Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

Enhanced Twitter Bot Detection via Static and Temporal Feature Integration

Zahraa E. H. Al-Khersan* and Nahla A. Flayh[b, *]

[a, b] *Department of Information Computer Systems, Faculty of Computer Science and Information Technology, Basra University, Basra, Iraq.*
zahraa.emadd@uobasrah.edu.iq, nahla.flayh@uobasrah.edu.iq

ARTICLE INFO

Article history:
Received: 07 /09/2025
Rrevised form: 08 /11/2025
Accepted : 29 /12/2025
Available online: 30 /12/2025

Keywords:

Temporal, Real, Features, Bot, Twitter

ABSTRACT

This study presents a robust supervised learning framework tailored for the precise identification of Twitter bots. The central contribution lies in the integration of both static and temporal features to establish a comprehensive representation of user behavior. Using the benchmark Cresci-2017 dataset, the framework constructs a rich feature space encompassing profile-level metadata, detailed posting frequency statistics, and fine-grained temporal activity features, enabling effective discrimination between sophisticated bots and genuine human users. To fully exploit this diverse information, an advanced stacking ensemble is employed. This architecture strategically combines the predictive strengths of multiple base learners Random Forest, LightGBM, XGBoost, and Support Vector Machine while a Logistic Regression meta-learner is trained to optimally integrate their outputs. The resulting model delivers state-of-the-art performance, achieving 98.71% accuracy, 99.26% precision, 99.04% recall, and a 99.15% F1-score, substantially outperforming individual classifiers. These findings underscore the value of unifying heterogeneous feature types with stacked generalization, highlighting that strategically diverse feature engineering coupled with advanced ensemble methods is essential for building resilient defenses against the rapidly evolving landscape of automated bot behavior.

MSC.

https://doi.org/10.29304/jqcsm.2025.17.42574

1. Introduction

Social bots on platforms such as Twitter represent a pressing challenge, as they are extensively employed to disseminate misinformation and influence public discourse [1]. With their increasing sophistication and ability to closely mimic human behavior, traditional detection strategies that rely primarily on static features (e.g., follower count, account age) are no longer sufficient [2]. Recognizing that temporal features signals such as posting frequency, engagement rhythms, and activity cycles play a pivotal role in accurate detection, this study introduces a novel framework that integrates these dynamic features with static profile attributes. This combined perspective enhances the ability to detect subtle irregularities that may reveal automated activity. The proposed model was developed and evaluated using the widely adopted Cresci-2017 benchmark dataset, which encompasses annotated genuine accounts alongside multiple categories of bots. To achieve high predictive performance, a stacking ensemble methodology was employed. This architecture capitalizes on the complementary strengths of four heterogeneous base classifiers: Random Forest, LightGBM, XGBoost, and Support Vector Machine (SVM). Their individual predictions were then

∗Corresponding author

Email addresses: zahraa.emadd@uobasrah.edu.iq

Communicated by ‘sub etitor’

consolidated through a Logistic Regression meta- learner, which synthesizes the outputs to generate the final decision. This layered design enables the model to more effectively capture complex and deceptive behaviors.

The key contributions of this study are as follows:

- Feature Integration – A unified representation combining static and temporal features for Twitter bot detection.
- Ensemble Optimization – Implementation of a stacking model that outperforms individual classifiers.
- High-Performance Results – Achieving 98.71% accuracy enhancing precision, recall, and F1-score metrics beyond what individual models can attain.

2. Related works

Several studies have explored ensemble learning techniques for social media analysis and bot detection, integrating diverse features to enhance classification accuracy. In [3], a hybrid sentiment analysis model was applied to tweets from the Clean India Mission. Using a manually labelled subset of 1,008 tweets, the authors combined lexicon-based scoring with a stacked ensemble of SVM, KNN, and C5.0, employing Random Forest as the meta-classifier. The approach achieved 91.24% accuracy in 10-fold cross-validation, outperforming individual models. A large-scale analysis in [4] examined 69,000 active Twitter accounts across six bot types using over 400 features spanning content, user, temporal, network, sentiment, and hashtags. A Random Forest model with ADASYN addressed binary classification, while three advanced multi-class models including a stacking ensemble—were evaluated. Temporal features were particularly effective for detecting political bots, whereas content features were most useful for identifying social bots. The best model achieved 89.8% accuracy and an F1-score of 0.904, with LIME providing interpretability.

