An Ensemble Machine Learning Framework for Malicious PDF Detection Using Static and Structural Features

Ashaf Uddaula[1], Mahmudul Hasan[1,*], Dip Sarker[2], Nafisa Tasneem Esha[1], Md Sabbir Hosen Hamim[1], and Sadrul Amin[1]

[1] Department of Computer Science and Engineering, Daffodil International University, Dhaka-1216, Bangladesh[1]
[2] Department of Computer Science, American International University-Bangladesh, Dhaka, Bangladesh[2]
{ashafuddaula.official, esha23105101084, hamim2305101803, amin23105101322}@diu.edu.bd, 22-49304-3@student.aiub.edu, hasan23105101093@diu.edu.bd*
*Corresponding author

Abstract. Identifying malicious PDF files is crucial for cybersecurity since attackers are increasingly using the flexible structure and embedded content of PDFs to circumvent signature-based defenses. This work formulates a binary classification task based on interpretable machine learning on static structural and metadata indicators to distinguish between malicious and benign PDFs. A curated pipeline resolves mixed-type entries, cleans and harmonizes fields, and maintains 19 features (such as stream markers, encryption flags, cross-reference table length, and the presence of action/scripts like JS, Javascript, and OpenAction). On a corpus of 10,026 labeled PDFs (CIC Evasive PDFMal2022), the study assesses Gaussian Naive Bayes, Decision Tree, Random Forest, and Logistic Regression. Following preprocessing, 9,708 usable samples are obtained. Experiments using stratified splits and Randomized Hyperparameter Tuning reveal that the optimized Random Forest achieves **99.48%** test accuracy, surpassing both classical baselines and a Deep Learning (MLP) benchmark (98.76%). To address transparency, SHAP (SHapley Additive exPlanations) analysis is integrated, confirming that structural features and JavaScript presence drive detection logic. The proposed framework offers a repeatable, highly accurate, and explainable solution suitable for operational PDF triage.

Keywords: PDF malware detection, Static features, Random Forest, Ensemble learning, Explainable AI

1 Introduction

Portable Document Format (PDF) files are widely used for reliable cross-platform document exchange, but their flexible structure and embedded content (for ex-

M. S. Arefin et al. (eds.), *Proceedings of the International Conference on Intelligent Data Analysis and Applications (IDAA 2025)*, Advances in Intelligent Systems Research 206,
https://doi.org/10.2991/978-94-6239-664-7_62

ample, JavaScript, launch actions, and embedded objects) make them attractive vectors for malware that can evade traditional signature and heuristic defenses [4]. Static rules often fail on previously unseen or obfuscated samples, which motivates learning-based detection that captures discriminative structural patterns beyond known indicators [1].

Prior work shows that learning on static, structure-oriented features such as cross-reference table properties, encryption flags, and action/script presence can achieve high accuracy while remaining efficient for pre-execution filtering and triage in operational pipelines [3, 6].

This study addresses the binary classification problem of distinguishing malicious from benign PDFs using interpretable machine learning on static and structural indicators that can be extracted without executing the file content [8]. The dataset is CIC Evasive PDFMal2022, a corpus of 10,025 labeled records curated from multiple sources and refined via clustering to emphasize difficult, near-benign samples, thereby stressing common learning algorithms and better reflecting evasive behaviors in the wild [7].

The proposed framework ingests CIC Evasive PDFMal2022, harmonizes mixed fields, resolves ambiguous textual indicators, and retains a compact set of 19 stable features. Classical models—Gaussian Naive Bayes, Decision Tree, Random Forest, and Logistic Regression—are trained with stratified splits and evaluated using confusion matrices and ROC curves [5]. This choice balances accuracy with interpretability and operational clarity, consistent with findings that tree ensembles on static indicators yield robust detection on evasive PDFs while supporting transparent investigation workflows [2, 10].

The key contributions of this research are summarized as follows:

- A transparent static feature pipeline for evasive PDF classification with clear preprocessing and feature selection, ensuring reproducibility.
- Comparative evaluation of classical models showing that ensemble learning on compact features achieves competitive performance while maintaining interpretability.
- Contextualization of results with emphasis on dataset design and evaluation rigor, highlighting the importance of representativeness and feature choice.

