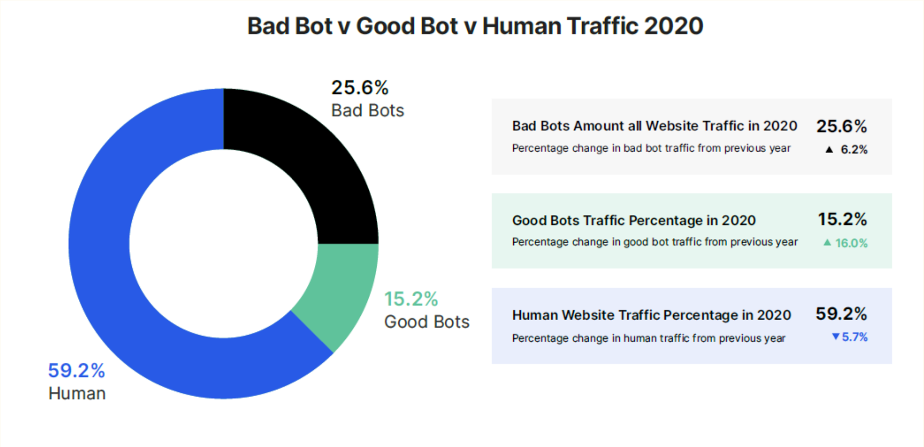
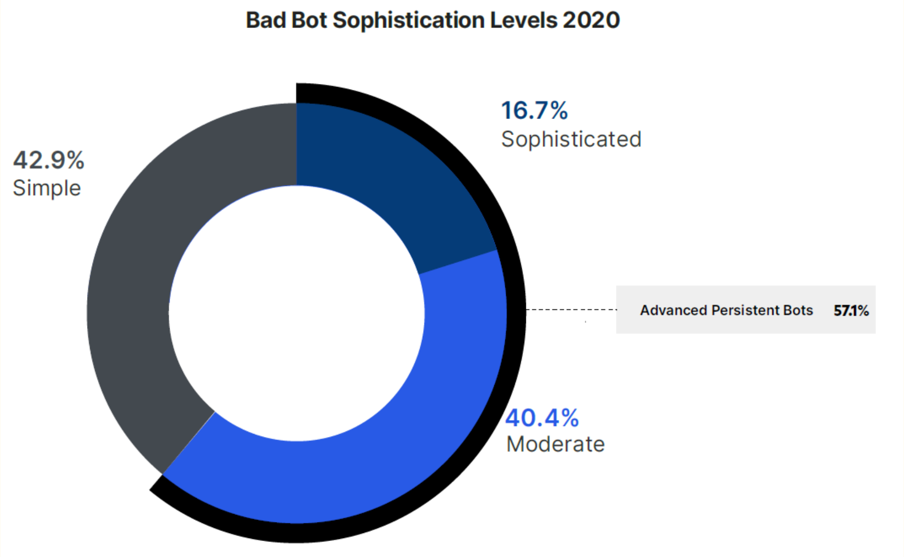
An Efficient Mouse Man-Machine Recognition Method Based On De-redundancy Of Mouse Sliding Trajectory Feature

Xiaofeng Lu, Beijing University of Posts and Telecommunication
Zhenhan Feng, Beijing University of Posts and Telecommunication

