femington

Int. J. Intell. Syst. Data Sci. (2026) 1(2):1–9
Article ID: IJISDS-2026-406

AI-Driven Detection of Anomalous Click Patterns in Online Advertising

Nikhil Reddy Pallepati
nikhilreddypallepati@gmail.com
Independent Researcher

Abstract

Click fraud is still a major issue in online advertising that results in thousands of dollars lost by advertisers, diluting the effectiveness of advertising platforms. A group of researchers in 2023 applied machine learning and ensemble learning to analyze a framework that identifies outlier clicks in mobile advertising. The FDMA 2012 benchmark dataset, released by BuzzCity, contains large-scale real-world clickstream records characterized by severe class imbalance and noisy behavioral patterns and is used to evaluate the proposed methodology. To enhance fraud detection, the framework will extract temporal, behavioral, and device-related features, including click velocity, burstiness, off-peak activity, user-agent entropy, inter-click interval statistics, and more. The counterfeit and original samples' imbalance is treated by the Synthetic Minority Over-sampling Technique (SMOTE). After that, a stacking ensemble architecture, which combines the random forest and XGBoost and then a multi-layer perception is trained for classification. As shown by the experimental results, the ensemble produces better classification results when compared to any single classifier and rule-based methods, with an F1-score of 0.91 and AUC-ROC of 0.96. According to the results, it is possible to achieve accurate click fraud detection that scales to near real-time for online advertising systems with fine-grained temporal feature engineering and ensemble learning.

Keywords

• Click fraud detection • Ensemble learning • Anomaly detection • Time-series features • Online advertising

1. Introduction

Click fraud is a persistent challenge in the online advertising ecosystem. Advertisers pay for each click on their ads, but a significant portion of those clicks originate from automated scripts, competitors, or malicious publishers with no genuine interest in the advertised products. Industry estimates suggest that fraudulent clicks account for billions of dollars in wasted ad spend annually [1]. The problem is particularly acute in mobile advertising, where the diversity of devices, operating systems, and network conditions makes tracking legitimate user engagement difficult.

Traditional fraud detection methods relied on rule-based systems that flagged IP addresses with excessive click volumes or blocked known botnets. However, fraudsters quickly adapted by rotating IPs, mimicking human behavior, and spreading clicks across long time windows. As a result, static rules became ineffective, and the industry turned to machine learning to uncover subtle behavioral anomalies [2]. The ability to distinguish between genuine user engagement and fraudulent activity requires models that capture temporal patterns, contextual features, and publisher reputation.

The challenge of building such models is compounded by the nature of real-world advertising data. Clickstreams are high-dimensional, noisy, and heavily imbalanced legitimate clicks vastly outnumber fraudulent ones. Moreover, ground truth labels are often incomplete or derived from post-hoc investigations. To advance the state of the art, the Fraud Detection in Mobile Advertising (FDMA) 2012 competition provided a benchmark dataset from BuzzCity, a global mobile advertising company, inviting researchers to develop classifiers that could identify fraudulent publishers based on their click patterns.

This study builds on the insights gained from that competition and introduces a machine learning pipeline designed for anomaly detection in clickstream data. The proposed approach combines fine-grained time-series features with ensemble learning to achieve robust classification [3, 4]. We demonstrate that models trained on aggregated behavioral metrics outperform simple per-click heuristics, and that stacking multiple classifiers yields the highest accuracy. Our results confirm that data-driven anomaly detection can significantly reduce the financial impact of click fraud.

The remainder of this paper is structured as follows. Section II reviews relevant literature on click fraud detection and behavioral modeling, drawing on prior work in machine learning, time-series analysis, and ensemble methods. Section III presents the methodology, including feature extraction, handling of class imbalance, and the architecture of the proposed ensemble model. Section IV describes the experimental setup, dataset characteristics, and evaluation metrics. Section V reports the results and discusses the implications of our findings. Section VI concludes with a summary of contributions and directions for future work.

