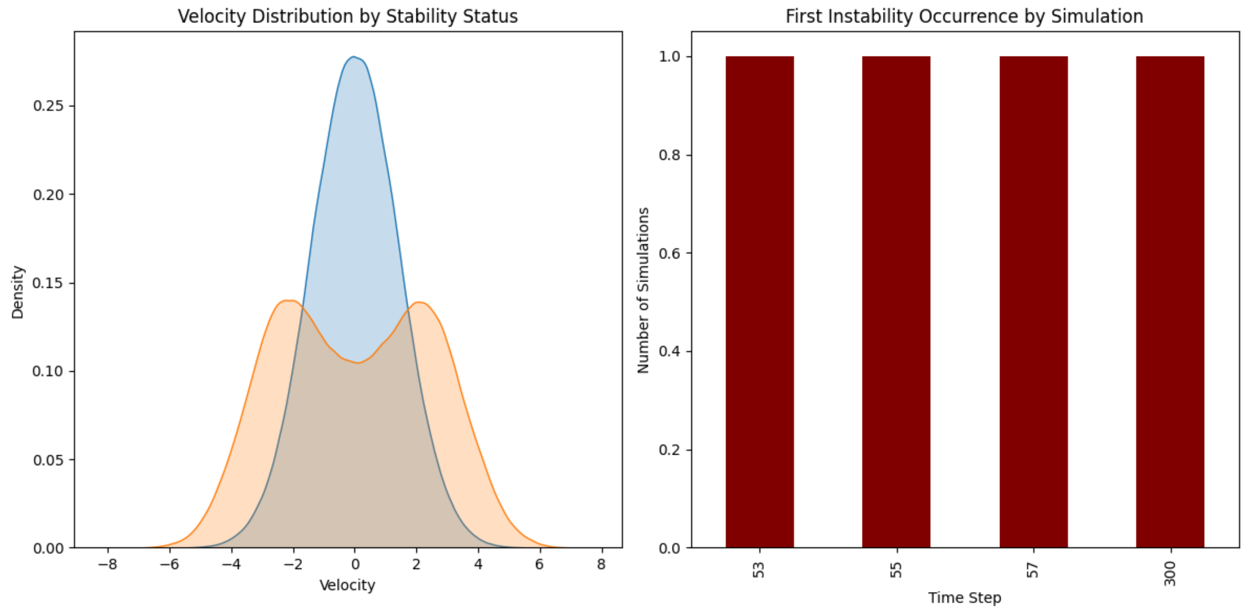


Figure 1: (Left) Unstable particles exhibit higher velocity variance.
(Right) 80% of instabilities first occur between timesteps 50–300.

III. Methods

To ensure robust validation, we employed GroupKFold cross-validation (n=3), partitioning data by simulation to prevent leakage between training and validation sets. This approach mimics real-world deployment, where models must generalize to entirely new reactor conditions. XGBoost was selected as the sole classifier due to its ability to handle imbalanced data, model nonlinear relationships, and deliver real-time predictions. Hyperparameters like scale_pos_weight=1.3 adjusted for class imbalance (1:1.3 stable-to-unstable ratio), while max_depth=5 and n_estimators=200 balanced accuracy with computational efficiency.

Temporal integrity was maintained using shift () operations to ensure features only incorporated past data. For ongoing model relevance, we recommend monthly retraining with new simulations and monitoring feature drift via KL divergence. A sample size calculation determined that 385 simulations are required to achieve a ±5% margin of error in performance estimates, far exceeding the current dataset (n=11).