Imperva’s Bad Bot Report 2021

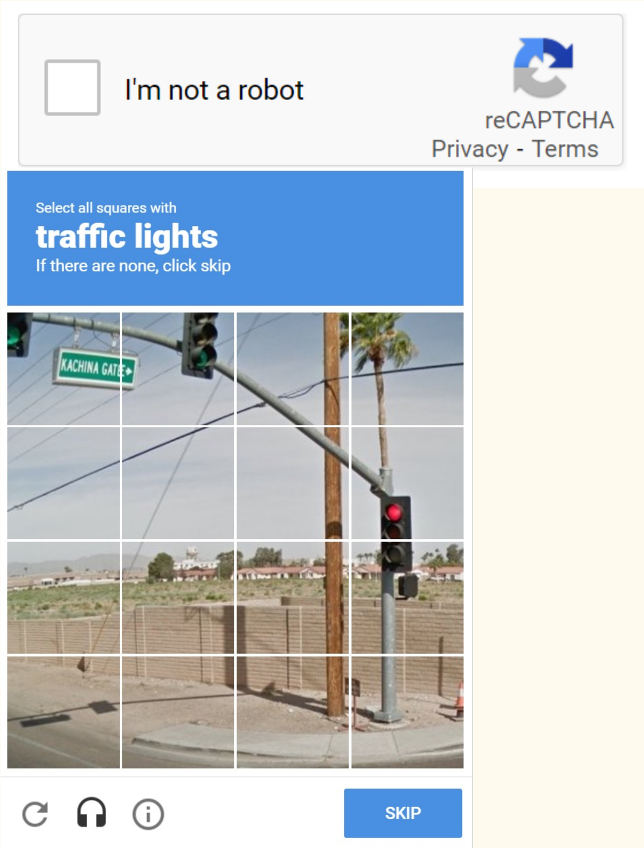




In 2020, the traffic of bad robots will maintain an upward trend.In 40.8% of the robot traffic, 25.6% of the robot traffic is malicious behavior, and only 15.2% of the robot traffic is normal traffic such as search engines.

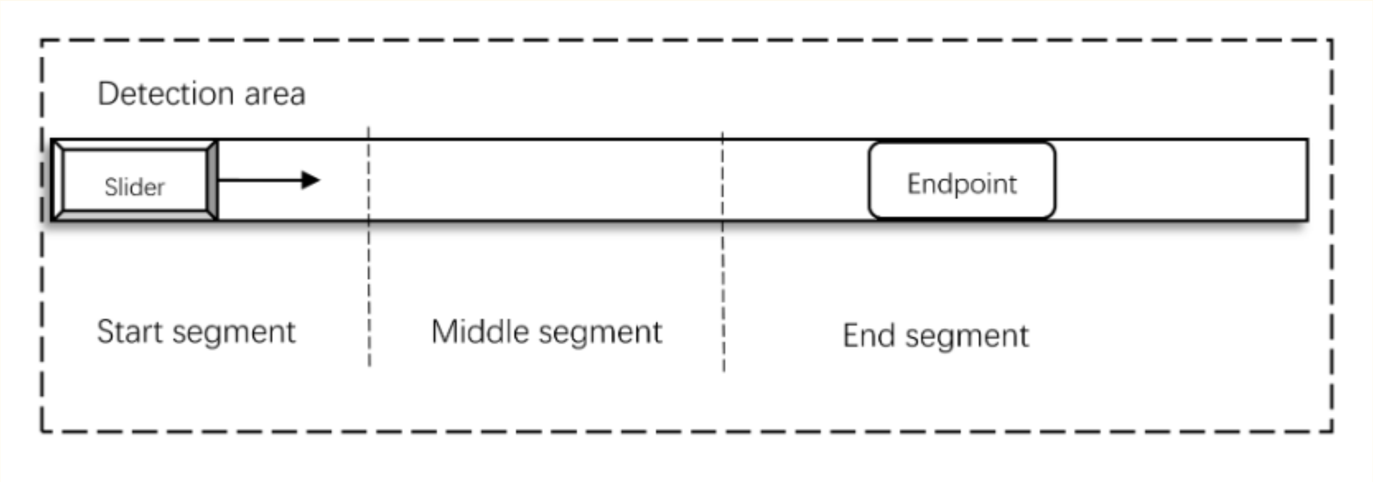
Advanced persistent robots (APBs) still account for the main part of bad robot traffic and are more difficult to detect and mitigate.