An analysis was conducted in [5] on 2,970 tweets collected from Twitter between July and September 2019, equally divided between vaping-related and general content, using keywords such as “vape” and “ecigarette”. After removing noise, including emojis and hyperlinks, the text was vectorized using the TF-IDF representation. Seven machine learning algorithms, such as SVM and Random Forest, along with a deep learning Transformer model, were evaluated. The Stacking Ensemble achieved superior performance at 97% accuracy, followed by Random Forest and Transformer, each with 96%. These models can support public health initiatives by analyzing user sentiment toward vaping and identifying individuals at higher risk from its harmful effects. In [6], an ensemble driven stacking framework merged six classifiers Logistic Regression, Decision Tree, Random Forest, XGBoost, SVM and MLP using a soft voting mechanism. Trained on a Kaggle dataset reinforced with bot and human account features, the model achieved 90.22% accuracy and 92.39% precision, consistently outperforming individual classifiers. [7] used the MIB dataset of 5,301 labelled Twitter accounts to evaluate a stacking ensemble of Random Forest, SVM and Naive Bayes with Logistic Regression as the meta-learner. Feature selection employed Spearman's rank correlation and the chi-square test. followed by Min-Max normalization. The model achieved 99.02% accuracy and a 99.22% F1score, outperforming all individual classifiers. A modular ensemble model for cross-platform bot detection was presented in [8], combining Decision Tree, Random Forest, and Gradient Boosting with calibrated probabilities. This method remained effective even with partial feature sets and achieved an average accuracy of 75.47% across nine datasets from Twitter, Reddit, and Instagram, outperforming Botometer and BotHunter. The researchers of [9] developed a hybrid framework using the TwiBot-20 dataset, combining engineered profile/content features with NLP-derived textual features. A stacking ensemble with Logistic Regression as the metaclassifier achieved state-of-the-art results, with an F1-score of 90.22%, precision of 84.66%, and recall of 96.56%. In [10], SStackGNN, a stacking-based graph neural network, integrated GCN, GAT, and SGC base models with an MLP meta-classifier. Tested on Cresci-15, TwiBot-20, and MGTAB datasets, it achieved accuracies of 97.69%, 84.87%, and 87.78%, respectively, improving F1-scores by up to 15.48% over baselines while reducing computation time by 80%. The work in [11] collected over 2.1 million tweets using targeted hashtags and Botometer scores to quantify bot likelihood. Thirty-seven user and content features were extracted, and a stacking ensemble of Random Forest and Gradient Boosting with RidgeCV as the meta-learner was implemented. The framework achieved a mean squared error of ≈0.60 and incorporated LIME for explainability, enabling deployment as the interactive “Bot Detective” application. Finally, [12] developed a multistage stacked ensemble for detecting fake accounts on Facebook, Twitter, and Instagram, incorporating Chi-squared feature selection, cost-sensitive learning, and an MLP meta-learner with Meta Cost. The approach achieved 95% accuracy on Facebook, 98.2% on Instagram, and 81% on Twitter, while maintaining low false positive rates.

3. Method and Methodology

3.1 Dataset and Preprocessing

This study uses the Cresci-2017 dataset, which contains labelled Twitter accounts, including genuine users, social spambots, traditional spambots, and fake followers [13]. The dataset was first cleaned to remove incomplete or duplicate records. Irrelevant attributes were discarded, and all features were standardized to ensure consistent scaling [14].

The Cresci-2017 dataset was deliberately used because it remains one of the most widely accepted and well-structured benchmark datasets for Twitter bot detection. It provides verified labels and balanced samples, allowing fair comparison with numerous prior works. The goal of this research was methodological improvement rather than dataset diversity; hence, using a standard dataset ensured that improvements were due to the proposed approach, not differences in data.

3.2 Data Analysis

Twitter account data was systematically analyzed to extract and expectation the characteristics of the behavioral features between bot accounts and real users. The procedure involved: (1) extracting a reliable set of distinguishing features, and (2) evaluate their relative contribution in classification performance (Random Forest, LightGBM, XGBoost, Support Vector Machines and Logistic Regression). Figure 1 shows the features importance collected across all models, emphasizing the most prominent discriminators in bot detection.

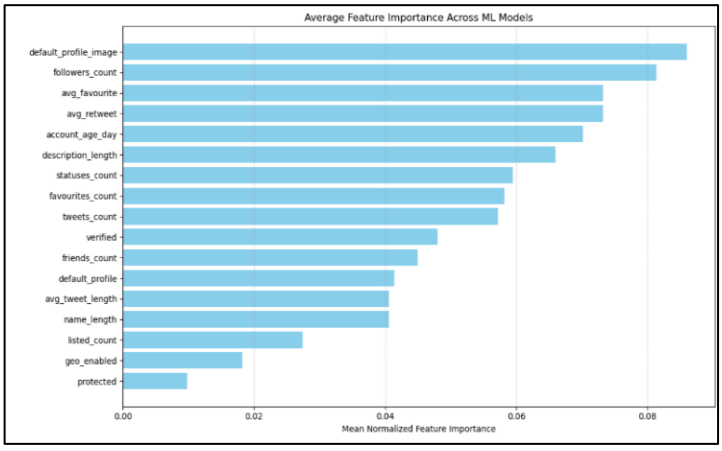


Fig. 1 - Average feature importance across machine learning models for Twitter bot detection