2 Related Work

Static learning on structural and metadata features has long been recognized as an effective pre-execution technique for triage and filtering in PDF malware detection [10]. This is primarily because obfuscation, mimicry, and evasive content embedding frequently defeat signature-based and heuristic defenses. Unlike dynamic analysis, which requires execution of the file content and can be resource-intensive, static structural analysis provides a lightweight approach suitable for real-time filtering pipelines. Early studies demonstrated that static ensembles, particularly tree-based classifiers, trained on metadata and document structure indicators, can achieve high accuracy with relatively low false positive rates [2].

A critical dimension in this area is dataset quality and representativeness [3]. Studies emphasize that evaluation protocols and dataset design materially affect reported accuracy and generalization across different deployment scenarios [9]. Issakhani et al. [4] introduced the CIC Evasive PDFMal2022 dataset and evaluated a stacking approach. Smutz and Stavrou [10] demonstrated the efficacy of Random Forests achieving over 99% accuracy. More recently, Yerima et al. [11] advanced explainable ensemble methods.

Table 1 summarizes prior works. Notably, our work focuses on balancing high accuracy with interpretability using a compact feature set on a modern evasive dataset.

Table 1. Summary of prior work on PDF malware detection

Reference	Dataset(s)	Method	Key Results
Issakhani et al. (2022) [1]	Contagio; CIC Evasive	Stacking ML (Static)	High Acc/F1 on Contagio
Smutz & Stavrou (2012) [9]	100k+ benign, 5k+ mal	Random Forest	>99% Acc, ∼0.2% FPR
Yerima et al. (2023) [10]	EvasivePDFMal2022	Ensemble	>98% Acc; 100% detection
Lin & Nicholas (2023) [12]	Multiple PDF sets	Small feature set	High Acc with few features
This work (2025)	**CIC Evasive (9,708)**	**Optimized RF**	**RF 99.48% Acc; SHAP XAI**

3 Materials and Methods

The framework builds a static, interpretable pipeline for classifying malicious versus benign PDFs using structural and metadata indicators from the CIC Evasive PDFMal2022 dataset. Figure 1 illustrates the workflow.

3.1 Dataset

The dataset used is CIC Evasive PDFMal2022 [4]. To ensure robustness against evolving threats, this dataset was selected for its diversity in representing modern evasion techniques. Unlike older datasets (e.g., Contagio), this corpus includes over 10,000 samples featuring varied attack patterns such as JavaScript obfuscation, structural anomalies, and embedded actions. After preprocessing, 9,708 samples remain.

3.2 Data Preprocessing

Preprocessing begins with cleaning and type harmonization. Rows with null entries are discarded, ambiguous entries in the `text` field (e.g., "-1" or "0") are mapped to categorical values, and mixed-type indicators are cast to integers. Object and binary indicators are label-encoded. Non-predictive and noisy