Google reCAPTCHA



- Google reCAPTCHA
- Poor user experience
- Hard to distinguish

Example diagram of slider verification codes

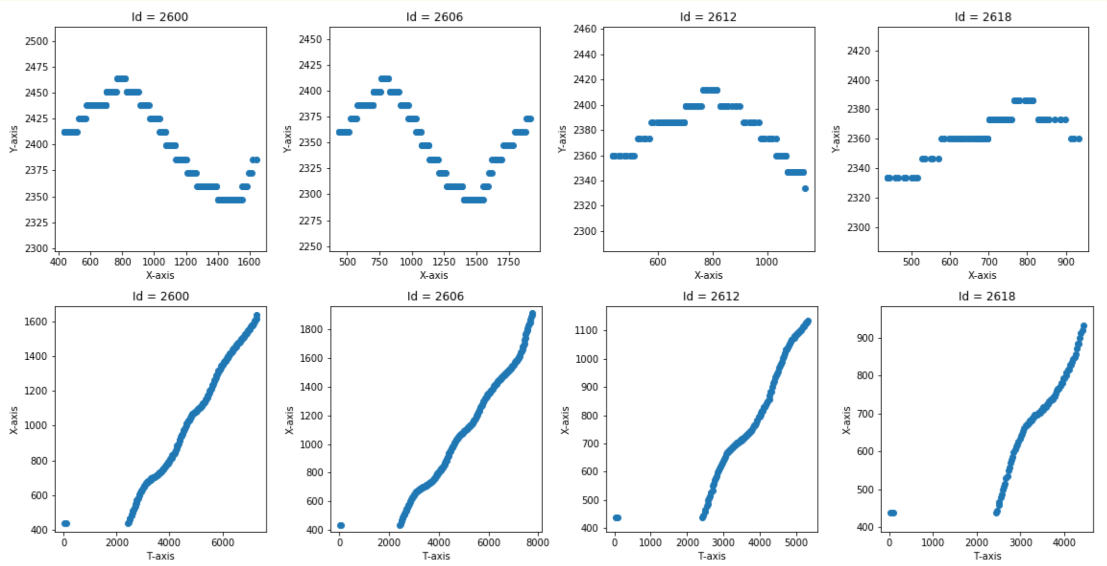


Machine behavior

The first attack mode

XY coordinate graph

TX coordinate graph



behavior ：

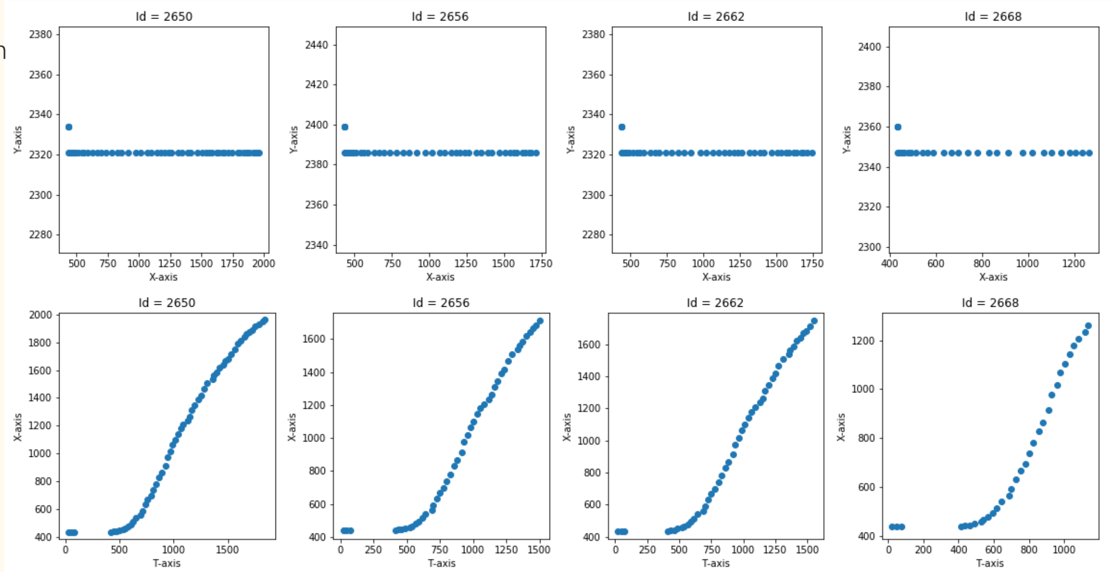
1. There is no significant acceleration or deceleration in robot program behavior
2. Dense sampling points
3. The mouse stays at the starting point for a long time

Machine behavior

The second attack mode

XY coordinate graph

TX coordinate graph



behavior ：

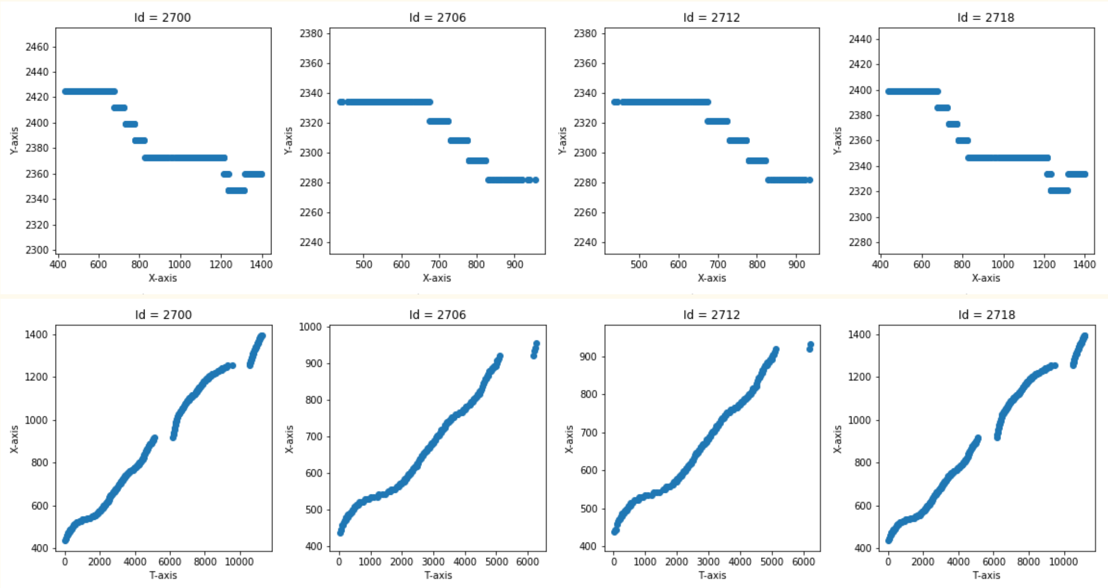
1. The robot program imitates the human acceleration and deceleration behavior to complete the sliding at an extremely fast speed
2. The acceleration time period is much greater than the deceleration time period
3. Mouse stays for a long time

