Bot Detection Task: Identifying Automated vs. Human User Behavior

Background

You are working for an e-commerce company that has been experiencing suspicious traffic patterns. The security team suspects that automated bots are accessing the website, potentially for price scraping, inventory monitoring, or fraudulent activities. You need to build a machine learning model to distinguish between legitimate human users and automated bot traffic.

Task Overview

Build a bot detection pipeline that analyzes user behavior patterns and classifies sessions as either **human users** or **automated bots** . ## Dataset Description

Web Session Data (`user_sessions.csv`)

Each row represents a user session with the following features:

Session Identifiers: - `session_id`: Unique session identifier - `user_agent`: Browser/device information string - `ip_address`: User's IP address (anonymized)

Behavioral Metrics: - `page_views`: Number of pages viewed in session - `session_duration`: Total time spent (seconds) - `avg_time_per_page`: Average time spent per page (seconds) - `bounce_rate`: Percentage of single-page sessions - `scroll_depth`: Average scroll depth percentage - `click_count`: Total number of clicks - `form_interactions`: Number of form fields interacted with - `search_queries`: Number of search queries made

Technical Patterns: - `request_frequency`: Requests per minute - `javascript_enabled`: Whether JavaScript is enabled (0/1) - `cookie_enabled`: Whether cookies are enabled (0/1) - `screen_resolution`: Screen resolution (e.g., "1920x1080") - `browser_language`: Browser language setting - `referer_type`: How user arrived (direct/search/social/referral)

Timing Patterns: - `hour_of_day`: Hour when session started (0-23) - `day_of_week`: Day of week (1-7) - `time_between_clicks`: Average time between clicks (milliseconds) - `page_load_time`: Average page load time (milliseconds)

Navigation Patterns: - `unique_pages_visited`: Number of unique pages - `sequential_page_views`: Pages viewed in sequence vs. jumping - `back_button_usage`: Number of times back button used - `tab_switches`: Number of tab switches detected - `copy_paste_events`: Number of copy-paste actions

Target Variable: - `is_bot`: 1 if bot, 0 if human user

Requirements

Data Exploration & Pattern Analysis

1. **Basic Data Profiling**
   - Load data and examine structure (~20K sessions)
   - Check data types, missing values, and basic statistics
   - Analyze distribution of bot vs. human labels
2. **Behavioral Pattern Analysis**
   - Compare bot vs. human behavior distributions
   - Identify suspicious patterns (e.g., very fast navigation, uniform timing)
   - Visualize key behavioral differences
   - Analyze user agent strings for bot signatures

Feature Engineering & Data Preprocessing

1. **Feature Engineering**
   - **Speed indicators**: clicks_per_minute, pages_per_minute
   - **Consistency metrics**: std_dev of time_between_clicks
   - **Human-like behavior**: scroll_interaction_ratio, form_completion_rate
   - **User agent parsing**: browser_type, is_mobile, os_type
   - **IP analysis**: requests_from_same_ip (if multiple sessions)
   - **Temporal patterns**: is_business_hours, is_weekend
2. **Data Preprocessing**
   - Handle missing values appropriately
   - Encode categorical variables (user_agent, browser_language, etc.)
   - Scale numerical features
   - Handle class imbalance if present

Model Development

1. **Baseline Models**
   - Logistic Regression with key behavioral features
   - Decision Tree for interpretability
2. **Advanced Models**
   - Random Forest (handles mixed data types well)
   - XGBoost or LightGBM for performance
   - Isolation Forest for anomaly detection approach
3. **Feature Selection**
   - Identify most discriminative features
   - Remove redundant or noisy features

Evaluation & Deployment Prep

1. **Model Evaluation**
   - Focus on **Precision** (minimize false positives - flagging humans as bots)

- Analyze **Recall** for bot detection
- Generate confusion matrix and classification report
- Feature importance analysis

2. **Real-time Scoring Preparation**
   - Create simple scoring function
   - Define risk thresholds (low/medium/high risk)
   - Document key behavioral red flags

Deliverables

1. Jupyter Notebook

- Complete data exploration with visualizations
- Feature engineering pipeline
- Model training and evaluation code
- Results analysis and interpretation

2. Trained Model

- Best performing model file (pickle/joblib)
- Feature preprocessing pipeline
- Simple prediction function

3. Bot Detection Report (1-2 pages)

- **Key Behavioral Patterns**: Top differences between bots and humans
- **Model Performance**: Accuracy, precision, recall metrics
- **Critical Features**: Top 10 features for bot detection
- **Risk Scoring**: How to interpret model scores
- **False Positive Analysis**: When humans might be flagged as bots

4. Detection Rules Summary

- **High-Risk Indicators**: Immediate bot flags
- **Suspicious Patterns**: Require further investigation
- **Whitelist Criteria**: Definitely human behavior

Expected Bot Behavior Patterns

Based on typical bot characteristics, look for:

Speed Anomalies

- Extremely fast page navigation (< 2 seconds per page)
- Uniform timing between actions
- High request frequency

Interaction Patterns

- No scrolling or clicking behavior
- Skipping JavaScript-heavy pages
- No form interactions despite visiting forms

Technical Signatures

- Disabled JavaScript or cookies
- Unusual user agent strings
- Outdated browser versions
- Non-standard screen resolutions

Navigation Anomalies

- Sequential page crawling patterns
- No use of back button or random navigation
- Accessing robots.txt or sitemap files

Success Metrics

Model Performance

- **Accuracy**: $> 85\%$
- **Precision**: $> 80\%$ (minimize false positives)
- **Recall**: $> 75\%$ (catch most bots)
- **F1-Score**: $> 77\%$

Technical Specifications

Tools Required

- Python: pandas, numpy, scikit-learn
- Visualization: matplotlib, seaborn
- Optional: xgboost, user-agents library

Data Size

- ~20,000 user sessions
- ~25 features per session
- Balanced dataset (40% bots, 60% humans)

Common Challenges to Address

1. **False Positives**: Power users or users with disabilities might exhibit bot-like patterns
2. **Sophisticated Bots**: Modern bots that mimic human behavior more closely

3. **Data Quality**: Inconsistent user agent strings or missing behavioral data
4. **Real-time Performance**: Model must score quickly for live traffic

Bonus Challenges (If Time Permits)

1. **User Agent Analysis**: Parse and categorize user agent strings for bot signatures
2. **IP Reputation**: Analyze IP addresses for known bot networks
3. **Time Series Analysis**: Look for periodic or scheduled bot activity patterns
4. **Ensemble Method**: Combine multiple detection approaches