3.3 Hyperparameter Tuning

Hyperparameters represent the foundational configuration settings that dictate the structure and control the learning mechanics of a machine learning model. These parameters stand in contrast to the model's internal, trainable parameters, which are iteratively adjusted and optimized from the data during the training phase itself. In contrast, hyperparameters are established prior to the commencement of training. They exert a profound influence on the model's core capabilities, including its expressive power (capacity), the computational efficiency of the training process (speed), and its ultimate ability to perform well on new, unseen data (generalization ability). The meticulous process of refining these values is consequently an essential and critical step in the model development workflow. This fine-tuning is paramount to securing a balanced performance profile and serves as the primary mechanism for preventing the twin pitfalls of underfitting, where the model is too simplistic to capture patterns in the data, and overfitting, where the model memorizes the training data too closely and fails to generalize. Within the scope of this research, a comprehensive hyperparameter optimization procedure was executed. This process was applied universally to every constituent model, encompassing all base learners as well as the Logistic Regression model that acts as the meta-learner within the stacking ensemble architecture. The specific strategy employed for this optimization was a grid search methodology coupled with cross-validation. This approach involves the

systematic and exhaustive evaluation of a pre-defined, multi-dimensional grid of numerous candidate hyperparameter value combinations. The core mechanism of cross-validation ensures that each configuration is assessed based on a robust and reliable estimate of its predictive performance, ultimately guiding the selection of the single hyperparameter configuration that maximizes predictive accuracy and model effectiveness. The specific hyperparameters subjected to tuning were selected based on the intrinsic nature of each model type to ensure a targeted and effective optimization. For the Random Forest algorithm, the key parameters adjusted were the number of individual decision trees within the ensemble and the maximum depth permitted for each tree. For the gradient boosting frameworks, namely LightGBM and XGBoost, the optimization focused on the learning rate which controls the step size during boosting, the number of boosting estimators (iterations), and the various sampling ratios which control the proportion of data or features used for training each tree. For the Support Vector Machine (SVM), a careful calibration was performed on the kernel function which defines the data transformation, the regularization term which balances the trade-off between achieving a large margin and minimizing classification error, and the gamma parameter which defines the influence range of a single training example. Finally, for the Logistic Regression meta-learner, the tuning process targeted the type of penalty applied to the coefficients (e.g., L1 or L2) and the magnitude of the regularization strength that governs this penalty.

This rigorous and model-specific hyperparameter optimization strategy, which was carefully aligned with the underlying characteristics of the dataset, enabled the final stacking ensemble to realize its complete and full potential. As a direct result of this meticulous configuration, the ensemble demonstrated significantly enhanced robustness, a greater degree of adaptability to complex patterns, and ultimately, superior detection and predictive outcomes as measured across the entire suite of evaluation metrics.

3.4 Feature Extraction

User profiles were represented through two complementary categories of features: static and temporal features. Static features (e.g., follower count, account age) capture properties that remain constant over time, while temporal features characterize behavioral features such as posting frequency, average engagement levels, and diurnal activity distributions. By combining these perspectives, the framework constructs a richer and more resilient representation of user behavior, which is essential for identifying advanced bots that can bypass detection when relying solely on static indicators. A detailed overview of all extracted features is presented in Table 1.

Table 1- Extracted features for Twitter bot detection.

Feature Name	Description	Type
Follower Count	Total number of followers	Static
Following Count	Total number of accounts followed	Static
Account Age	Days since account creation	Static
Verified Status	Boolean indicating if account is verified	Static
Geo-enabled	Boolean indicating if location tagging is enabled	Static
Posting Frequency	Average number of tweets per day	Temporal
Average Engagement Rate	Average number of likes/retweets per tweet	Temporal
Temporal Activity Trends	Distribution of tweets over hours/days	Temporal
Retweet Frequency	Average number of retweets per day	Temporal
Reply Frequency	Average number of replies per day	Temporal

This table outlines the feature set employed to train the machine learning model for Twitter bot detection. The features are organized into two primary groups: Static and Temporal. Static Features consist of attributes that are typically constant or slow to change, reflecting the core profile properties of a Twitter account. These features encompass:

- Follower Count and Following Count: These metrics represent the social reach and following behavior of an account. Bots often have anomalous follower-to-following ratios.
- Account Age: The number of days since the account was created. Many bots are recently created accounts.
- Verified Status: A boolean (true/false) indicating if Twitter has officially verified the account. Legitimate influential users are often verified, while bots are not.
- Geo-enabled: A Boolean indicating if the account has location tagging enabled for its tweets. This can be a signal of genuine user activity.