Machine behavior

The third attack mode

XY coordinate graph

TX coordinate graph



behavior ：

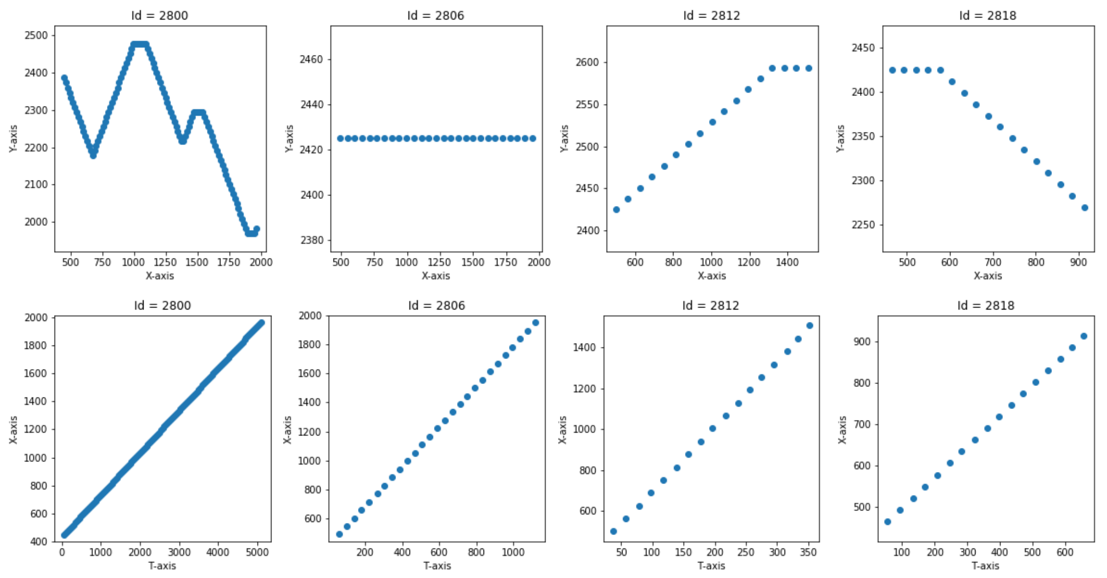
1. The entire sliding process moves at a nearly uniform speed
2. Short random jumping movements
3. There are a lot of similar trajectories

