Analysis of ECONet: Automating Weather Station Quality Assurance (CSC 522, Group 12)

Manu Jayachandran
NC State University

Bedant Lohani
NC State University

Mukul Sauhta
NC State University

1 Introduction and Background

1.1 Problem Statement

The North Carolina State Climate Office manages the ECONet project, which uses a network of more than 50 weather stations to collect environmental data every minute across the state. This data supports both operational forecasting and scientific research, so it is important that it be accurate and usable. To help maintain data quality, an automated quality-assurance (QA) system flags suspicious measurements, which are then manually reviewed by experts to decide whether they are true sensor errors or false positives. Because the network produces such a large volume of data, this review process has become a bottleneck.

The goal of this project is to build a machine learning pipeline that can classify flagged measurements as true hardware errors or false positives. A useful model would reduce manual review effort, help experts focus on the most important cases, and improve the operational efficiency of the current QA workflow without replacing the existing rule-based system entirely.

1.2 Related Work

ECONet collects critical environmental observations across North Carolina, and Saia et al. [3] explain how the network functions as a regional mesonet, the kinds of measurements it collects, its maintenance and siting procedures, and the operational importance of the data. That paper gives the necessary domain context for understanding why weather-station QA matters and why station metadata and sensor context may be useful predictors.

In addition, Heuser et al. [2] describe the rule-based quality-control process already used in ECONet, including the range, buddy, intersensor, and trend checks that generate the QA flags. This is directly relevant to our problem because the labeled prediction task begins after this automated flagging stage. Understanding those routines helped us interpret the QA flag columns and design features around how suspicious readings are currently identified.

Finally, Das and Parthasarathy [1] highlight the importance of spatial and temporal structure in anomaly detection for climate-related data. That perspective is important here because a flagged weather-station measurement is often not best understood as an isolated row. Instead, whether a measurement is truly erroneous may depend on recent station behavior, seasonal timing, and local sensor patterns over time. This motivates the temporal and contextual feature engineering used in our project.

2 Method

2.1 Approach

Our first step was to preprocess the labeled data so that it would be ready for training. The target label was converted into a numeric binary form for modeling. Columns intended to be categorical, such as station and measurement type, were treated as categorical features. Timestamps were parsed into proper datetime values, duplicates were removed where appropriate, and the dataset was cleaned into consistent datatypes. Missing values introduced during feature construction were handled through a controlled pipeline, and earlier cleaning work used short-window interpolation only where it was reasonable to avoid inventing unrealistic patterns.

After preparing the labeled data, we performed feature engineering so that each reviewed flagged reading could be represented in a tabular form suitable for standard classifiers. For every reviewed measurement, we retained the current reading attributes, including the observed value, station identifier, measurement type, and automated QA flags. We then linked each reviewed row to surrounding station history from the `full/*.csv` files and computed lag-based, change-based, rolling-summary, and deviation features that describe recent local sensor behavior. This produced three progressively richer feature settings: a current-row representation, an engineered base representation, and a full base-plus-context representation.

Our project includes two explicit novelty contributions beyond the guided projects. First, we augment reviewed ECONet QA examples with context from the full station time-series files by linking each labeled flagged measurement to its surrounding temporal history and engineering lag, trend, rolling-summary, and deviation-based features. This lets us move beyond a single-row tabular formulation and test whether historical station behavior improves classification. Second, we use domain-informed feature engineering tailored to operational weather-station quality assurance, including QA-summary features, cyclic calendar encodings, station and location metadata, and measures of recent volatility or deviation from local norms. Together, these additions make the project more than a standard classifier on the provided labels and directly address the CSC 522 novelty requirement.

For model training, we compared a majority-class baseline with Logistic Regression, Decision Tree, Random Forest, and HistGradientBoosting. For the largest base-plus-context experiment, we used a Spark-based implementation to keep training feasible on the full dataset while preserving the same modeling logic and evaluation protocol.