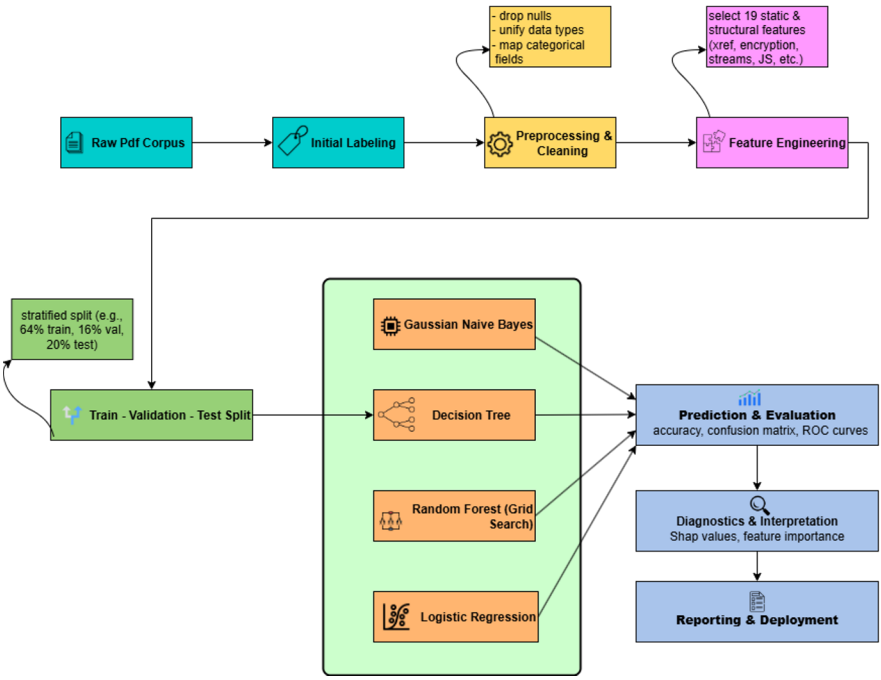

Fig. 1. Workflow of the proposed PDF malware detection pipeline.

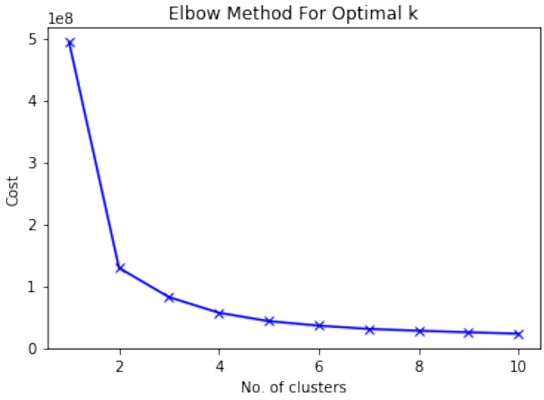

Fig. 2. Illustration of the Elbow method used in dataset curation (k vs. SSE).

columns are removed, leaving a compact set of 19 structural features. Hyperparameter tuning was conducted using **RandomizedSearchCV** to explore a broader search space efficiently.

3.3 Machine Learning Models

Four classical models are evaluated: Gaussian Naive Bayes, Decision Tree, Random Forest, and Logistic Regression.To understand the classification logic, we briefly outline the mathematical foundations of the primary algorithms used in this framework.

Random Forest Algorithm Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by the majority voting of individual trees. Given a training set $X = \{x_1, \ldots, x_n\}$ with responses $Y = \{y_1, \ldots, y_n\}$, the algorithm repeatedly selects a random sample with replacement (bagging) and fits trees to these samples.

The prediction $\hat{y}$ for a new instance x' is obtained by aggregating the predictions of K individual trees $h_k(x')$:

$$\hat{y} = \text{majority\_vote}\{h_1(x'), h_2(x'), \ldots, h_K(x')\} \tag{1}$$

This bagging technique reduces variance and prevents overfitting, which is crucial for our high-dimensional feature space containing sparse structural data.

SHAP (SHapley Additive exPlanations) SHAP values provide a unified measure of feature importance based on cooperative game theory. It assigns each feature an importance value for a particular prediction. The SHAP value ϕ_i for a feature i is calculated as the weighted average of marginal contributions:

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f_x(S \cup \{i\}) - f_x(S)] \tag{2}$$

where F is the set of all features, S is a subset of features excluding i, and $f_x(S)$ is the prediction model trained only on feature subset S. This formulation allows us to mathematically quantify the contribution of specific features, such as `Javascript` or `OpenAction`, in classifying a PDF as malicious. Random Forest is the primary model, designed to capture non-linear interactions among structural flags.

4 Results and Discussion

4.1 Experimental Setup

After preprocessing and feature selection, 9,708 samples are available for modeling with labels Malicious (5,555) and Benign (4,468). Experiments use a stratified 64/16/20 train/validation/test split. All models are trained on the training