Machine behavior

The fourth attack mode

XY coordinate graph

TX coordinate graph



behavior ：

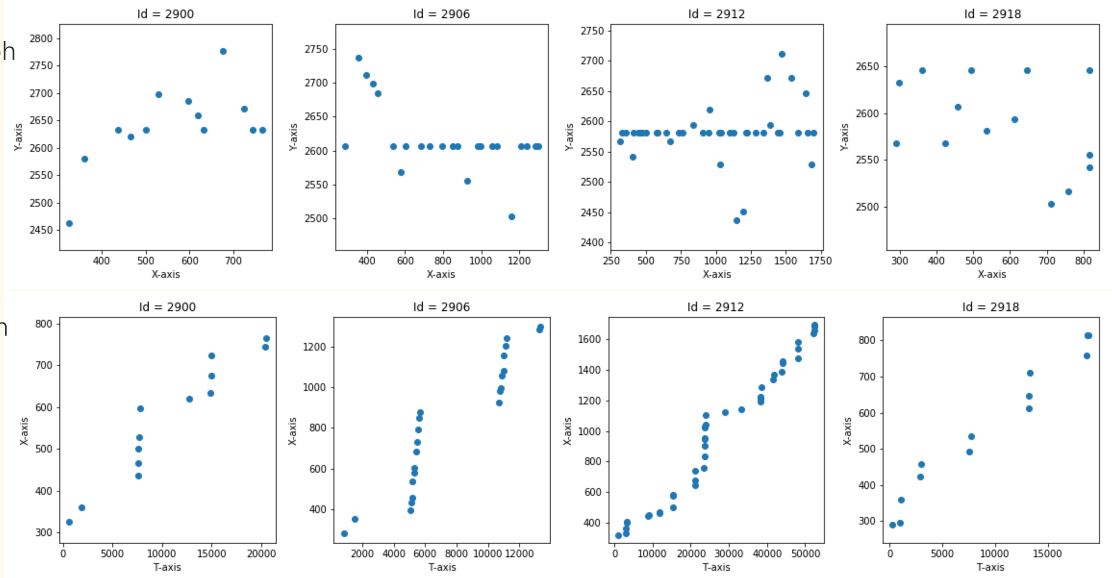
1. Approximately uniform motion with longitudinal deviation
2. Move to the next coordinate point at the same time interval, and the time interval is nearly equal
3. Simple machine behavior

Machine behavior

The fifth attack mode

XY coordinate graph

TX coordinate graph



behavior ：

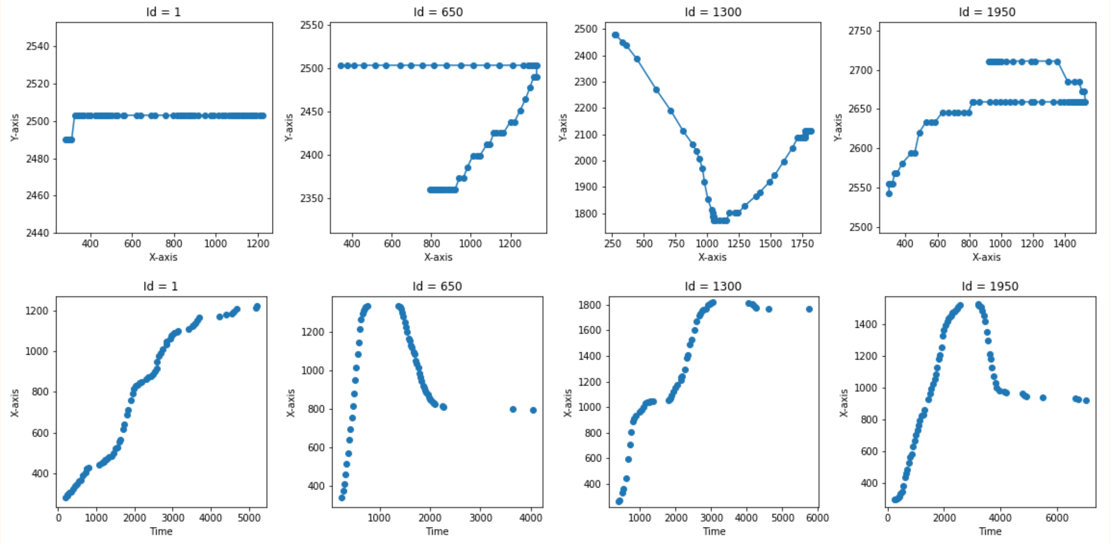
1. Longitudinal irregular random offset
2. Longitudinal irregular random offset
3. The time span between two adjacent acceleration segments is large

