Summary of the Notebook: bot-detection-prototype-notebook.ipynb

This Jupyter Notebook presents a comprehensive, data-driven prototype for a bot detection system. It systematically evaluates four distinct methodologies on a web server access log dataset to identify the most effective approach for real-world deployment.

Notebook Structure and Workflow:

1. **Step 1 (Log Parsing and Exploratory Data Analysis):**
    - **Data Processing and EDA:** The notebook begins by parsing 10,000 raw log entries into a structured format. An initial Exploratory Data Analysis (EDA) is conducted to understand the dataset's characteristics, identifying traffic volume, top IP addresses, and the prevalence of known web crawlers.
    - **Advanced Feature Engineering:** A robust feature engineering pipeline is implemented to create **distinct behavioral features** for each of the 334 unique IP addresses that made multiple requests. To test algorithm performance under different conditions, two feature sets are generated:
        1. **Basic Features (8):** A lightweight set of core metrics like request count, rate, and error ratios.
        2. **Advanced Features (25):** A comprehensive set capturing more complex temporal and behavioral patterns.
2. **Step 2 (Algorithm Implementation and Evaluation):** Four different bot detection algorithms are implemented and rigorously tested:
    - **Rule-Based System:** A heuristic-based detector is built first to serve as an interpretable benchmark and "ground truth" for the unsupervised models. It flags **14 IPs (4.2%)** as bots.
    - **Isolation Forest:** A standard tree-based anomaly detection algorithm is trained on both basic and advanced feature sets.
    - **XStream:** The streaming anomaly detection algorithm from the research paper is implemented, revealing a critical performance dependency on the feature set.
    - **SSGAD (Streaming Sequence-Aware Graph Anomaly Detection):** This graph-based approach is prototyped in three modes: a fast feature-based mode (using basic and advanced features) and a more computationally intensive full graph-based mode.
3. **Final Step (Comparative Analysis and Conclusion):** All algorithm configurations are ranked based on their **F1-score**, precision, and recall. The notebook concludes by providing a final ranking and a strategic recommendation for a production-level bot detection system.

Short Explanation of Results:

The analysis produced a clear and decisive ranking of the bot detection algorithms. The results not only identified the best-performing model but also revealed crucial insights into how each algorithm interacts with different feature sets.

Final Algorithm Performance Ranking

The models were ranked by their **F1-score**, which provides the best measure of overall effectiveness by balancing precision and recall. **XStream, when used with the basic feature set, was the undisputed champion**, significantly outperforming all other configurations.

Rank	Algorithm	Features/Mode	F1-Score	Recall	Precision
🥇 **1st**	**XStream**	**Basic**	**0.7778**	1.0000	0.6364
🥈 **2nd**	**SSGAD**	**Feature (Basic)**	0.6829	1.0000	0.5185
🥉 **3rd**	**Isolation Forest**	**Basic**	0.6512	1.0000	0.4828
4th	SSGAD	Graph	0.6222	1.0000	0.4516
5th	SSGAD	Feature (Advanced)	0.6000	0.8571	0.4615
6th	Isolation Forest	Advanced	0.5217	0.8571	0.3750
7th	XStream	Advanced	0.0805	1.0000	0.0419

Key Finding 1: XStream's Dominance with a Curated Feature Set

XStream demonstrated outstanding performance when paired with the smaller, curated set of **8 basic features**. It achieved a **perfect recall of 100%**, successfully identifying every bot flagged by the rule-based system. More importantly, it also achieved the **highest precision (64%)** of any model, making it both the most effective and the most efficient choice.

Key Finding 2: The Critical Importance of Simplicity ("Less is More")

The most significant finding was that for all three algorithms, the **basic feature set consistently outperformed the advanced feature set**. This was especially dramatic for XStream, whose performance catastrophically collapsed with more features, indicating a high sensitivity to the "curse of dimensionality."

This highlights a crucial lesson: for these unsupervised models, **feature quality and relevance are far more important than feature quantity.**

Key Finding 3: Perfect Recall is Achievable

The top three performing models, **XStream (Basic)**, **SSGAD (Basic)**, and **Isolation Forest (Basic)**, all achieved **100% recall**. This means they are all highly capable of identifying the full scope of bot activity present in the dataset. The primary differentiator among them becomes precision, or how many false positives they generate. **XStream (Basic)** sits in the most desirable position, combining perfect recall with the highest precision.

Final Recommendation

Based on this comprehensive analysis, a hybrid, multi-layered strategy is recommended for production:

1. **Primary System:** Use **XStream with a basic, carefully selected feature set** for optimal, high-efficiency bot detection.
2. **High-Efficiency Alternative:** The **SSGAD Feature Mode (Basic)** serves as an excellent and computationally fast alternative, also providing perfect recall with strong precision.
3. **Avoid Complexity:** Avoid using large, uncurated feature sets, as they were shown to degrade the performance of every tested algorithm.