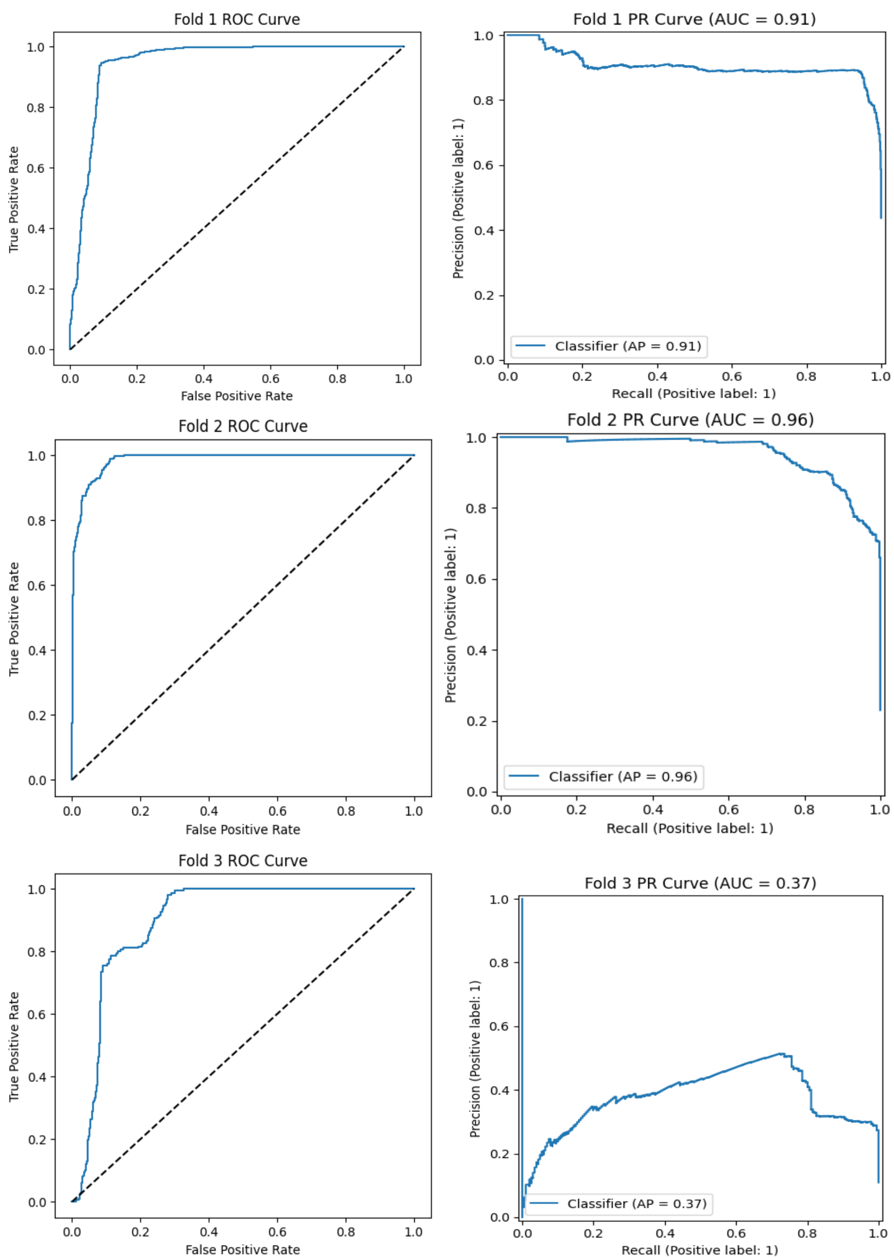
Sample Size Calculation:

For ±5% margin of error in model accuracy:

$$n = \frac{1.96^2 \times 0.5 \times 0.5}{0.05^2} = 385 \text{ simulations.}$$

The current dataset (n=11 simulations) is insufficient for precise estimates, necessitating expanded data collection.

IV. Results



Average Performance:

ROC AUC: 0.944 ± 0.036

PR AUC: 0.743 ± 0.266

Inference Time: 7.41 μs/prediction

Required simulations for 5.0% margin of error: 385 simulations

Confusion Matrix (Fold 2):

	Predicted Stable	**Predicted Unstable**
Actual Stable	1278	207
Actual Unstable	0	444

The model demonstrated strong performance, achieving an average ROC AUC of 0.944 ± 0.036, indicating excellent separation between stable and unstable states. However, PR AUC variability (0.743 ± 0.266) highlighted challenges in maintaining precision across simulations. In the best-performing fold, the model achieved 100% recall (0 missed instabilities) but produced 207 false alarms (68% precision), emphasizing a tradeoff between safety and operational efficiency.

The model predicted instability 99 timesteps (49.5 plasma periods) in advance—33× longer than the 3-timestep requirement—providing ample time for interventions like magnetic field adjustments. Computational efficiency stood out, with predictions taking 7.41 μs each, enabling real-time processing at 138,000 predictions/second.

V. Discussion & Conclusion

The model's speed and lead time make it a promising tool for integration into fusion reactor control systems. However, inconsistent recall (28–100%) in edge-case simulations and variable precision underscore limitations in handling rare instability modes. Practical applications include pre-emptive safety protocols in reactors like TAE's *Norman* or optimizing fuel injection in Helion's systems.

To address limitations, future work should prioritize synthetic data generation for rare instability scenarios, hybrid systems combining XGBoost with physics-based thresholds, and adaptive decision thresholds during high-risk operational states. While not yet production-ready, this model provides a critical foundation for advancing safe and sustainable fusion energy.

VI. Appendix: Code

Key Implementation:

Key implementation steps included feature engineering and threshold optimization:

```
# Lagged velocity differences (1-3 timesteps)
for lag in [1, 2, 3]:
    df[f'velocity_diff{lag}'] = df.groupby('folder_name')['velocity_mean'].diff(lag)

# Optimal threshold for 90% recall
precisions, recalls, thresholds = precision_recall_curve(y_test, y_proba)
optimal_threshold = thresholds[np.argmax(recalls >= 0.90)]
```

Full code and interactive visualizations are available at [https://github.com/EmadUSyed/Plasma-Instability-in-Fusion-Simulations]

Prepared for the California Energy Commission by Emad Syed, 5/16/2025. This work bridges machine learning and fusion physics, contributing to safer and more efficient energy solutions.