Human behavior

Behavior pattern

XY coordinate graph

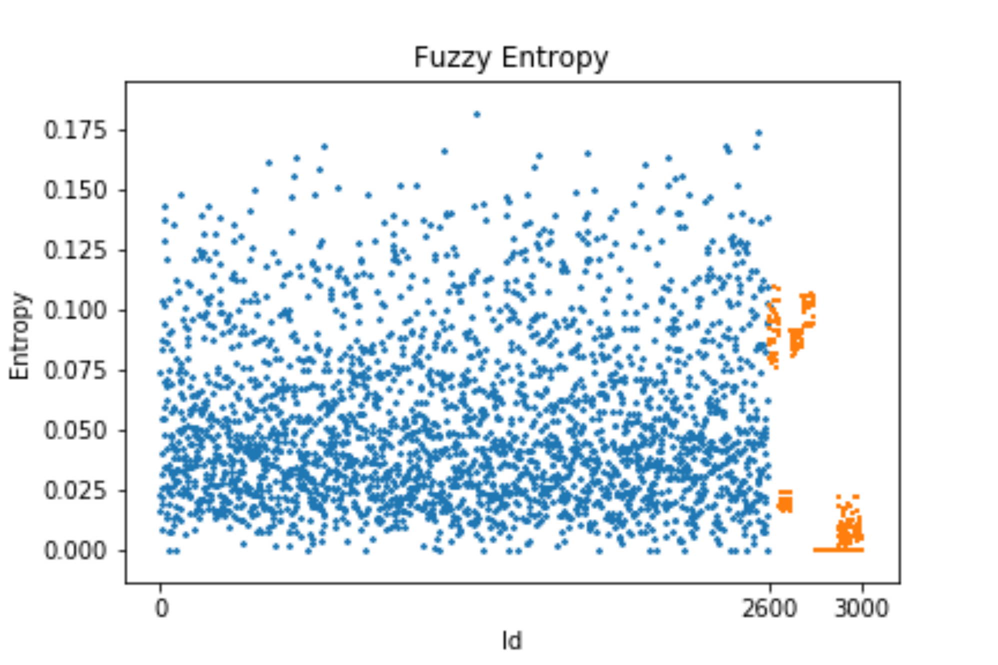
TX coordinate graph



behavior ：

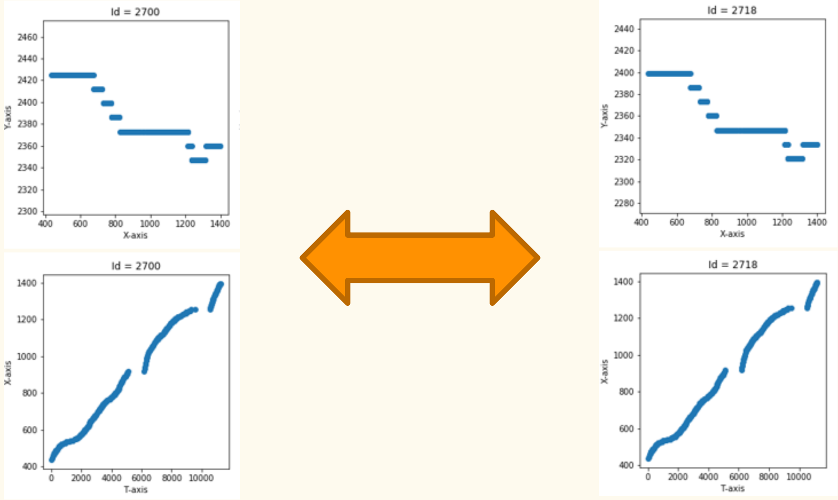
1. Significant acceleration and deceleration
2. Retracement
3. There is no big breakpoint in continuous time sampling

summary



- Difference in Entropy
- We use X-axis velocity as fuzzy entropy input
- Id 0-2599 is human and 2600-2999 is machine

Evasion Attack



behavior ：

1. Need to establish an attack database
2. Calculate the distance difference between the start point and the end point of the abscissa
3. Search for sliding samples that match the length in the attack library, and calculate new coordinate points and delay time.

Feature Engineering

Define the abscissa vector $\vec{X} = [x_1, x_2, x_3, \dots, x_n]$,
Ordinate vector $\vec{Y} = [y_1, y_2, y_3, \dots, y_n]$,
Time vector $\vec{T} = [t_1, t_2, t_3, \dots, t_n]$.
Calculate difference eigenvector $\overrightarrow{\Delta X} = [\Delta x_1, \Delta x_2, \Delta x_3, \dots, \Delta x_{n-1}]$,

$$\overrightarrow{\Delta Y} = [\Delta y_1, \Delta y_2, \Delta y_3, \dots, \Delta y_{n-1}],$$

$$\overrightarrow{\Delta T} = [\Delta t_1, \Delta t_2, \Delta t_3, \dots, \Delta t_{n-1}],$$

Velocity vector $\vec{V} = \frac{\overrightarrow{\Delta X}}{\overrightarrow{\Delta T}} = [v_1, v_2, v_3, \dots, v_{n-1}]$,
Acceleration vector $\vec{A} = \frac{\vec{V}}{\overrightarrow{\Delta T}} = [a_1, a_2, a_3, \dots, a_{n-1}]$.
Differential speed $\overrightarrow{\Delta V} = [\Delta v_1, \Delta v_2, \Delta v_3, \dots, \Delta v_{n-2}]$.
Differential acceleration $\overrightarrow{\Delta A} = [\Delta a_1, \Delta a_2, \Delta a_3, \dots, \Delta a_{n-2}]$.