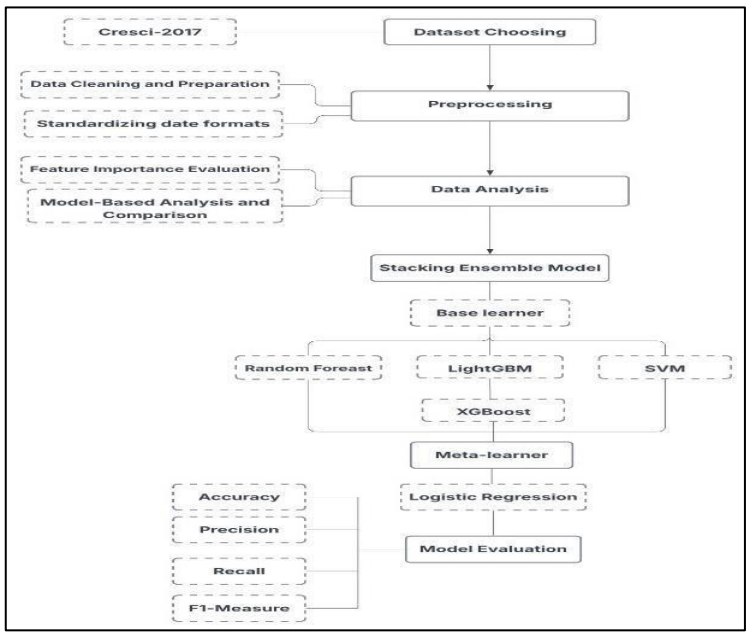
Temporal Features are time-dependent attributes that capture the dynamic behavioral patterns of an account, which are crucial for identifying automated activity. They include:

- Posting Frequency: The average number of tweets posted per day. Bots often exhibit extremely high or low and consistent posting rates unlike humans.
- Average Engagement Rate: The average number of likes and retweets a user's tweets receive. Bot accounts might have low engagement despite high posting activity or show suspiciously coordinated engagement patterns.
- Temporal Activity Trends: This interprets the distribution of tweets over different hours of the day or days of the week. Human users have activity patterns correlated with sleep and work, while bots may post 24/7 or at highly regular intervals.
- Retweet Frequency and Reply Frequency: The average number of retweets and replies carried out per day. Bots are often used to amplify certain content (high retweet frequency) or spam replies, and their behavior can be quantified through these metrics.

3.5 Stacking Ensemble Framework

The proposed detection framework follows these steps: (Fig. 2)

- Base Learner Training – Four classifiers - Random Forest, LightGBM, XGBoost and SVM - train separately using identical features.
- Meta-Learner Training – base model predictions become input features for a Logistic Regression classifier.
- Final Prediction – The meta-classifier produces the final classification result.



The stacking strategy leverages the strengths of diverse algorithms, allowing the model detects both linear and non-linear data patterns [15]. Logistic Regression was selected as the meta-learner for its interpretability and robustness in binary classification.

Fig. 2 - Flowchart showed the Stacking Model

The figure illustrates a structured workflow for Twitter bot detection using the Cresci-2017 dataset. The process begins with Dataset Choosing, where the Cresci-2017 dataset is selected. This is followed by Data Cleaning and Preparation, which includes standardizing the date format to ensure consistency. Next, Feature Importance Evaluation is performed to identify the most relevant attributes for detection. The core of the workflow is the Model-Based Analysis and Stacking Ensemble Model, which combines multiple machine learning algorithms specifically Random Forest and Logistic Regression into a stacked ensemble to improve prediction accuracy and robustness. This integrated approach leverages both individual model strengths and meta-learning for enhanced bot detection performance.

3.6 Model Training

The integrated dataset (static + temporal features) was split into 80% training and 20% testing using stratified sampling to preserve class distribution. All models were implemented in Python using Scikit-learn, XGBoost, and LightGBM libraries. Base learner hyperparameters were tuned through exhaustive grid search with 5-fold cross-validation.

3.7 Classifier Descriptions

- **Random Forest:** This algorithm builds a multitude of decision trees. Each tree is trained on a unique subset of the data and a random selection of features. For prediction, each tree votes on the outcome, and the class with the most votes becomes the final prediction. This ensemble approach enhances prediction accuracy by mitigating errors from bias and variance, resulting in a more stable and reliable model [16].

- **Support Vector Machine:** This supervised learning algorithm effectively handles both classification and regression tasks. It's mainly aim is to reach to the best decision boundary (hyperplane) that maximizes the distance between classes [17].

- **LightGBM:** is a highly competence gradation strengthening framework developed by Microsoft, designed for speed, expandability and high performance in machine learning aims. It builds set of decision trees step by step, where each new tree corrects errors from previous ones, making it more effect for large-scale datasets and complex feature interactions [18].

- **XGBoost:** is a highly designed, able to develop gradient-boosted decision trees (GBDT), planned for speed, accuracy and efficiency. Sequentially, it builds an ensemble of weak learners (decision trees), whoever, each new tree tries to correct errors made by previous ones, individually improving predictive performance one by one. It is widely applied in classification, regression and ranking tasks, including bot detection [19].

- **Logistic Regression:** One of the simplest methods of classification, yet highly effective in estimating the probability of a given sample belonging to a specific class. It was selected as the meta-learner due to its ability to combine the outputs of base models effectively and without complexity, enhancing overall model performance while reducing the risk of overfitting [20].