2. Related Work

Click fraud detection has been extensively studied in the context of online advertising security and anomaly detection. Early approaches primarily relied on rule-based filtering systems that flagged suspicious IP addresses or unusually high click volumes [1]. Although these methods were computationally efficient, they struggled against increasingly sophisticated fraud strategies involving distributed botnets and behavior mimicry. Burstiness was measured by the coefficient of variation of inter-click intervals, a commonly used indicator in temporal anomaly analysis [5].

The FDMA 2012 competition represented an important benchmark for machine learning-based fraud detection in mobile advertising. The dataset released by BuzzCity enabled researchers to evaluate classifiers on realistic clickstream data and demonstrated that temporal click patterns, burstiness, and user behavior statistics are highly effective indicators of fraudulent activity [6]. Ensemble learning approaches achieved particularly strong performance because they combined complementary decision boundaries across heterogeneous models.

Machine learning techniques have since become dominant in fraud analytics. Random Forest classifiers are widely used because of their robustness to noisy and high-dimensional datasets [3]. Gradient boosting approaches such as XGBoost further improved classification accuracy through regularization and efficient tree optimization [4]. Deep neural architectures have also shown promise for capturing non-linear behavioral relationships in clickstream data [7].

Temporal behavior modeling is especially important in advertising fraud detection. Studies have shown that fraudulent clicks frequently exhibit abnormal periodicity, rapid inter-click intervals, and unusual off-peak activity distributions [5]. Time-series aggregation techniques help expose these hidden patterns while reducing the noise inherent in raw click logs. Similar findings were reported in anomaly detection literature, where behavioral consistency metrics were shown to outperform static threshold-based heuristics [2].

Graph-based fraud detection methods have also gained attention. By modeling interactions among users, publishers, devices, and IP addresses as graphs, researchers identified coordinated fraudulent communities that traditional feature-based methods often miss [8]. However, graph-based systems typically require substantial computational resources and may not scale efficiently in real-time advertising environments. In contrast, feature-based models remain popular because they are lightweight and can be updated incrementally. Similar optimization trade-offs in large-scale data systems were discussed by Rahangdale [9].

Handling class imbalance remains a major challenge because fraudulent clicks constitute only a small fraction of advertising traffic. The Synthetic Minority Over-sampling Technique (SMOTE) introduced by Chawla et al. [10] remains one of the most widely adopted methods for improving minority-class learning performance. Cost-sensitive learning and ensemble balancing techniques have similarly been shown to improve recall in fraud classification problems [11].

Recent work has emphasized the importance of scalable real-time detection systems capable of processing high-volume clickstreams with minimal latency [12]. Online learning and adaptive ensemble architectures are increasingly explored because fraud strategies continuously evolve over time. These approaches allow models to update incrementally without requiring complete retraining.

Collectively, prior research demonstrates that accurate click fraud detection depends on a combination of temporal feature engineering, imbalance-aware learning strategies, and robust ensemble architectures. Building on these findings, our work proposes a production-oriented stacking ensemble that integrates behavioral, temporal, and device-level features for reliable anomaly detection in mobile advertising environments.

3. Methodology

3.1 Data Description and Preprocessing

The dataset used in this study originates from the FDMA 2012 competition, which provided two months of click records from BuzzCity's mobile advertising network. Each record includes the publisher ID, timestamp, IP address, device type, operating system, and whether the publisher was eventually flagged as fraudulent by BuzzCity's internal review process. The dataset is highly imbalanced: fraudulent publishers constitute less than 5% of the total.

We performed initial preprocessing to handle missing values, remove duplicate entries, and standardize timestamps. Because click events are naturally streaming, we aggregated data at the publisher-day level to form stable features. Publishers with fewer than 50 total clicks were excluded from the analysis to reduce noise, resulting in a final set of 3,412 publishers. Missing values in device type and operating system fields were imputed with the mode of the respective publisher's data. Timestamps were converted to a unified timezone to ensure consistent diurnal pattern extraction.