Table 2. Detailed description of static features extracted for detection

Feature Name	Description & Significance
`pdfsize`	Total size of the PDF file (KB). Malware often has anomalous file sizes compared to benign documents.
`metadata size`	Size of the metadata stream. Attackers frequently inject malicious scripts or shellcodes here.
`pages`	Number of pages. Malicious PDFs often contain only a single page to deliver the payload quickly.
`isEncrypted`	Boolean flag indicating encryption. Encrypted files hide content from simple scanners.
`embedded files`	Count of files embedded within the PDF. This is a primary vector for dropping executable malware.
`images`	Number of image objects. Malformed images (e.g., buffer overflow exploits) are common attack vectors.
`text`	Presence of text streams. Lack of text may indicate a wrapper file for malware.
`header`	Validity of the PDF header signature. Corrupt headers can crash parsers.
`obj`	Total count of objects. A mismatch with `endobj` indicates structural manipulation.
`endobj`	Count of end-object markers. Essential for structural integrity checks.
`stream`	Count of stream objects. High stream counts can hide heavily obfuscated code.
`endstream`	Count of end-stream markers. Used to detect broken or malformed streams.
`xref`	Presence of the Cross-Reference table. Manipulation here hides malicious objects from readers.
`trailer`	Presence of the trailer dictionary, which points to the root object.
`startxref`	Byte offset of the xref table. Anomalies here are typical of evasive malware.
`Javascript`	Count of `Javascript` tags. The most common vector for heap spraying and drive-by downloads.
`JS`	Abbreviated tag for Javascript. Both forms must be checked to avoid evasion.
`OpenAction`	Flag for actions triggered automatically upon opening. Highly suspicious in unknown files.
`Acroform`	Presence of interactive forms. Often used to conceal malicious XFA (XML Forms Architecture) scripts.
`JBIG2Decode`	Filter for image compression. Exploits targeting this filter are common (e.g., zero-click attacks).
`RichMedia`	Presence of embedded Flash or video content, which can host vulnerabilities.
`launch`	Command to launch external applications. A critical security risk if found.

fold, validation is used for model selection and diagnostics, and the test fold is reserved for final accuracy.

Hyperparameter tuning was conducted using **RandomizedSearchCV** to explore a broader search space efficiently. We optimized critical parameters for the Random Forest model, including the number of estimators (100, 200, 300), maximum depth (None, 10, 20, 30), and bootstrap settings. This ensures the model avoids overfitting while maximizing generalized performance. Additional ROC and confusion matrix analyses on the validation set are included to compare separability and error patterns.

4.2 Test Performance

Tree ensembles demonstrate distinct advantages on compact, static structural indicators. By applying Randomized Hyperparameter Tuning to optimize the Random Forest model, performance improved significantly.

As shown in Table 3, the optimized Random Forest achieved the highest test accuracy of **99.48%**, surpassing the Deep Learning benchmark (MLP) which reached 98.76%. This confirms that for tabular static features, ensemble methods can outperform complex neural networks.

Table 3. Performance Comparison of Proposed Framework vs. Baselines

Model	Acc (%)	Precision	Recall	F1
Random Forest (Opt.)	**99.48**	**1.00**	**0.99**	**0.99**
Deep Learning (MLP)	98.76	0.99	0.99	0.99
Decision Tree	97.79	0.98	0.98	0.98
Gaussian NB	88.11	0.89	0.81	0.85
Logistic Regression	88.05	0.88	0.82	0.85

4.3 Validation Diagnostics

Validation confusion matrices for Decision Tree and Random Forest are nearly perfect, with minimal false positives and false negatives. As seen in Fig. 3, the Random Forest model demonstrates strong class separation. Furthermore, ROC curves (Fig. 4) show the highest separability (AUC = 1.00) for tree-based models.

