# Mô tả thực nghiệm Feature Selection

## I. Trong file total\_origin.csv

Khi sử dụng dataset đã trích xuất tỏng total.csv cần lưu ý

Trong 1. extract feature Reverse-APK-master.zip dùng để trích xuất file .APK sang dạng đặc trưng ở file .csv

Trong đó có file total.csv là file đã được trích xuất khi lấy dữ liệu từ drebin và AMD

Cấu trúc của file total.csv như sau:

1. cột đầu là tên file

2. tên họ mã độc

3. cột 3: đánh nhãn dạng số

4. 877 cột tiếp theo là permission (sắp xếp theo số lượng giảm dần các permission được sử dụng ở trong dataset)

5. 1001 cột tiếp theo là API ((sắp xếp theo số lượng giảm dần các API được sử dụng ở trong dataset)

Như vậy tổng số cột trong file là 1881

Số hàng 26715 hàng (trong đó hàng đầu là Header)

=> Nếu muốn nhanh thì chỉ chạy trên một bộ nhỏ (riêng AMD hoặc riêng Drebin hoặc có thể sử dụng dataset khác để tạo ra file total\_origin.csv này => đang xin của nhóm này mà nó ko thấy cho: https://zenodo.org/record/2593596#.Ymt4Ve3P1D8)

## II. Sử dụng TF\_IDF để tạo ra file total\_weight\_tf\_idf

các thư viện đều cho phép chạy TF\_IDF này đặc biệt là python. Ta có thể tham khảo:

- <https://viblo.asia/p/tf-idf-term-frequency-inverse-document-frequency-JQVkVZgKkyd> (tiếng việt, mô ta rõ cách hoạt động của tf-idf) => hiểu được cách hoạt động

- <https://towardsdatascience.com/tf-term-frequency-idf-inverse-document-frequency-from-scratch-in-python-6c2b61b78558> (tiếng anh, mô tả code sử dụng)

Nhìn chung, với mỗi feature sẽ có trọng số riêng cho từng file (cùng đặc trưng nhưng với mỗi file sẽ có giá trị khác nhau)

=> cách sử dụng khá dễ và dự kiến thời gian chạy sẽ nhanh.

=> Sau khi chạy xong code sẽ cho ra file: total\_feature\_weight\_id\_idf.csv

Sau tìm hiểu nếu cần có thể dùng phương pháp khác TF\_IDF để có thể so sánh.

Phương pháp đánh (tính toán) trọng số:

- IG – Information Gain

- PSO - particle swarm optimization (chưa đọc kỹ phương pháp). Tham khảo:

+ <https://csdlkhoahoc.hueuni.edu.vn/data/2021/6/NguyenHoangHa_2021.pdfs>

+https://www.researchgate.net/publication/317031499\_Computing\_Adaptive\_Feature\_Weights\_with\_PSO\_to\_Improve\_Android\_Malware\_Detection (hoặc: https://www.hindawi.com/journals/scn/2017/3284080/)

## III. Chạy code tạo ra feature selection

Chia thành 6 bộ

1. Bộ 1: lấy 30% feature (của riêng Permission và riêng API)

2. Bộ 2: Lấy 50% feature (của riêng Permission và riêng API)

3. Bộ 3: lấy 70% feature (của riêng Permission và riêng API)

4. bộ 4: Lấy 80% feature (của riêng Permission và riêng API)

5. bộ 5: lấy 90% feature (của riêng Permission và riêng API)

6. Bộ 6: Giữ nguyên 100% feature (của riêng Permission và riêng API)

Các đặc trưng trong file .csv đã được sắp xếp theo thứ tự giảm dần của tần suất xuất hiện của đặc trưng đó trong tất cả các file trong cả bộ dữ liệu.

- Chia thành các bộ để mô tả và đưa vào chương trình, còn thực ra vẫn có thể để ở 1 file .csv và khi chạy chương trình thì mới thao tác cắt % trong code.

## IV. Chạy chương trình

Chạy chương trình sẽ chạy trên một số mô hình học máy, học sâu như DT, RF, SVM, CNN, KNN

Chạy chương trình dự kiến sẽ mất nhiều thời gian

1. Thực nghiệm 1

Chạy 6 bộ trên nhưng chỉ của riêng Permission => cho vào các mô hình học máy học sâu

2. Thực nghiệm 2

Chạy 6 bộ trên nhưng chỉ của riêng API => cho vào các mô hình học máy học sâu

3. Thực nghiệm 3

Chạy 6 bộ trên và lấy cả Permission + API => cho vào các mô hình học máy học sâu

Mục đích việc chạy nhiều như vậy để đánh giá nhóm đặc trưng tốt, và các đặc trưng tốt trong mỗi nhóm.