2.2 Rationale

We based our modeling choices on both the structure of the ECONet task and the properties of the dataset. The prediction problem is a binary classification task in which only a small proportion of reviewed flagged measurements are true errors. In such a setting, plain accuracy would be misleading because a model could look strong simply by predicting the majority class almost all the time. For that reason, we chose F1 as the primary metric and reported precision, recall, PR-AUC, and ROC-AUC as supporting metrics.

F1 is especially appropriate here because the practical goal is to identify true errors while balancing the cost of false alarms and missed errors.

Our model choices were also motivated by the need to compare interpretability against expressive power. The DummyClassifier serves as a minimum bar, showing whether learned models outperform a trivial majority-class strategy. Logistic Regression provides a simple, interpretable linear baseline and helps determine whether the relationship between the features and the target can be captured with a linear decision boundary. Decision Tree adds a transparent non-linear baseline that can still be read as a threshold-based QA rule system. Random Forest and HistGradientBoosting were chosen because the ECONet task likely contains non-linear interactions among QA flags, sensor values, station identity, seasonal timing, and temporal context. This mix of models allows us to address the question of how close an interpretable model can come to a stronger ensemble approach.

Our feature-engineering design was motivated by the QA setting itself. A flagged weather-station reading often cannot be judged correctly from a single row alone. Many true errors look suspicious only when compared to recent station behavior, recent volatility, or sudden shifts in the signal. For that reason, we engineered temporal context features such as recent lags, rolling summaries, and deviation-from-history measures using the station time-series files. These features are intended to capture abrupt jumps, unstable short-term behavior, and departures from local norms that may not be visible from the flagged reading alone.

Finally, our hyperparameter choices were constrained by the size of the dataset and the need to keep the experiments computationally feasible. We used a staged tuning strategy on a fixed validation protocol inside the training data. For simpler models, the main concerns were regularization and class weighting. For tree-based models, the main concerns were complexity control, pruning, and leaf size. Rather than performing an exhaustive search over all possible settings, we focused on the hyperparameters most likely to affect generalization and class-imbalance behavior.

3 Plan & Experiment

3.1 Dataset

The ECONet QA dataset contains reviewed weather-station measurements collected across North Carolina. For model development, we used the provided labeled training data, where each row corresponds to one reviewed flagged measurement. The main columns include `Station` (station identifier), `Ob` (timestamp), `measure` (sensor or measurement type), `value` (recorded sensor reading), `target` (0 for likely accurate, 1 for likely erroneous), and the automated QA flags `R_flag`, `I_flag`, `Z_flag`, and `B_flag`. We also used an engineered feature table derived from the same training rows, which added location metadata, cyclic calendar encodings, QA-summary features, and temporal context variables from the full station time-series.

The cleaned labeled dataset contained 6,593,274 reviewed rows. The class distribution was highly imbalanced, with 6,358,102 negatives (96.43%) and 235,172 positives (3.57%). This severe imbalance motivated our choice to optimize F1 rather than accuracy. In the final report, we evaluate three feature settings: (1) a cleaned current-row feature representation built directly from the labeled training data, (2) a richer engineered base feature set supplied by our feature-engineering pipeline, and (3) a full base-plus-context feature set that adds lag, rolling-window, and deviation features from the station time-series. The final context-rich feature table contained 30 base engineered features and 44 additional context features, for a total of 74 modeled predictors excluding passthrough columns such as identifiers, timestamps, and the target.

The provided `test.csv` did not include target labels. As a result, it functioned as a blind holdout set: the file could be used only for generating final predictions, not for computing local performance metrics. This is consistent with our original project design of keeping development, threshold tuning, and model comparison entirely within the provided training data.

3.2 Hypotheses

We investigated three main hypotheses. First, we hypothesized that ensemble tree methods would outperform simpler interpretable baselines, since the relationship between QA flags, recorded measurements, station identity, and temporal behavior is likely non-linear. Second, we hypothesized that richer engineered non-context features would improve performance over using only the current reviewed row. Third, we hypothesized that adding full temporal context would be most helpful for detecting true errors, especially in terms of recall and PR-AUC, even if it did not necessarily maximize F1 after thresholding.