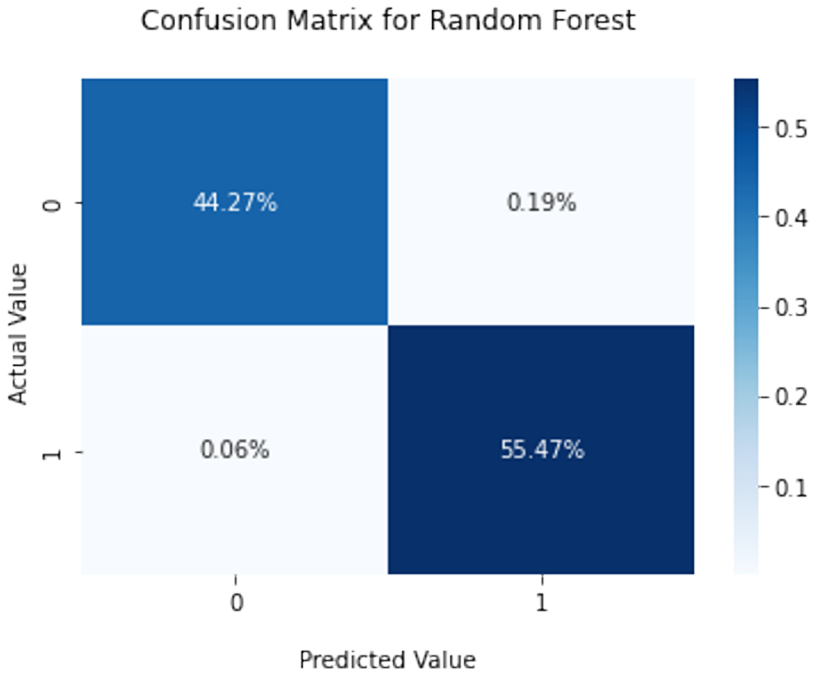

Fig. 3. Validation confusion matrix for Random Forest.

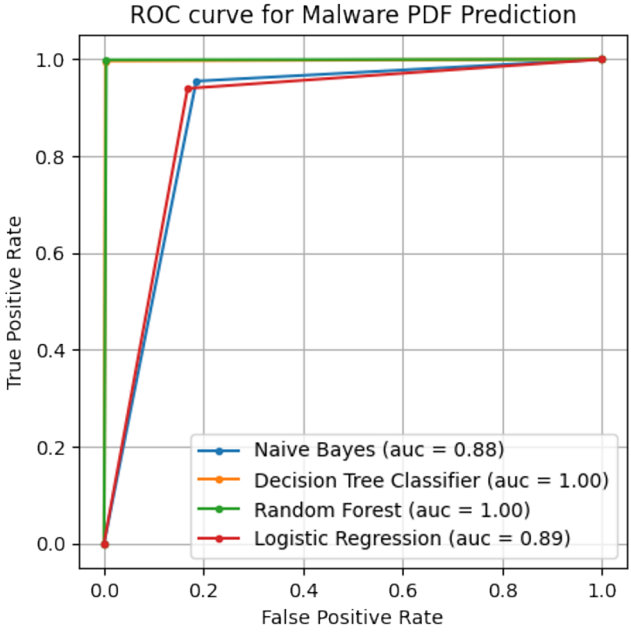

Fig. 4. ROC curves for all models (GNB, DT, RF, LR).

4.4 Comparative Analysis with Deep Learning

To evaluate robustness, we benchmarked against a Deep Learning model (Multi-Layer Perceptron - MLP) constructed with two hidden layers (128 and 64 neurons). As shown in Table 4, our optimized Random Forest ensemble achieved an accuracy of 99.48%, outperforming the MLP model (98.76%).

Table 4. Comparison with Deep Learning Benchmark

Model Architecture	Accuracy (%)
Proposed Optimized Random Forest	**99.48**
Deep Learning Benchmark (MLP)	98.76

4.5 Model Interpretability using SHAP

To ensure transparency and trust in our ensemble framework, we employed SHAP (SHapley Additive exPlanations) to interpret the model's decision-making process. As illustrated in Fig. 5 (SHAP Summary Plot) and Fig. 6 (Feature Importance Bar Plot), the analysis clarifies which features contribute most to the classification of malicious PDFs.

The results clearly indicate that `Javascript` and `JS` are the most dominant features. This aligns with domain knowledge, as attackers frequently embed malicious code within JavaScript objects in PDF files. Additionally, structural features like `metadata size` and `startxref` also play a significant role, confirming that our model effectively detects structural anomalies typical of evasive malware. This interpretability confirms that the high accuracy (99.48%) is driven by relevant security features rather than dataset bias.