3.8 Model Evaluation

The execution of the utilized supervised machine learning algorithms was assessed using four essential measures:

- Accuracy: presents the ratio of correct predictions (positive and negative) among all estimations made by the model. It calculates as the equation below: [21]

$$Accuracy = \frac{T_P + T_N}{T_P + T_N + F_P + F_N} \dots\dots\dots (1)$$

- Precision: determine the percentage of correctly estimated positive instances to the total estimated positives. It is defined by the equation as follow: [22]

$$Precision = \frac{T_P}{T_P + F_P} \dots\dots\dots (2)$$

- Recall: indicates the ratio of real positive cases that the model accurately built in according to the following equation: [23]

$$Recall = \frac{T_P}{T_P + F_N} \dots\dots\dots (3)$$

- F1-score: F1-score gives an equilibrium between precision and recall by taking the harmonic mean of the two measures depends on: [24].

$$F - Measure = 2 * \frac{Precision*Recall}{Precision+Recall} \dots\dots\dots (4)$$

where:

TP = true positives TN = true negatives FP = false positives FN = false negatives

3.9 Technical Environment

The methodological implementation of this study was carried out entirely in Python, selected for its extensive ecosystem of scientific computing libraries. Model development, training, and evaluation were performed using widely adopted, industry-standard tools. Traditional machine learning algorithms including Support Vector Machines (SVM), Random Forest, and Logistic Regression were implemented using the Scikit-learn library. To incorporate advanced gradient boosting methods, the workflow was extended with XGBoost and LightGBM, both of which are recognized for their computational efficiency and strong predictive capabilities. Model optimization was achieved through systematic hyperparameter tuning using Grid Search in combination with Cross-Validation. This procedure evaluates all candidate hyperparameter configurations within a predefined search space, while the cross-validation mechanism provides a robust performance estimate to guide the selection of optimal parameters. The experimental analysis was conducted on the Cresci-2017 dataset, a comprehensive and publicly available benchmark hosted on GitHub. This dataset, which includes diverse bot classes alongside genuine user accounts, is widely acknowledged within the research community as a standard reference for bot detection studies, ensuring both comparability and credibility of the reported findings.

previous to model training, the raw data underwent a meticulous preprocessing pipeline to ensure data quality and feature robustness. The pipeline comprised of:

(1) a data cleaning stage involving the removal of entries with missing values and the elimination of duplicate records to mitigate potential bias.

(2) feature standardization, which rescales features to a mean of zero and a standard deviation of one.

This normalization is fundamental for ensuring the stable convergence and performance of scale-sensitive algorithms. Subsequently, a crucial step involved the engineering and extraction of a diverse set of features from the raw data; this included deriving static features from user profile metadata and crafting temporal features based on patterns of tweet activity, thereby enriching the informational content available to the models A stacking ensemble architecture was applied to optimize predictive performance and robustness. This approach combines the strengths of multiple, diverse base classifiers, with a first layer consisting of Random Forest, LightGBM, XGBoost, and SVM models.

The predictions generated by these base models were then used as input features for a second-level meta-learner, which was a Logistic Regression model tasked with learning the optimal way to combine these individual predictions to form a final, precis and stable classification. The evaluation of model performance was executed using a comprehensive set of metrics to ensure a holistic assessment. Model performance was evaluated using standard classification metrics: Accuracy, Precision, Recall, and F1-Score. as well, a confusion matrix was analyzed to obtain granular insights into the types of classification errors, distinguishing between false positives and false negatives. allowing for a thorough examination of the types of mistakes made by the models and facilitating a deeper understanding of their respective strengths and weaknesses.

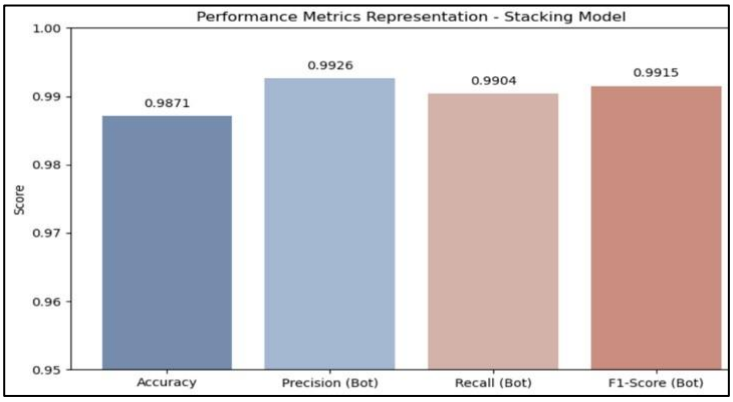
4. Results and Discussion

The proposed stacking ensemble model was evaluated against individual classifiers using the integrated static and temporal feature set. Table 2 shows the achievement metrics for each approach.