Finally, definition $\vec{Q} = [q_1, q_2, q_3, \dots, q_t]$ Is the above physical feature vector ， q_i Is a vector value.

Feature Engineering

Max:

$$Arg_{max} = \max(\vec{Q})$$

Min:

$$Arg_{min} = \min(\vec{Q})$$

Average:

$$Arg_{mean} = \frac{q_1 + q_2 + q_3 + \cdots + q_t}{t}$$

StartingPoint:

$$Arg_{start} = q_1$$

MidPoint:

$$Arg_{mid} = q_{\frac{t}{2}}$$

EndPoint:

$$Arg_{end} = q_t$$

Standard deviation:

$$Arg_{\sigma} = \sqrt{\frac{1}{t}\sum_{i=1}^{t}(x_i - Arg_{mean})^2}$$

Feature derivation

1. XGBoost obtains the model score according to the feature gain of the construction tree, and uses all the features to fit the training set to obtain the high-scoring feature

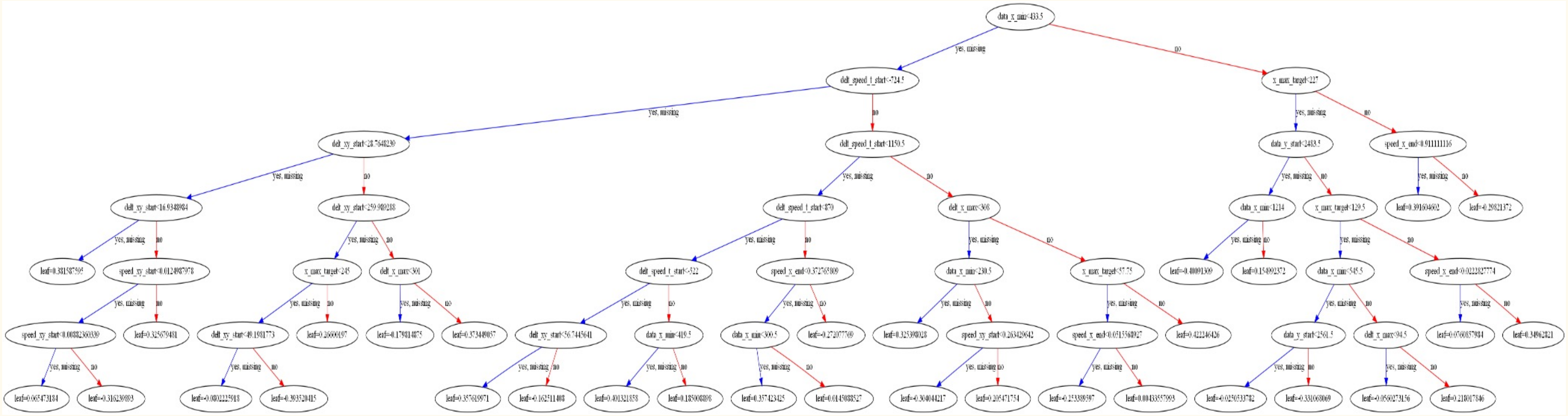
2. Use Pearson's correlation coefficient to remove collinearity features

$$r = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}}$$

3. Choose attack model classification features

feature	description
x_max_target	Maximum distance between abscissa and target point
data_x_min	Minimum horizontal sliding coordinate point
delt_x_max	Maximum abscissa difference
acc_speed_x_start	Initial acceleration of lateral sliding
speed_x_end	Lateral end speed
data_y_start	Longitudinal start value
speed_xy_start	Sliding start speed
delt_xy_start	Starting difference between two points
delt_speed_t_start	The difference in the degree of change of the start time

XGBoost model



Evasion Attack Detection

Evasion Detection using Manhattan Distance Algorithm

Algorithm 1: Evasion_Detect

Input: threshold k, Evasion_Feature feat, Reload_lib r, Suspect_sample s

Output: bool res

1. absValue = abs(r-s)
2. GreaterValues = where(absValue>k)
3. GValue = unique(GreaterValues)
4. AllLib = range(len(r))
5. SusIdList = difference(AllLib, GValue)
6. **if** SusIdList not empty **then**
7.   **for** id in SusIdList **do**
8.   **if** sum(GValue[i]) < k **then**
9.   res = 0
10.   **else**
11.   res = 1
12.else
13. res = 1
14.return res

Data preprocessing

The training data set and the test data set have the problem of unbalanced data proportions

Our Solution ：
Randomly select 70% of the coordinate points for each training sample as a new training sample and add it to the training data set. This process is repeated twice to obtain a total of 9000 training data sets, including 7800 human data and 1200 machine data.

