A SIMPLE MODEL AND ADVANCED STRATEGIES ARE ALL YOU NEED

Isaac Haik, Francois Porcher
isaac.haik@gmail.com, GitHub

1 Introduction

Representing **Team QR_IS**, we achieved fourth place in Phase I of the competition. Despite not topping the leader-board, our model demonstrated the best balance in **performance**, **extrapolation ability**, and **computational efficiency**, as shown in Table 1. Our results underscore our model's robustness, simplicity, and effective resource management, making it ideally suited for practical deployment in space traffic management.

Table 1: Results of the competition

Rank	Participant team	F2 Total	F2 Partial	Difference
1	Millennial-IUP	0,811	0,917	-0,11
2	Hawaii2024	0,805	0,952	-0,15
3	MiseryModel	0,801	0,944	-0,14
4	QR_Is	0,795	0,852	**-0,06**
5	FuturifAI	0,781	0,822	**-0,04**
6	Go4Aero	0,773	0,916	-0,14
7	K-PAX	0,772	0,935	-0,16
8	Colt	0,758	0,949	-0,19
9	Yzzzz	0,731	0,864	-0,13
10	Astrokinetix	0,709	0,829	-0,12

2 Feature Engineering and Selection

A feature is a transformation of the raw data that is fed as input to a model. In this section, we detail our highest quality features with important predictive power. You can find the exhaustive list of features on our GitHub.

2.1 Trajectory-Based Features (257 features)

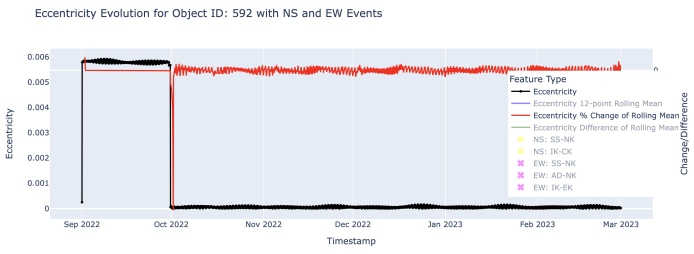


Figure 1: Eccentricity change

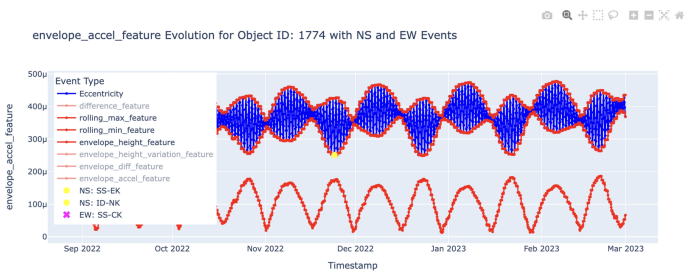


Figure 2: Envelope diff

Our analysis uses 257 features from trajectory analysis, derived from various time-series transformations to detect changes and patterns. These include Lag *(180 features)*, Difference *(36 features)*, Percentage Change *(6 features)*, Rolling Average *(10 features)*, and Rolling Standard Deviation *(10 features)*. For example, Figure 1 shows the percentage change in eccentricity, highlighting its activity near a node.

Furthermore, the feature termed "envelope", which represents the rolling minimum and maximum of a time-series, demonstrates significant predictive power. More specifically, fluctuations in the distance between the maximum and minimum values of the envelope often coincide with the occurrence of a node, as illustrated in Figure 2, where the feature becomes active at the node.

2.2 Satellite Characterized Features (16 features)

In this section, we explore features that consistently characterize satellites. Notably, we observe that satellites with high variance in their features often have a greater number of nodes. This relationship is demonstrated in Figure 3, which shows a correlation between the total standard deviation of eccentricity and the number of nodes in the East-West (EW) direction. This finding is particularly useful for our predictive model, as it can leverage knowledge of high volatility at any time step to trigger a prediction more effectively.

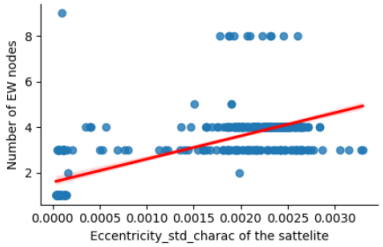


Figure 3: Scatter plot for 500 satellites (each point represents one satellite)

The specific features derived include average values across the trajectory *(9 features)* and standard deviation *(7 features)*

2.3 Time-Based Features (5 features)

We developed five time-based features that capture cyclical patterns to reflect temporal progress in our model. For example, one feature cycles through values from 1 to 12 over 12 timestamps, each representing two hours, equating to one day. These features, including cycles of 12, 24 (one day), 72 (three days), 168 (one week), and 336 hours (two weeks), allow the model to recognize time-related patterns, such as node occurrences more likely towards the end of a day. This concept is akin to positional encoding in transformers [1], which tracks sequence positions.