3.3 Experimental Design

We designed our experiments to answer three main questions: which classifier performs best on the ECONet QA task, whether better engineered features improve performance, and whether temporal context adds value beyond the current observation and metadata. Because the same station and measurement type appear repeatedly over time, we avoided a random split and instead used a time-aware grouped split inside the training data. Rows were grouped by station and measurement type, sorted by timestamp, and later observations were reserved for later-stage evaluation. This reduced leakage from nearby repeated measurements and better mimicked the real use case of predicting on future flagged observations from known stations.

Our workflow used three splits derived from the cleaned training data: a fit split of 4,219,352 rows, a tuning split of 1,055,059 rows, and a final validation split of 1,318,837 rows. The validation set contained 33,240 positive examples, preserving the strong class imbalance of the original problem. Hyperparameters and classification thresholds were selected using only the fit and tuning portions, while the validation split was used once for final comparison. We intentionally kept the main evaluation protocol fixed across experiments so that differences in model performance would be attributable to model and feature choices rather than changing data partitions.

We compared five models: a majority-class `DummyClassifier` baseline, Logistic Regression, Decision Tree, Random Forest, and HistGradientBoosting. Logistic Regression was included as a simple interpretable baseline; Decision Tree as a transparent non-linear

model; and Random Forest and HistGradientBoosting as stronger ensemble methods for tabular data with non-linear interactions. In line with our proposal, F1 was the primary model-selection metric, while precision, recall, PR-AUC, and ROC-AUC were reported as supporting metrics. We also tuned the decision threshold for each model on the tuning split rather than fixing it at 0.5, since the positive class is rare and the precision-recall tradeoff is important for this application.

We completed three experiment settings. The first used cleaned current-row features derived directly from the reviewed labeled data, including value, measurement type, station, QA flags, and simple calendar or QA-summary features. The second used a richer engineered base feature set that added cyclic time encodings, additional QA summaries, and station or location metadata. The third used the full base-plus-context feature table, which added lag, rolling, and deviation features from the station time-series. For the largest context-rich setting, we used a Spark-based pipeline and saved per-model artifacts, prediction outputs, summary tables, and interpretability figures. Because of the scale of the dataset and the runtime required even on distributed infrastructure, we prioritized completing the main planned experiments and the blind holdout submission before exploring additional optional extensions such as neural-network baselines or broader ablation studies.

After selecting the final context-rich model on the validation protocol, we also generated blind predictions on the unlabeled `test.csv` file. The final blind-submission model was the base-plus-context Random Forest with `numTrees = 100`, `maxDepth = 20`, `minInstancesPerNode = 4`, `featureSubsetStrategy = sqrt`, and no class balancing. Because the holdout labels were hidden, we do not report test-set accuracy, F1, or PR-AUC in this paper.

4 Results

4.1 Results

Table 1: Validation results across the three completed experiment settings.

Setting	Model	Threshold	F1	Precision	Recall	PR-AUC
Current-row	Random Forest	0.30	0.861	0.918	0.811	0.905
Current-row	HistGradientBoosting	0.25	0.838	0.984	0.730	0.898
Current-row	Logistic Regression	0.25	0.746	0.801	0.698	0.736
Current-row	Decision Tree	0.70	0.736	0.749	0.724	0.556
Current-row	Dummy	0.50	0.000	0.000	0.000	0.025
Engineered base	HistGradientBoosting	0.10	0.869	0.912	0.831	0.913
Engineered base	Random Forest	0.25	0.811	0.793	0.830	0.907
Engineered base	Logistic Regression	0.35	0.769	0.842	0.708	0.770
Engineered base	Decision Tree	0.70	0.755	0.761	0.749	0.591
Engineered base	Dummy	0.50	0.000	0.000	0.000	0.025
Base + context	Random Forest	0.15	0.801	0.741	0.871	0.927
Base + context	HistGradientBoosting	0.15	0.769	0.752	0.786	0.848
Base + context	Logistic Regression	0.60	0.758	0.853	0.682	0.789
Base + context	Decision Tree	0.95	0.732	0.714	0.750	0.638
Base + context	Dummy	0.50	0.000	0.000	0.000	0.025

Table 1 summarizes the completed validation results across all three feature settings. In the first setting, which used cleaned current-row features only, Random Forest was the strongest model with an F1 score of 0.861, precision of 0.918, recall of 0.811, and PR-AUC of 0.905. HistGradientBoosting was competitive in this setting, achieving slightly lower F1 (0.838) but extremely high precision (0.984). Logistic Regression and Decision Tree were clearly weaker, while the Dummy baseline produced an F1 score of 0, confirming that a majority-class strategy is not useful for this task.