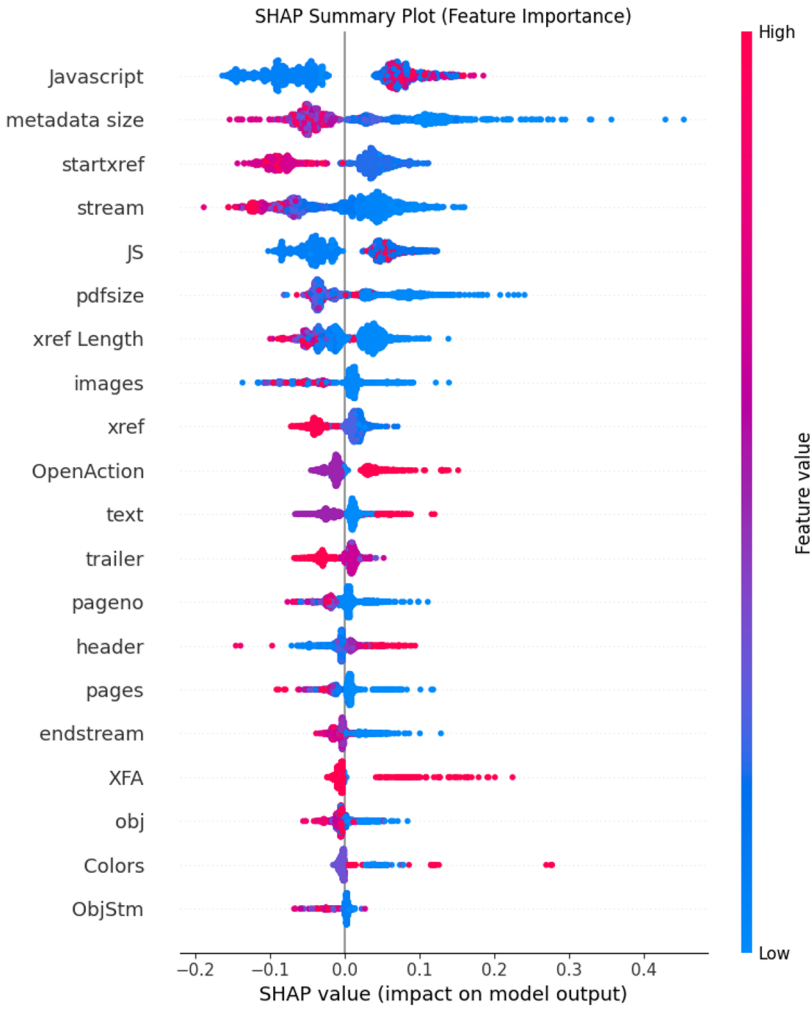

Fig. 5. SHAP Summary Plot showing the impact of features on model output.