2.4 Insights on Frequency Analysis

During the initial phase of the challenge, we hypothesized that the periodic nature and noticeable seasonality in the data suggested that frequency patterns possess significant predictive power.

As demonstrated in Figure 5, there is a noticeable change in frequency within the Y (m) data around nodes. Specifically, the envelope exhibits high frequency fluctuations before a node and slows down afterwards. This is confirmed by a Fast Fourier Transform (FFT) analyses of the time series data. After the node, the FFT reveals that the magnitudes increased in the zone of longer periods, suggesting a slower modulation of the envelope [2].

To quantify these observations, we project the time series data from a three-day rolling window onto a Fourier basis and compute the $L2$ distance between successive vectors. This approach has proven effective in capturing the peak frequency changes during events, as shown in Figure 4.

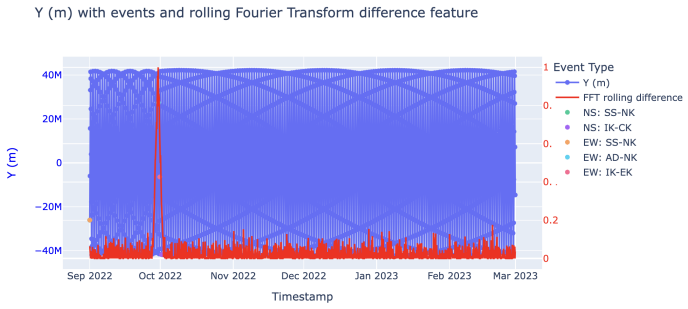


Figure 4: Change in envelope frequency before and after the node

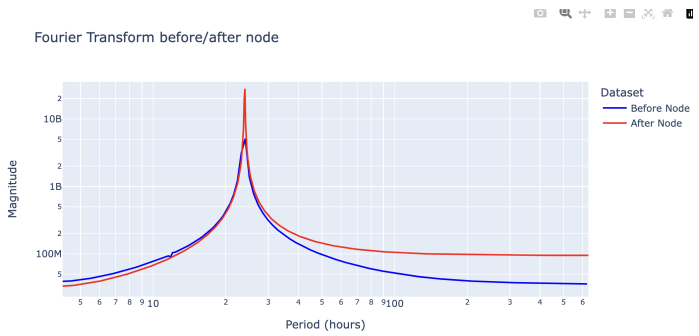


Figure 5: FFT Transform before and after node

3 Methodology

3.1 Problem formulation

Each timestamp records the *node* for North-South (NS) and East-West (EW) directions. The goal is to identify *nodes* within a 6 time-steps tolerance, treating it as **two separate multi-class classification tasks**, one for each direction.

Table 2: Node Classifications for EW and NS Directions

Direction	Classes
NS	Nothing, SS-NK, SS-CK, SS-HK, SS-EK, ID-NK, IK-CK, IK-HK, IK-EK
EW	Nothing, SS-NK, SS-CK, SS-HK, SS-EK, ID-NK, IK-CK, IK-HK, IK-EK, AD-NK

Here, 'Nothing' signifies "no node." Although this could be treated as a single multi-class classification task, the possibility of simultaneous events in both directions necessitates handling them as separate problems. This separation is essential as a single classifier per timestamp might overlook a concurrent event. Further justification is provided in the Model Separation and Recombination Strategy.

3.2 Model choice

We selected **gradient-boosted trees** using the **CatBoost** Python package as our primary model. This decision was influenced by the small size of our training dataset, which limits the effectiveness of data-intensive architectures like deep neural networks or transformers [3] . Given the dataset constraints, our choices narrowed to gradient-boosted trees and shallow neural networks.

Feature	Gradient-Boosted Trees	Neural Networks
Complexity	Easy, only a few hyperparameters to tune.	High. Requires architecture choices and tuning.
Training	Fast training, low ressources, less prone to overfitting.	Slow training, requires expensive GPU, more prone to over-fitting, training less stable.
Interpretability	High, with clear decision paths.	Low, considered a "black box."
Noise Handling	Robust against data noise and outliers.	Sensitive; often requires additional steps.

Table 3: Comparison of Gradient-Boosted Trees and Neural Networks in Multi-class Classification

Considering that gradient-boosted trees are faster to train, less prone to over-fitting, resources efficient, robust against outliers, and explainable, and that they had tremendous success in the competitive data science community, we chose them [4, 5].