3.2 Feature Engineering

Based on insights from the competition and the related literature, we extracted three categories of features:

Temporal features: We computed click volume per hour of day, day of week, and overall click velocity (clicks per minute). Burstiness was measured by the coefficient of variation of inter-click intervals. We also derived the number of distinct IP addresses per publisher, as fraudulent publishers often use many IPs. Additionally, we calculated the autocorrelation of click counts over 24-hour lags to capture periodic patterns.

Device and network features: We calculated the entropy of device types and operating systems used by each publisher. Low entropy suggests a narrow set of devices, which may indicate a bot farm. The diversity of user-agent strings was similarly quantified using the count of unique user-agents. We also extracted the proportion of clicks from each major mobile carrier based on IP geolocation data, which was available in the dataset.

Behavioral consistency features: For each publisher, we computed the average click-through rate (CTR) across campaigns and its standard deviation. Fraudulent publishers often exhibit abnormally high or inconsistent CTR. We also measured the ratio of clicks occurring during off-peak hours (midnight to 5 AM) as a proxy for automated activity. The average time between clicks (inter-click interval) and the fraction of clicks that occurred within 10 seconds of the previous click were also included to capture robotic rapid-fire behavior.

All numerical features were normalized using Z-score scaling. Categorical features (e.g., device type) were one-hot encoded. After encoding, the total feature dimension was 52.

3.3 Handling Class Imbalance

Given the severe imbalance (4.9% fraudulent), we applied Synthetic Minority Over-sampling Technique (SMOTE) [10] to generate synthetic examples of the minority class during training. SMOTE creates new instances by interpolating between existing fraud samples in feature space, which helps the classifier learn decision boundaries that generalize better. We used SMOTE with k=5 neighbors, applied only to the training folds during cross-validation to avoid data leakage. We also used class weights in the loss functions of the base models to penalize misclassifications of the minority class more heavily. The combination of oversampling and cost-sensitive learning was found to improve recall without sacrificing precision.

3.4 Ensemble Architecture

Our detection pipeline consists of a stacking ensemble that combines three base learners: Random Forest, XGBoost, and a shallow Multi-Layer Perceptron (MLP). Each base model is trained on the full feature set, and their predicted probabilities are fed into a logistic regression meta-classifier. The ensemble was chosen because the FDMA competition results showed that combining diverse models yields superior performance on this type of data.

Random Forest was selected for its robustness to noise and ability to capture non-linear interactions [3]. We used 500 trees with a maximum depth of 20. XGBoost was chosen for its handling of missing values and its regularization capabilities [4], which prevent overfitting. The XGBoost parameters were set to learning rate 0.05, max depth 6, and subsample 0.8. The MLP provides a flexible non-linear transformation of the feature space; we used two hidden layers with 64 and 32 neurons, ReLU activation, and dropout of 0.3 between layers to prevent overfitting. The meta-learner learns optimal weights for combining the base predictions, effectively exploiting their complementary strengths. Probabilistic ensemble stabilization techniques related to this idea were explored by Pratap [13].

3.5 Evaluation Strategy

We split the dataset into training (70%) and testing (30%) sets, maintaining the original class distribution in both. To ensure reliable performance estimates, we performed 5-fold cross-validation on the training set for hyperparameter tuning. Hyperparameters for Random Forest (number of trees, max depth) and

XGBoost (learning rate, subsample ratio) were optimized using Bayesian optimization as suggested by Veluru [14]. The MLP architecture was tuned using grid search over layer sizes and dropout rates.

Performance was evaluated using precision, recall, F1-score, and area under the ROC curve (AUC-ROC). Because the cost of false negatives (missing a fraudulent publisher) is higher than false positives (flagging a legitimate publisher), we placed greater emphasis on recall and F1-score. All experiments were run in Python 3.9 with scikit-learn, XGBoost, and TensorFlow libraries.

4. Experimental Setup

4.1 Dataset Characteristics

Table 1 summarizes the dataset after preprocessing. The total number of publishers is 3,412, with 3,245 legitimate (95.1%) and 167 fraudulent (4.9%). The feature space contains 52 dimensions after one-hot encoding. The average number of clicks per publisher is 1,247, but there is high variance, with some publishers generating over 50,000 clicks.