Table 2 - Achievement comparison of single models vs. stacking (static + temporal features)

Model	Accuracy	Precision	Recall	F1-score
Random Forest	0.9793	0.9850	0.9815	0.9832
LightGBM	0.9818	0.9874	0.9837	0.9855
XGBoost	0.9804	0.9862	0.9824	0.9843
Support Vector Machine	0.9725	0.9806	0.9761	0.9783
Stacking (Proposed)	0.9871	0.9926	0.9904	0.9915

This table presents the performance metrics of five distinct machine learning models for a classification task, which in this context is Twitter bot detection. Each model is evaluated based on four standard metrics: Accuracy, Precision, Recall, and F1-score. The scores, all falling between 0.97 and 1.0, indicate a very high level of performance across all models. Random Forest achieves an accuracy of 97.93%, meaning it correctly classified the bot/human status of accounts nearly 98 times out of 100. Its precision of 98.50% indicates that when it predicted an account was a bot, it was correct over 98.5% of the time. A recall of 98.15% shows it successfully identified 98.15% of all actual bots in the dataset. The F1score, which balances precision and recall, is 98.32%. LightGBM demonstrates a performance with an accuracy of 98.18%. Its precision is 98.74%, meaning its positive predictions were highly reliable. It achieved a recall of 98.37%, indicating a strong ability to find the vast majority of positive cases. The model's balanced performance is summarized by an F1-score of 98.55%. XGBoost shows results with an accuracy of 98.04%. It has a precision of 98.62%, reflecting a high rate of correct bot identifications. Its recall is 98.24%, demonstrating effectiveness in capturing most bot accounts. The harmonic mean of these two values is an F1-score of 98.43%. Support Vector Machine (SVM) performs with an accuracy of 97.25%. The model's precision is 98.06%, indicating that its bot predictions were correct most of the time.



It achieved a recall of 97.61%, meaning it located a large portion of all bots present. The F1-score for this model is 97.83%. Stacking (Proposed) is an ensemble model that combines the predictions of other models. It attained an accuracy of 98.71%. Its precision is 99.26%, and its recall is 99.04%. The F1-score, which harmonizes these two near-perfect metrics, is 99.15%. This demonstrates that integrating predictions from diverse classifiers allows the system to leverage their complementary strengths, leading to superior detection performance. Figure 3. shows the performance of the stacking model.

Fig.3 - The Performance of the Stacking Model Using Logistic Regression as Final Estimator

This figure provides a detailed visual and quantitative summary of the evaluation results for the proposed Stacking Model designed to identify automated bot accounts on Twitter. The data is presented using a bar chart format, which offers an immediate, at-a-glance comparison of the model's effectiveness across four fundamental performance indicators. These metrics are specifically calculated for the positive class, which in this detection task is the 'bot' category. The vertical axis of the chart, representing the performance score, is scaled from 0.95 to 1.00, a range chosen to highlight the model's near-perfect results and to allow for fine-grained comparison between the metrics. Each bar in the graph corresponds to one of the four key evaluation metrics, allowing viewers to easily discern the model's strengths. The chart reveals exceptionally high performance across all measures. An Accuracy of 0.9871 signifies that the model's overall prediction correctness is 98.71%, meaning it correctly determined the status (either bot or human) for the vast majority of accounts in the test dataset. The Precision score, reaching 0.9926, is a critical metric for this application. It indicates that the model's positive predictions are highly reliable; when it flags an account as a bot, it is correct 99.26% of the time. This high precision is crucial for minimizing false positives, which in a real-world scenario would involve mistakenly accusing legitimate human users of being bots. The Recall rate of 0.9904 demonstrates the model's comprehensive detection capability. It shows that the system is successful in identifying 99.04% of all actual bots present within the dataset. A high recall is essential for ensuring that very few bots escape detection. Finally, the F1-Score, which is the harmonic mean of Precision and Recall, stands at 0.9915. This consolidated metric confirms that the model does not achieve its high precision at the expense of recall, or vice versa. Instead, it demonstrates a robust, well-rounded, and balanced performance, excelling in both the correctness of its bot identifications and its completeness in finding all bots. This balance is vital for a deployment-ready bot detection system that must be both accurate and thorough.

- ***Confusion Matrix Analysis***

Figure 4 presents the confusion matrix for the proposed stacking model. Out of 2,874 test instances, the model correctly classified 679 genuine users and 2,158 bots. Only 16 cases were misclassified 6 false positives (legitimate users incorrectly flagged as bots) and 10 false negatives (bots incorrectly classified as humans). Examples of Misclassification:

- False Positives: These typically involved highly active legitimate accounts, such as news outlets, whose posting frequency and engagement patterns resembled automated behaviour.
- False Negatives: These often corresponded to bots with irregular posting schedules and diverse content types, effectively mimicking human patterns.

This analysis highlights that while the model is highly accurate, incorporating additional features such as linguistic patterns or network connectivity could further reduce misclassifications.