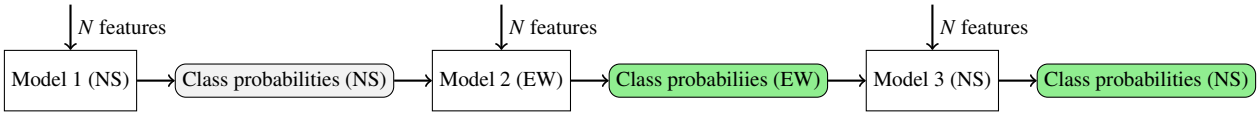
3.3 Model Separation and Recombination strategy

This section evaluates the decision to use separate models for NS and EW directions, and describes our **recombination strategy** to share information between models. Here is the trade-off between using a simple model vs two independent models for each direction:

	Single Model	Two Independent Models
Pros	• Accounts for NS-EW interaction • Simpler development and monitoring	• More control over precision and recall per direction • Can predict two events simultaneously
Cons	• Reduced control over individual direction metrics • Difficulty managing 16 classes simultaneously	• Assumes independence of EW and NS • Longer development due to individual tuning

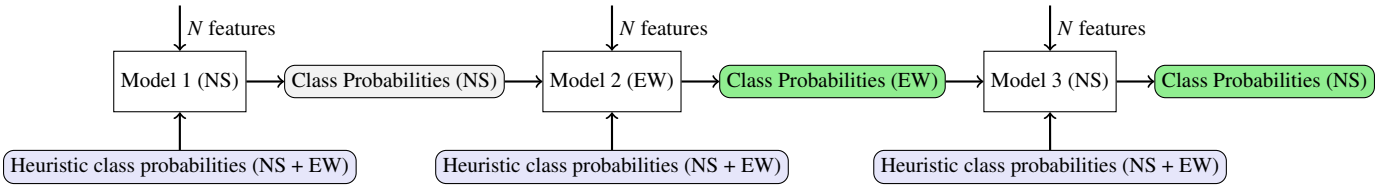
Table 4: Pros and cons of model separation strategies

Recombination Strategy: Our approach integrates independent predictions with interactions between NS and EW through a recombination mechanism. **Model 1** uses N input features for initial NS predictions (then discarded). **Model 2** then adds these class probabilities to N features to determine EW. Finally, **Model 3** uses Model 2's class probabilities and N features to refine NS predictions.



3.4 Model Stacking

Model stacking enhances prediction accuracy by using multiple models sequentially. The heuristic model from challenge organizers tracks changes in satellite longitude and inclination, providing station-keeping and propulsion insights. This method is somewhat orthogonal to our tree-based models, adding diversity. We integrated this heuristic model into our pipeline, using its predictions as additional inputs for the tree-based model, as shown in the diagram below:



3.5 Model Prediction Strategy

This section describes our strategy to derive classes from model-generated probabilities. Notably, the F_2 score prioritizes Recall, weighting it four times more than Precision. This insight inspired a **dual-threshold mechanism** introducing two **degrees of freedom** to better manage precision and recall trade-offs in multi-class classification, vital for handling multiple thresholds in unbalanced datasets.

- **Highest Probability and Class** ($p_{\mathbf{max}}, c_{\mathbf{max}}$): Represents the model's top prediction. If $p_{\max} \leq \tau_{\text{precision}}$, the prediction is set to "no event", ensuring that classes other than "no event" are sufficiently confident, thereby increasing precision.
- **Second Highest Probability and Class** ($p_{\mathbf{second\ max}}, c_{\mathbf{second\ max}}$): Represents the second most probable class. If $c_{\max}$ is "no event" and $p_{\text{second max}}$ exceeds the **recall threshold** ($\tau_{\mathbf{recall}}$), the model opts for $c_{\text{second max}}$, thus prioritizing recall. This strategy is particularly useful in unbalanced datasets to avoid bias toward the most frequent class.

Parameter values are documented for reproducibility. Notably, the τ_{recall} for NS is higher than for EW to address the greater class count in EW (10) versus NS (9). This higher threshold helps improve recall, as more classes typically reduce the model's accuracy per class.