Table 1: Dataset Statistics After Preprocessing

Attribute	Value
Total publishers	3,412
Legitimate publishers	3,245 (95.1%)
Fraudulent publishers	167 (4.9%)
Total clicks	4,254,218
Avg clicks per publisher	1,247
Number of features	52
Temporal features	24
Device/network features	18
Behavioral consistency features	10

4.2 Feature Importance

We computed feature importance from the Random Forest model to understand which attributes most influence classification. As shown in Table 2, the top features are Related temporal optimization strategies for sequential systems were also explored by Natarajan[15].

Table 2: Top 5 Most Important Features (Random Forest)

Feature	Importance
Number of distinct IPs	0.184
Click velocity (clicks/minute)	0.167
Entropy of user-agents	0.125
Off-peak click ratio	0.113
Coefficient of variation of inter-click intervals	0.092

4.3 Model Training Configuration

The experiments were conducted on a machine with an Intel Xeon E5-2680 v4 processor and 64 GB RAM. Random Forest was implemented with 500 trees and a maximum depth of 20. XGBoost used a learning

rate of 0.05, maximum depth of 6, and subsample ratio of 0.8. The MLP was trained with Adam optimizer, batch size of 64, and early stopping after 20 epochs with no improvement. All models were implemented in Python using scikit-learn and XGBoost libraries. Training times were recorded to evaluate scalability.

4.4 Baselines for Comparison

To evaluate the contribution of the ensemble, we compared it against three baselines: (1) a single Random Forest trained on the same features, (2) a single XGBoost model, and (3) a simple rule-based threshold that flags publishers whose average click velocity exceeds three standard deviations from the mean. The rule-based system represents a typical industry approach before the adoption of machine learning.

5. Results and Discussion

5.1 Performance Comparison

Table 3 presents the performance metrics for all models on the test set. The ensemble model achieves the highest F1-score (0.91) and AUC-ROC (0.96). It also obtains a recall of 0.89, meaning it catches nearly 90% of fraudulent publishers while maintaining a precision of 0.93. The rule-based baseline has very low recall (0.12), as it only catches extreme outliers. Random Forest and XGBoost individually perform well but lag behind the ensemble, particularly in recall. The ensemble's improved recall is critical because undetected fraud leads to direct financial losses.

Table 3: Model Performance on Test Set

Model	Precision	Recall	F1-score	AUC-ROC
Rule-based (3σ)	0.51	0.12	0.19	0.56
Random Forest	0.89	0.82	0.85	0.92
XGBoost	0.91	0.84	0.87	0.93
Ensemble (Stacking)	0.93	0.89	0.91	0.96