Experiment

1. Test set experiment

Use Precision and Recall as the evaluation criteria of the experimental results, they are:

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

Classification Model	Precision	Recall	Average time
GBDT	98.31	99.84	1.32s
SVM	96.96	98.95	1.10s
Random Forest	98.69	99.88	5.40s
XGBoost	98.94	99.86	0.23s

Experiment

2. Validation set experiment

Use false rejection rate FRR as an evaluation indicator on the validation set

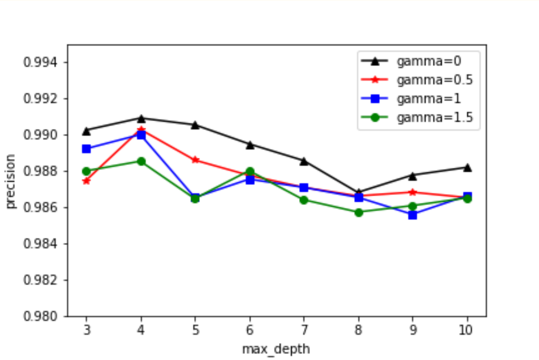
$$FRR = \frac{N_{FA}}{N_{GRA}}$$

Trajectory in 8 sliding directions and only extract the trajectory of the positive sliding of the X-Axis

Classification Model	FRR	FRR(X-Axis forward only)
GBDT	13.28	1.09
SVM	100	100
Random Forest	13.27	0.93
XGBoost	13.26	0.93

Experiment

3. Parameter optimization



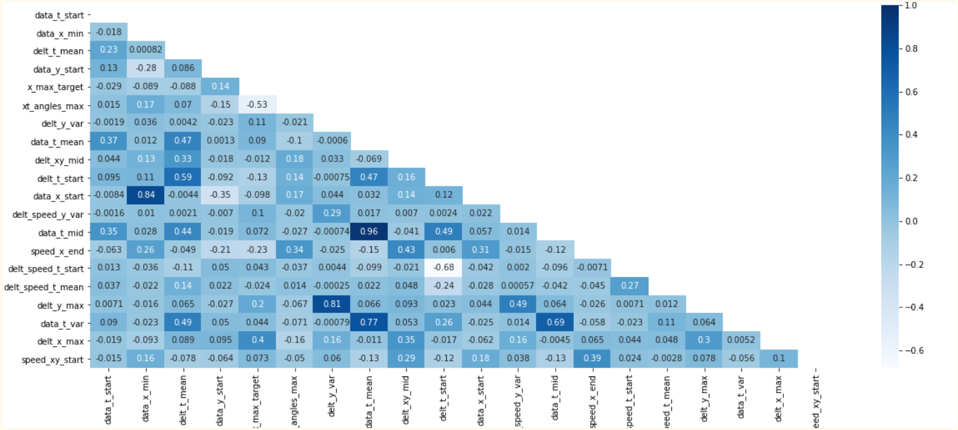
eta : 0.3
min_child_weight : 1.5

Method	Precision	Recall
Before adjusting parameters	98.94	99.86
After adjusting parameters	99.09	99.88

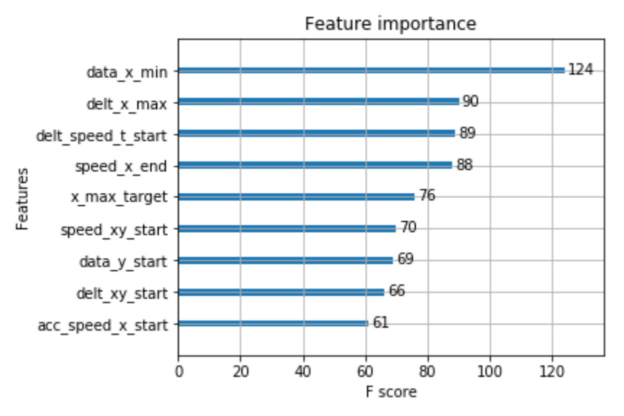
Experiment

4. Feature importance

Pearson correlation coefficient feature similarity



Feature importance score



Method	Precision	Recall
Before removing correlation	95.41	99.91
After removing correlation	98.94	99.86

Demo operation

Discussion

1. Feature learning for specific attack patterns may have limitations
2. It is necessary to learn and extract strong classification features for more attack patterns
3. Evasion attack detection may have misjudgment when the data set is large, and bring higher time consumption

THANK YOU FOR WATCHING