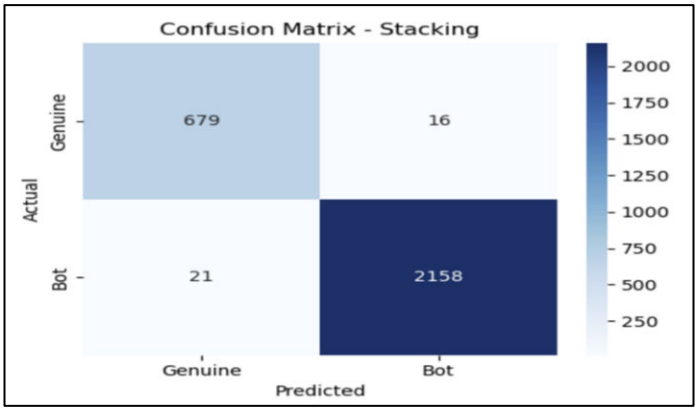


Fig.4 - The Confusion Matrix - Stacking Model

5. Conclusions and Future Work

This research suggested a supervised learning framework for Twitter bot detection that extract static and temporal features to get a complete picture of user behaviour. Using the Cresci-2017 dataset, Random Forest, LightGBM, XGBoost and SVM were combined as base learners in a stacking ensemble, with Logistic Regression as the meta-learner. The proposed model carry out 98.71% accuracy, outperforming all individual classifiers.

The results clear that gathering heterogeneous feature types within a stacked architecture enhances the adaptability and robustness of bot detection systems, enabling them to effectively identify sophisticated bots that mimic human behaviour. The confusion matrix analysis further revealed that misclassifications primarily occurred in cases where legitimate accounts exhibited bot-like activity or where bots closely imitated human posting patterns. To further improve detection performance and generalizability, the following directions are recommended: Network Dynamics Integration – Incorporate graph-based features such as retweet networks, follower/following clustering, and interaction graphs to capture relational patterns. Advanced Modelling Approaches – Explore transformer architectures for cross-platform temporal modelling, enabling better sequence understanding of posting behaviour. Graph Neural Networks (GNNs) – Apply GNN-based methods to leverage the structural information in user networks for improved detection. Explainable AI (XAI) – Integrate explainability techniques such as SHAP or LIME to provide transparent decision-making, which can build trust in automated systems. Real-time Detection – Develop low-latency implementations for real-time bot detection in live Twitter streams. By integrating richer feature sets and advanced modelling techniques, future bot detection frameworks can achieve higher accuracy, improved interpretability, and enhanced robustness against evolving bot strategies.

Acknowledgment

We sincerely thank the creators of the Cresci-2017 dataset for publicly sharing their comprehensive collection. This valuable resource was fundamental to our work, enabling the training, evaluation, and comparative analysis performed in this study. Furthermore, we extend our appreciation to the Department of Information Computer Systems, Faculty of Computer Science and Information Technology, Basrah.

References

[1] Mazza, M., Avvenuti, M., Cresci, S., and Tesconi, M., “Investigating the difference between trolls, social bots, and humans on Twitter", Computer Communications, vol. 199, (2022), pp: 111–126, https://www.sciencedirect.com/science/article/abs/pii/S0140366422003711?via %3Dihub.

[2] Aljabri, M., Zagrouba, R., Shaahid, A., Alnasser, F., Saleh, A., and Alomari, D. M., "Machine learning-based social media bot detection: a comprehensive literature review", Social Network Analysis and Mining, vol. 13, Article 20, (2023), https://www.researchgate.net/publication/366895683_Machine_learningbased_social_media_bot_detection_a_comprehensive_literature_review

[3] Rani, S., and Gill, N. S., "Hybrid Model for Twitter Data Sentiment Analysis Based on Ensemble of Dictionary Based Classifier and Stacked Machine Learning Classifiers—SVM, KNN and C5.0", Journal of Theoretical and Applied Information Technology, vol. 98, no. 04, (2020), PP: 610–619

https://www.researchgate.net/publication/345197750_Hybrid_model_for_twitter data_sentiment_analysis_based_on_ensemble_of_dictionary_based_classifier _and_stacked_machine_learning_classifiers-SVM_KNN_and_C50.

[4] Dimitriadis, I., Georgiou, K., & Vakali, A., "Social botomics: A systematic ensemble ML approach for explainable and multi-class bot detection" , Applied Sciences, 11(21), 9857, (2021), https://www.researchgate.net/publication/355502305_Social_Botomics_A_Systematic_Ensemble_ML_Approach_for_Explainable_and_MultiClass_Bot_Detection.

[5] Ren, Y., Wu, D., Singh, A., Kasson, E., Huang, M., and Cavazos-Rehg, P., "Automated Detection of Vaping-Related Tweets on Twitter During the 2019 EVALI Outbreak Using Machine Learning Classification", Frontiers in Big Data, vol. 5, Article 770585, (2022), https://www.frontiersin.org/journals/bigdata/articles/10.3389/fdata.2022.770585/full.

[6] Darem, A. A., Alhashmi, A. A., Alanazi, M. H., Alanezi, A. F., Said, Y., Darem, L. A., & Hussain, M. M., "Cybersecurity in social networks: An ensemble model for Twitter bot detection", International Journal of Advanced and Applied Sciences, 11(11), (2024), PP: 130–141, https://www.researchgate.net/publication/386064983_Cybersecurity_in_social_ networks_An_ensemble_model_for_Twitter_bot_detection.

[7] Kadhim, A., and Abdullah, A. M., "Fake accounts detection on social media using stack ensemble system", International Journal of Electrical and Computer Engineering (IJECE), vol. 12, no. 3, (2022), PP: 3013–3022, https://www.researchgate.net/publication/359602764_Fake_accounts_detection _on_social_media_using_stack_ensemble_system.