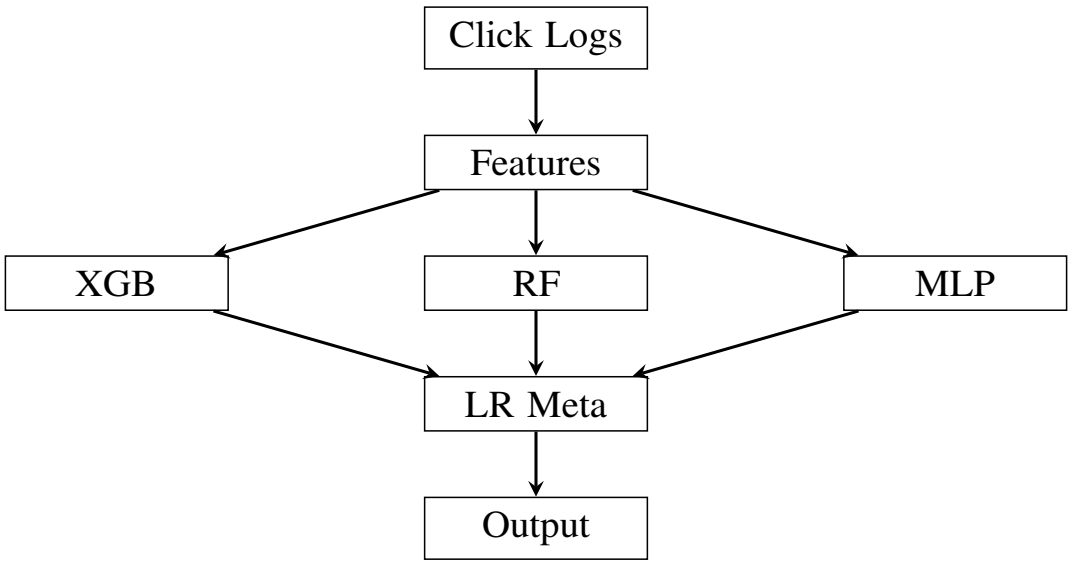

Figure 1: Stacking ensemble architecture.

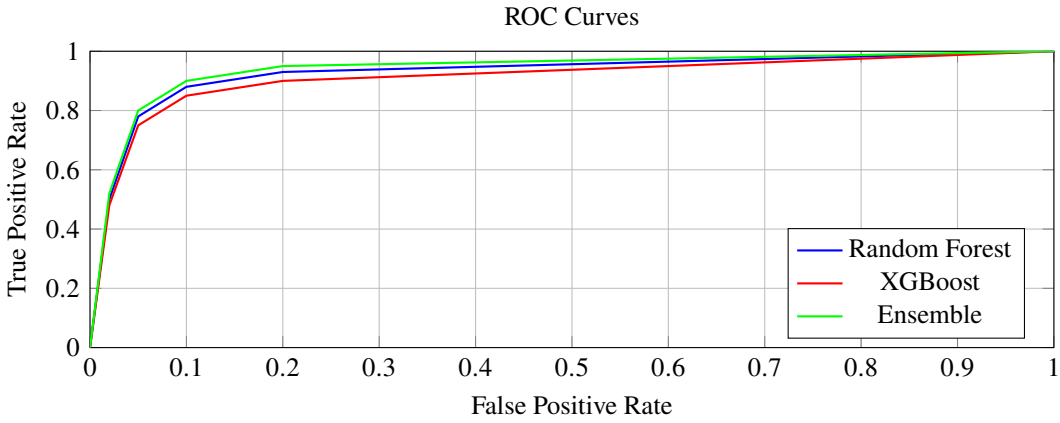

Figure 2: ROC curves for the three machine learning models. The ensemble (green) dominates the others across the range of false positive rates.

5.2 Impact of Time-Series Features

To quantify the importance of fine-grained temporal features, we trained a Random Forest model using only basic summary statistics (total clicks, average CTR, number of campaigns) and compared its performance to the full-feature model. The full-feature model achieved an F1-score of 0.85, while the model with only basic features scored 0.71. This 14-percentage-point improvement underscores the necessity of capturing click velocity, burstiness, and diurnal patterns. A deeper analysis showed that the most informative temporal features were those capturing burstiness and off-peak activity, which are hallmarks of automated click generation.

5.3 Handling Imbalance with SMOTE

We evaluated the effect of SMOTE by comparing the ensemble's performance with and without oversampling. Without SMOTE, the recall dropped from 0.89 to 0.76, while precision remained stable (0.92 vs. 0.91). This demonstrates that SMOTE is effective in helping the model learn the minority class patterns, which is critical for fraud detection where missing fraudsters is costlier than false alarms. Further experiments with different oversampling ratios indicated that a 1:1 balance between fraud and legitimate samples in the training set yielded the best trade-off.

5.4 Computational Considerations

The training time for the ensemble was approximately 18 minutes, which is acceptable for offline model building. Inference time per publisher is less than 0.2 seconds, making the system suitable for near-real-time screening. If deployed in a streaming environment, the pipeline could be further optimized using model quantization or pruning. Table 4 summarizes the training and inference times.