Direction	$\tau_{\text{precision}}$	τ_{recall}
EW	0.1	0.11
NS	0.1	0.26

Table 5: Sensitivity Thresholds

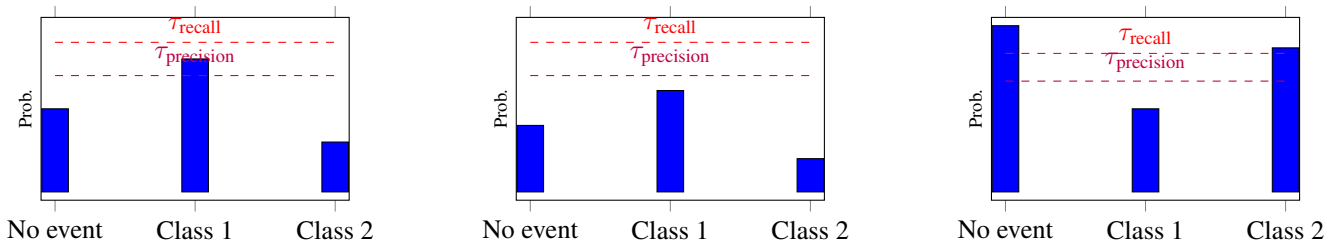


Figure 6: Illustration of double-thresholding mechanism

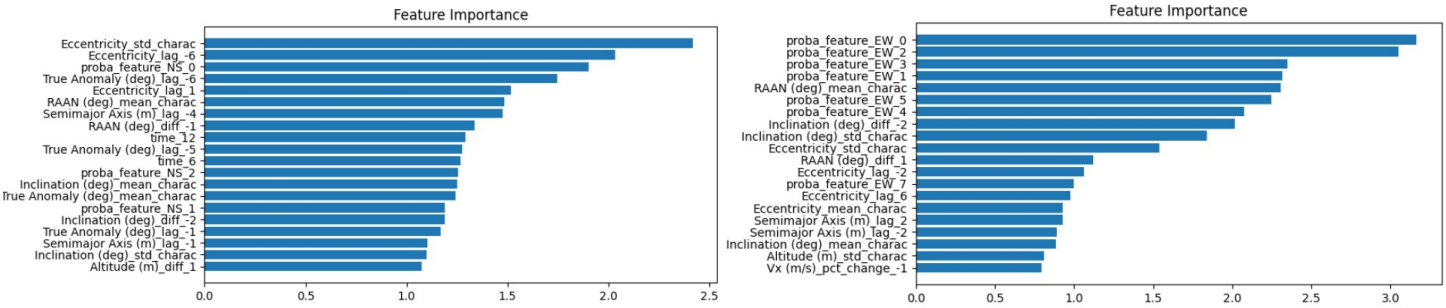
In figure (a), $Class\ 1$, exceeding $\tau_{\text{precision}}$, leads to a prediction of $Class\ 1$. In figure (b), though $Class\ 1$ is most probable, it falls below $\tau_{\text{precision}}$, prompting a *No event* prediction. In figure (c), despite *No event* being most probable, $Class\ 2$ surpasses τ_{recall}, resulting in a $Class\ 2$ prediction.

In summary, the **dual-threshold** mechanism allows us to effectively and easily control precision and recall in multi-class classification, significantly increasing the performance.

4 Core Observations and Reproducibility

4.1 Features Importance

The feature importance plots below validate our modeling choices by highlighting the predictive power of specific features, leveraging the high interpretability of decision trees [6].



(a) Model 2 (prediction of EW node)

(b) Model 3 (prediction of NS node)

Figure 7: Top 20 feature importance for each model

The plots confirm that features related to **Eccentricity** and the outputs from previous models (**proba_features**) are critically important. This supports our strategies of **model stacking** and **model recombination**. Additionally, the significance of "Satellite Characterized Features" (see Section 2.2) highlights the benefits of pre-informing the model about the satellite types it is analyzing.

4.2 Training Details, Hardware and Software Environment

Initially, we trained using 500 trajectories and validated with 1,400. For the final model training, all 1,900 trajectories were utilized. Minimal hyper-parameter tuning was implemented, with the primary adjustment being an increase in *n_estimators* from 100 in experimental testing to 1,000 for final training to ensure model robustness without overfitting, which was confirmed by the final evaluation.

Hardware	Type
CPU	Intel Xeon E5-2699 v4 (22 cores / 44 threads)
RAM	256 GB
Storage	512 GB SSD NVMe in PCI Express + 1 TB HDD

Model	Nb of features	Training Time
NS Model 1	321	1 hour
EW Model 2	321 + 9	3 hours
NS Model 3	321 + 10	1 hour

Training was done on a CPU, demonstrating that deep learning's higher computational demands and cost are not always necessary. At equal computational resources, our model leads to superior performance. The full code and guidelines for replication are available here.

5 Conclusion

This paper introduces novel methods enhancing AI performance in trajectory tracking. Our analysis underscores the importance of satellite-specific features and envelope variations for prediction. Using **frequency analysis**, we established that seasonal and cyclical changes are key predictors. Techniques like **Model Recombination** and **Model Stacking** have refined our predictions, while a **dual-threshold mechanism** optimized the precision-recall trade-off. Although not leading the leaderboard, our model's simplicity, computational efficiency, and robustness make it suitable for real-world applications. For future enhancements, we recommend a deeper investigation in spectral analysis. With **wavelet transforms**, we anticipate more nuanced detections of cyclical patterns, potentially leading to significant performance gains.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[2] John G. Proakis and Dimitris K. Manolakis. *Digital Signal Processing: Principles, Algorithms, and Applications*. Pearson, 4 edition, 2006.

[3] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention, 2021.

[4] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[5] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[6] Andrei V. Konstantinov and Lev V. Utkin. Interpretable machine learning with an ensemble of gradient boosting machines. *arXiv preprint arXiv:2101.03334*, 2021.