In the second setting, which used the richer engineered base feature set without the full temporal context block, HistGradientBoosting became the strongest model by F1. It achieved an F1 score of 0.869, precision of 0.912, recall of 0.831, PR-AUC of 0.913, and ROC-AUC of 0.994. Relative to the cleaned current-row run, this represents a modest but meaningful gain for the best boosting model, suggesting that stronger domain-informed non-context features already add value.

In the third setting, which used the full base-plus-context feature set, Random Forest was the strongest model. It achieved an F1 score of 0.801, precision of 0.741, recall of 0.871, PR-AUC of 0.927, and ROC-AUC of 0.996 at a threshold of 0.15. Although this did not exceed the best engineered-base F1 score, it produced the best PR-AUC and the highest recall among the strongest completed models. This indicates that the full temporal context features improved the model's ranking quality and sensitivity to true errors, even if the best thresholded F1 remained lower than the engineered-base HistGradientBoosting result.

These experiments answer our research questions in a more nuanced way than we initially expected. For RQ1, the strongest model by F1 is HistGradientBoosting on the engineered base feature set, while the strongest model by PR-AUC and recall is Random Forest on the full base-plus-context feature set. For RQ3, the interpretable Logistic Regression model remains meaningfully behind the best ensemble model in every setting, suggesting that the relationship between QA flags, recorded measurement values, site metadata, and temporal context is not well captured by a simple linear boundary.

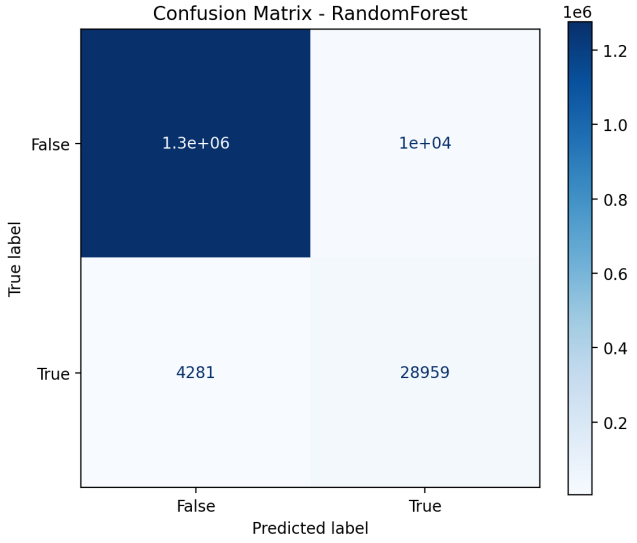

Figure 1: Confusion matrix for the best base-plus-context model (Random Forest).

Figure 1 shows the confusion matrix for the best base-plus-context model, Random Forest. At the selected threshold of 0.15,

the model correctly identified 28,959 true errors and missed 4,281, while producing 10,107 false positives and 1,275,490 true negatives. This means the model captured a large share of the rare positive class, which is important for the intended QA workflow.

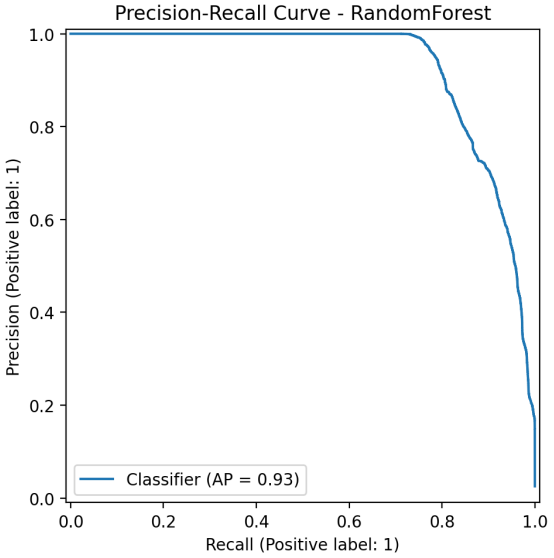

Figure 2: Precision-recall curve for the best base-plus-context model (Random Forest).