[8] Ng, L. H. X., & Carley, K. M., "Assembling a multi-platform ensemble social bot detector with applications to US 2020 elections" ,Social Network Analysis and Mining, 14(45), (2024), https://www.researchgate.net/publication/378464325_Assembling_a_multiplatform_ensemble_social_bot_detector_with_applications_to_US_2020_elections.

[9] Hannousse, A., and Talha, Z., "A Hybrid Ensemble Learning Approach for Detecting Bots on Twitter", International Journal of Performability Engineering, vol. 20, no. 10, (2024), PP: 610–620, https://www.researchgate.net/publication/391943243_A_Hybrid_Ensemble_Learning_Approach_for_Detecting_Bots_on_Twitter.

[10] Shi, S., Chen, J., Wang, Z., Zhang, Y., Zhang, Y., Fu, C., Qiao, K., and Yan, B., "SStackGNN: Graph Data Augmentation Simplified Stacking Graph Neural Network for Twitter Bot Detection", International Journal of Computational Intelligence Systems, vol. 17, Article 106, (2024), https://www.researchgate.net/publication/380196886_SStackGNN_Graph_Data_Augmentation_Simplified_Stacking_Graph_Neural_Network_for_Twitter_Bo t_Detection.

[11] Kouvela, M., "Bot Detective: Explainable Bot Detection in Twitter", Master's Thesis, Department of Data & Web Science, Aristotle University of Thessaloniki, Greece, (2020), https://ikee.lib.auth.gr/record/319595/files/GRI-202027499.pdf.

[12] Chikkasabbenahalli Venkatesh, S., Shaji, S. and Meenakshi Sundaram, B., "A Fake Profile Detection Model Using Multistage Stacked Ensemble Classification", Proceedings of Engineering and Technology Innovation, vol. x, no. x, (2024), https://www.researchgate.net/publication/378250893_A_Fake_Profile_Detectio n_Model_Using_Multistage_Stacked_Ensemble_Classification.

[13] Cresci, S. "Cresci-2017 Twitter Bot Dataset," GitHub, [Online]. Available: https://github.com/emanuelecrescini/cresci (2017).

[14] Han, J., Pei, J., and Kamber, M., Data Mining: Concepts and Techniques, Morgan Kaufman, (2011) https://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-MorganKaufmann-Series-in-Data-Management-Systems-Jiawei-Han-MichelineKamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-MorganKaufmann-2011.pdf.

[15] Zhou, Z.-H., "Ensemble Methods: Foundations and Algorithms", CRC Press, (2012), ISBN: 13:978-1-4398-3005-5, https://tjzhifei.github.io/links/EMFA.pdf.

[16] Breiman, L., "Random Forests", Machine Learning, vol. 45, no. 1, pp. 5–32, (2001), https://link.springer.com/article/10.1023/A:1010933404324.

[17] Cortes, C. and Vapnik, V., "Support-Vector Networks", Machine Learning, vol. 20, no.3, (1995), PP: 273–297, https://link.springer.com/article/10.1007/BF00994018.

[18] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.Y., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree", Advances in Neural Information Processing Systems (NeurIPS), vol. 30, (2017), PP: 3146–3154, https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde8486 69bdd9eb6b76fa-Paper.pdf.

[19] Tianqi Chen, T. and Guestrin, C., "XGBoost: A Scalable Tree Boosting System", ACM. ISBN 978-1-4503-4232-2/16/08, DOI: http://dx.doi.org/10.1145/2939672.2939785, PP: 785-794, (2016) https://github.com/dmlc/xgboost.

[20] Wolpert, D. H., "Stacked Generalization", Neural Networks, vol. 5, no. 2, (1992), PP: 241–259, https://www.researchgate.net/publication/222467943_Stacked_Generalization.

[21] Zheng, H. A Review of Evaluation Metrics in Machine Learning Algorithms. In Lecture notes in networks and systems, , (2023), PP: 15–25, https://doi.org/10.1007/978-3-031-35314-7_2

[22] Alarfaj, F. K., Ahmad, H., Khan, H. U., Alomair, A. M., Almusallam, N., and Ahmed, M., "Twitter Bot Detection Using Diverse Content Features and Applying Machine Learning Algorithms", Sustainability, vol. 15, no. 8, Article 6662, MDPI, (2023), https://www.researchgate.net/publication/370038291_Twitter_Bot_Detection_ Using_Diverse_Content_Features_and_Applying_Machine_Learning_Algorithms

[23] Dracewicz, W. and Sepczuk, M. "Detecting fake accounts on social media portals—the X portal case study," Electronics, vol. 13, no. 13, Art. no. 2542, Jul. (2024). DOI: 10.3390/electronics13132542.

[24] Geng, S. Analysis of the Different Statistical Metrics in Machine Learning. Highlights in Science Engineering and Technology, 88, 350–356, (2024) , https://doi.org/10.54097/jhq3tv19