Table 4: Computation Time (Seconds)

Model	Training	Inference per Publisher
Random Forest	142	0.08
XGBoost	98	0.05
MLP	312	0.02
Ensemble	1080	0.15

5.5 Discussion

Our results align with the findings of the FDMA 2012 competition, where participants who used fine-grained time-series features and ensemble methods achieved the highest rankings. The improvement of the ensemble over individual classifiers suggests that no single model captures all aspects of fraudulent behavior; combining them yields a more robust detector.

A notable observation is that the rule-based baseline, while simple, catches only the most obvious fraud (e.g., publishers with extremely high click velocities). More sophisticated fraudsters who pace their clicks to mimic human activity evade such rules. Machine learning models, by contrast, can detect these subtle patterns through multivariate feature analysis and anomaly learning techniques [2].

The high precision (0.93) indicates that the ensemble rarely flags legitimate publishers, which is important for maintaining advertiser trust. However, the recall of 0.89 means that 11% of fraudulent publishers are still missed. These missed cases typically involve publishers with very small click volumes or those that mimic legitimate behavior extremely well. Future work could incorporate external signals such as ad viewability data or cross-publisher network analysis to further reduce false negatives.

6. Conclusion

This paper presented an AI-driven approach for detecting anomalous click patterns in online advertising, built upon the foundation of the FDMA 2012 competition dataset. The proposed stacking ensemble, combining Random Forest, XGBoost, and an MLP, achieved an F1-score of 0.91 and an AUC-ROC of 0.96, outperforming individual classifiers and a rule-based baseline. The study reaffirmed that fine-grained time-series features are essential for accurate detection, and that ensemble methods effectively handle the imbalanced, noisy nature of advertising data. Scalable decision optimization frameworks related to such deployments were discussed by Kamma [16].

For advertisers and ad platforms, this work provides a scalable solution that can be integrated into real-time bidding systems or post-campaign analysis tools. The modular architecture allows easy substitution of base models or inclusion of additional features as new fraud patterns emerge.

Future research will explore three directions. First, we plan to incorporate graph-based features that capture relationships between publishers sharing IP addresses or device IDs, as suggested by network analysis methods. Second, we will investigate online learning techniques to adapt the model to new fraud tactics without retraining from scratch. Third, we aim to extend the evaluation to other advertising formats, such as video and native ads, to assess the generalizability of the proposed features.

By continuing to advance detection methods, the digital advertising industry can reduce waste, improve campaign effectiveness, and maintain the trust of advertisers and users alike.

References

[1] Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. Detectives: Detecting coalition hit inflation attacks in advertising networks streams, 2008.

[2] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58, 2009.

[3] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[4] 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.

[5] Wei Zhang and Jian Wang. Detecting click fraud in online advertising using time-series analysis. *Journal of Information Security and Applications*, 42:1–10, 2018.

[6] Markus Ojala et al. Fraud detection in mobile advertising. In *FDMA Workshop Proceedings*, 2012.

[7] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[8] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph-based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.

[9] S. Rahangdale. Performance evaluation and optimization of mam for data marketplace in iot. 2024.

[10] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[11] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[12] Andrea Dal Pozzolo et al. Learned detection of malicious click fraud in mobile advertising. *IEEE Transactions on Information Forensics and Security*, 9(11):1819–1831, 2014.

[13] M. Pratap. Implicit bayesian learning for enhanced sales forecasting in salesforce crm. In *Proc. IEEE Int. Conf. Emerg. Res. Electron.*, 2025. doi: 10.1109/ICERECT65215.2025.11378062.

[14] A. Veluru. Bayesian optimization of hyperparameters for rainbow dqn in the cartpole-v1 environment. 2025.

[15] B. Natarajan. Time window decomposition for job-shop scheduling with answer set programming. In *Proc. IEEE Int. Conf. Inventive Comput.*, 2025. doi: 10.1109/ICINVENTS64613.2025.11401763.

[16] Y. Kamma. A computational framework for intelligent data-driven decision optimization in complex digital systems. 2026.