Figure 2 shows the precision-recall curve for that same Random Forest model. The average precision is approximately 0.93, which is strong given the severe class imbalance. The curve shows that the model maintains high precision over a substantial range of recall before precision drops more sharply near the far-right end. This supports the decision to emphasize PR-AUC in addition to F1 when evaluating context-rich models.

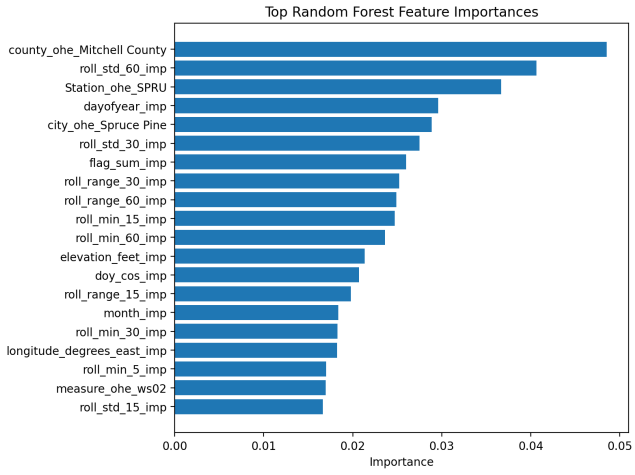

Figure 3: Top Random Forest feature importances from the base-plus-context experiment.

To better understand what the strongest context-rich model was using, we examined a feature-importance plot for the best base-plus-context model. Figure 3 shows the top Random Forest feature importances for the base-plus-context run. Several of the strongest predictors are station- or location-specific variables, such as Mitchell County and Spruce Pine, as well as rolling-window variability features like `roll_std_60`, `roll_std_30`, and `roll_range_30`. Seasonal timing (`dayofyear`) and QA-summary information (`flag_sum`) also remain important. This suggests that true-error detection depends not only on the current flagged reading, but also on recent local instability and persistent station-specific QA patterns.

In addition to the validation analysis, we generated blind predictions for the unlabeled `test.csv` holdout using the final Random Forest model. The output file contained predictions for 1,856,106 rows. Of these, 79,562 were predicted as positive, which is approximately 4.29% of the blind test set. This predicted-positive rate is close to, but slightly higher than, the 3.57% positive rate observed in the labeled training data, which is reasonable given that the model was tuned to recover rare true errors. Because hidden labels were not available to us locally, these outputs functioned as a final submission artifact rather than a directly scorable test-set result inside the report.

4.2 Discussion

The final results support our first hypothesis: ensemble tree methods are the strongest model family for this task. Across all completed experiments, either Random Forest or HistGradientBoosting outperformed the simpler baselines, and Logistic Regression never matched the best ensemble method. This is consistent with our expectation that the ECONet QA problem contains non-linear interactions among flags, sensor values, station identity, seasonal timing, and recent local behavior. This pattern is also consistent with the QA and anomaly-detection literature presented by Das and Parthasarathy [1], which suggests that sensor-quality decisions often depend on temporal structure and local context rather than only on isolated point measurements.

Our second hypothesis is also supported. Moving from the cleaned current-row representation to the engineered base feature set improved several models, especially HistGradientBoosting and Logistic Regression. This indicates that careful domain-informed feature engineering matters for this problem, even before full temporal context is introduced.

The third hypothesis is only partially supported. Adding full temporal context did not improve the best F1 score; the best F1 in the paper remains 0.869 from HistGradientBoosting on the engineered base feature set. However, the full base-plus-context Random Forest did achieve the best PR-AUC (0.927) and the highest recall (0.871), while also producing a very strong ROC-AUC. This suggests that the temporal context features improve the model's ability to rank and recover true errors, even if the final thresholded F1 is not maximized by that setting. In practical terms, the context-rich model may be more useful when the QA team prefers sensitivity and ranking quality over a single threshold-optimized metric to prioritize expert review, directly addressing the operational bottlenecks of the rule-based flagging system described by Heuser et al. [2].