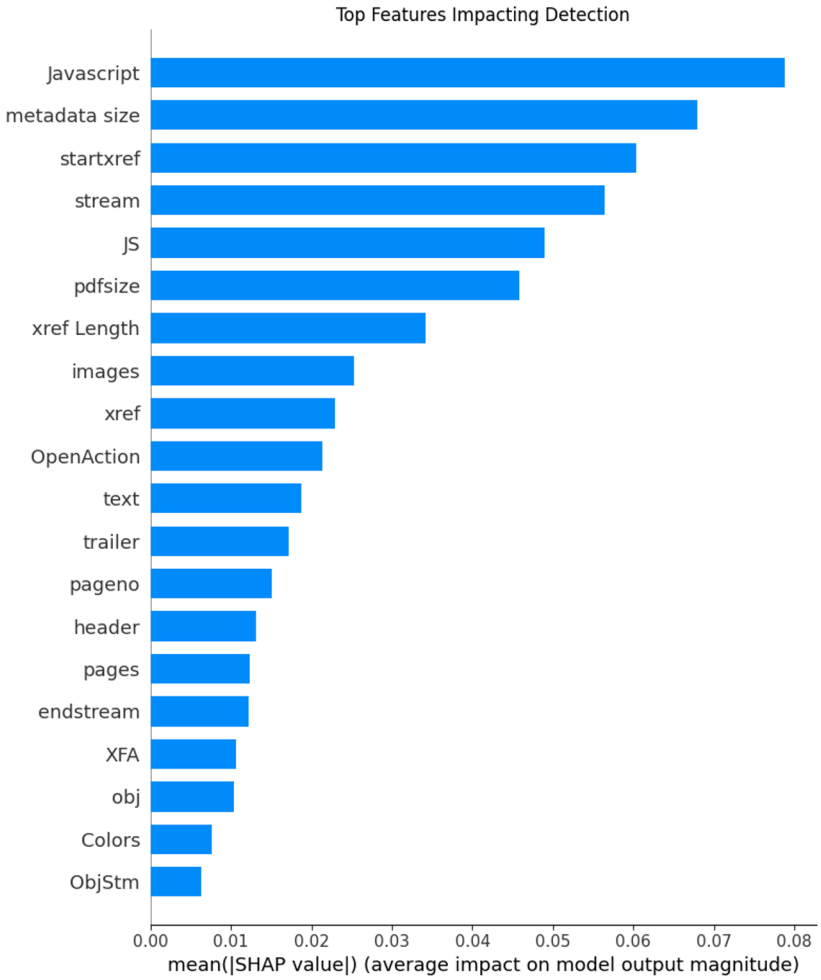

Fig. 6. Top Features Impacting Detection (Bar Chart).

5 Limitations and Future Work

This study demonstrates exceptional accuracy using static features. However, purely static analysis may miss sophisticated attacks that trigger only at runtime; future work will incorporate lightweight dynamic analysis. Additionally, broader cross-dataset validation (e.g., on Contagio) is required to ensure generalization against zero-day threats.

6 Conclusion

This work presented an interpretable static feature framework for detecting evasive malicious PDFs. By applying transparent preprocessing and Randomized Hyperparameter Tuning, the optimized Random Forest model achieved 99.48% accuracy, outperforming both classical baselines and a Deep Learning benchmark (MLP). SHAP analysis confirmed that the model relies on robust security features. The pipeline offers a balance of accuracy and interpretability, making it highly suitable for real-world cybersecurity deployment.

References

1. Bacci, A., Bartoli, A., Martinelli, F., Medvet, E., Mercaldo, F., Visaggio, C.A.: Impact of code obfuscation on android malware detection based on static and dynamic analysis. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), pp. 379–385. SCITEPRESS (2018)
2. Borno, Z.S., Sakib, N., Anwar, S.S.: Performance analysis of ensemble machine learning algorithms in PDF malware detection. In: 2023 IEEE 9th International Women in Engineering (WIE) Conference, pp. 195–200. IEEE (2023)
3. Hulkund, N., et al.: DataS3: Dataset subset selection for specialization. arXiv preprint arXiv:2504.16277 (2025)
4. Issakhani, M., Victor, P., Tekeoglu, A., Lashkari, A.: PDF malware detection based on stacking learning. In: Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP), pp. 562–570. SCITEPRESS (2022)
5. Kumar, A., Sharma, I.: SGWeS: A framework to safeguard web servers from PDF malware attacks. In: 2023 2nd International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN), pp. 1–6. IEEE (2023)
6. Laurenza, G., Aniello, L., Lazzeretti, R., Baldoni, R.: Malware triage based on static features and public APT reports. In: Cyber Security Cryptography and Machine Learning (CSCML 2017), *Lecture Notes in Computer Science*, vol. 10332, pp. 288–305. Springer, Cham (2017)
7. Nandinee, N., Tomar, D.S., Sharma, Y.K., Dehalwar, V.: Malware-BERT: Enhancing evasive malware detection with multi-head BERT attention. In: 2025 3rd International Conference on Disruptive Technologies (ICDT), pp. 962–967. IEEE (2025)
8. Rajagopal, S., Gaur, A., Vinod, P.: Interpretable PDF malware detector. In: 2023 16th International Conference on Security of Information and Networks (SIN), pp. 1–6. IEEE (2023)
9. Rattanaphan, S., Briassouli, A.: Evaluating generalization, bias, and fairness in deep learning for metal surface defect detection. Processes **12**(3), 456 (2024)
10. Smutz, C., Stavrou, A.: Malicious PDF detection using metadata and structural features. In: Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC), pp. 239–248. ACM (2012)
11. Yerima, S.Y., Bashar, A.: Explainable ensemble learning based detection of evasive malicious PDF documents. Electronics **12**(14), 3148 (2023)