The interpretation plot also helps explain the pattern of results. Station and location variables appear prominently in the Random Forest importance view, suggesting that some QA patterns are persistent at particular sites, reinforcing the importance of the regional mesonet siting procedures and metadata context outlined by Saia et al. [3]. Rolling-window variability and range features also become important once temporal context is available, which fits the domain intuition that many true errors appear as unstable local behavior rather than as isolated single-row anomalies. At the same time, the dominance of some site-specific variables suggests a limitation: because the model is trained and validated on known stations, part of its strength may come from learning persistent station-specific behavior rather than only universal sensor-error rules.

A second limitation is that the strongest results reported here are based on the fixed validation protocol derived from `train.csv`. While this protocol was designed carefully and kept constant across experiments, a separate held-out test evaluation with revealed labels would provide an even stronger estimate of final deployment performance. In addition, the computational cost of the largest feature-engineered runs was substantial even on distributed infrastructure, so we focused on fully completing and analyzing the main model comparisons rather than adding optional extensions such as neural-network baselines or larger ablation sweeps. Even so, the completed experiments are strong enough to answer the main project questions and demonstrate clear value from both ensemble methods and domain-informed feature engineering. The blind test prediction step at least confirms that the final selected model could be applied end-to-end to the unlabeled holdout set without changing the overall pipeline.

5 Conclusion

This project shows that machine learning can meaningfully support the existing ECONet QA workflow. Across millions of reviewed flagged measurements, learned classifiers substantially outperformed a majority-class baseline and provided a useful way to prioritize likely true errors for expert review. More broadly, the project demonstrates that combining operational QA flags with domain-informed feature engineering can produce strong performance even in a highly imbalanced real-world setting.

One of the most important lessons from the project is that better features do not necessarily improve every metric in the same way. The engineered base feature set produced the strongest F1 score, while the full base-plus-context feature set produced the best PR-AUC and recall. This is a useful reminder that in an operational QA application, the best model depends partly on what kind of mistakes matter most.

A second lesson is that temporal context is helpful, but not in the simplest possible way. Rather than uniformly improving all model scores, context-rich features seem to make the model more sensitive to true errors and better at ranking suspicious cases. That suggests that future work should not only ask whether context helps, but also how that context should be represented and which context features are worth the added computational cost.

Overall, the project suggests that a hybrid QA workflow is promising: rule-based checks can continue to flag suspicious observations, while a learned ranking or classification model can help prioritize which cases are most likely to be true errors. That broader lesson matters beyond ECONet itself, since many operational sensor networks face the same challenge of high data volume, rare failures, and limited expert review time.

References

[1] Mahashweta Das and Srinivasan Parthasarathy. 2009. Anomaly detection and spatio-temporal analysis of global climate system. In *Proceedings of the Third International Workshop on Knowledge Discovery from Sensor Data*. ACM, 142–150.

[2] Sean P Heuser, A P Sims, J A McGuire, H A Dinon Aldridge, and Ryan P Boyles. 2014. Quality Control Methods for the North Carolina Environment and Climate Observing Network. In *17th Symposium on Meteorological Observation and Instrumentation*. American Meteorological Society.

[3] Sheila M Saia, Sean P Heuser, Myleigh D Neill, William A LaForce IV, J Jared A McGuire, and Kathie D Dello. 2023. A Technical Overview of the North Carolina ECONet. *Journal of Atmospheric and Oceanic Technology* 40, 6 (2023), 701–717.

A Appendix

A.1 Meeting Attendance

To ensure steady progress and divide the work evenly for the project, our group scheduled and attended the following meetings:

- **Meeting 1**
  - **Date:** 03/25/2026
  - **Time:** 5:00 PM–6:00 PM
  - **Attendees:** Manu, Bedant, Mukul
- **Meeting 2**
  - **Date:** 03/29/2026
  - **Time:** 4:30 PM–5:30 PM
  - **Attendees:** Manu, Bedant, Mukul
- **Meeting 3**
  - **Date:** 04/03/2026
  - **Time:** 4:30 PM–6:00 PM
  - **Attendees:** Manu, Bedant, Mukul
- **Meeting 4**
  - **Date:** 04/06/2026
  - **Time:** 5:30 PM–7:00 PM
  - **Attendees:** Manu, Bedant, Mukul
- **Meeting 5**
  - **Date:** 04/17/2026
  - **Time:** 6:30 PM-7:30 PM
  - **Attendees:** Manu, Bedant, Mukul
- **Meeting 6**
  - **Date:** 04/28/2026
  - **Time:** 9:00 PM-10:00 PM
  - **Attendees:** Manu, Bedant, Mukul

A.2 Hyperparameter Tuning Approach

Primary Evaluation Metric: F1 score, chosen because the positive class is rare and plain accuracy would be misleading.

Tuning Approach: Random search for the tree-based models and a smaller grid-style search for the simpler linear model. This kept the experiments computationally feasible while still exploring the most important hyperparameters.

Validation Approach: A fixed holdout validation split derived entirely from the provided training data, with fit, tuning, and final validation portions. The same validation protocol was preserved across experiments for fair comparison.

Models and Search Spaces. For each model family below, we selected hyperparameters that primarily control regularization, model complexity, and class-imbalance handling. These are the most important factors for this problem because the feature space contains both categorical and engineered temporal signals, and the target distribution is highly skewed.

Table 2: Hyperparameter Search Spaces Used in Final Experiments

Model	Hyperparameters and Search Space
Logistic Regression	`C` $\in \{0.01, 0.1, 1, 10\}$; `penalty` $\in \{$`l1`, `l2`$\}$; `class_weight` $\in \{$`None`, `balanced`$\}$ These control regularization strength, the sparsity or smoothness of the linear decision boundary, and whether the rare positive class is upweighted.
Decision Tree	`max_depth` $\in \{3, 5, 10,$ `None`$\}$; `min_samples_split` $\in \{2, 5, 10\}$; `min_samples_leaf` $\in \{1, 2, 5\}$; `ccp_alpha` $\in \{0.0, 0.001, 0.01\}$; `class_weight` $\in \{$`None`, `balanced`$\}$ These hyperparameters mainly control tree size, pruning, and overfitting while also allowing us to test whether class reweighting improves recall on true errors.
Random Forest	`n_estimators` $\in \{100, 300, 500\}$; `max_depth` $\in \{$`None`, 10, 20$\}$; `min_samples_split` $\in \{2, 5, 10\}$; `min_samples_leaf` $\in \{1, 2, 4\}$; `max_features` $\in \{$`sqrt`, `log2`, 0.5$\}$; `class_weight` $\in \{$`None`, `balanced`$\}$ These are the main controls on ensemble size, tree complexity, feature subsampling, and imbalance handling for a tabular random forest.
HistGradientBoosting / Gradient Boosting	`max_iter` or `n_estimators` $\in \{100, 200, 300\}$; `learning_rate` $\in \{0.01, 0.05, 0.1\}$; `max_depth` $\in \{3, 5, 7\}$; `min_samples_leaf` $\in \{20, 50, 100\}$; `l2_regularization` $\in \{0.0, 0.1, 1.0\}$ These control the strength of boosting, the complexity of individual trees, and the amount of regularization applied to reduce overfitting.

Computation Strategy: Because the dataset is massive, we used a staged tuning strategy. Simpler models and smaller search spaces were evaluated first to establish reasonable baselines. We then focused more computation on the strongest tree-based methods and used Spark for the largest context-rich experiments to keep the full project computationally feasible while preserving the same validation protocol.

Final blind test submission model: For the unlabeled `test.csv` submission, we used the base-plus-context Random Forest with `numTrees = 100`, `maxDepth = 20`, `minInstancesPerNode = 4`, `featureSubsetStrategy = sqrt`, and no class balancing.