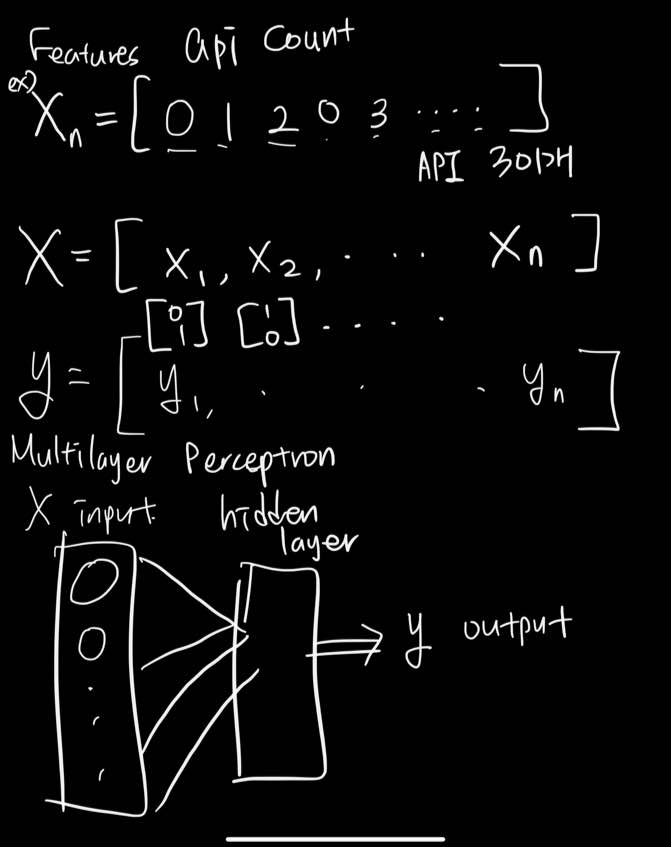
Assignment2 Malware Classification 보고서

2017029770 서윤희

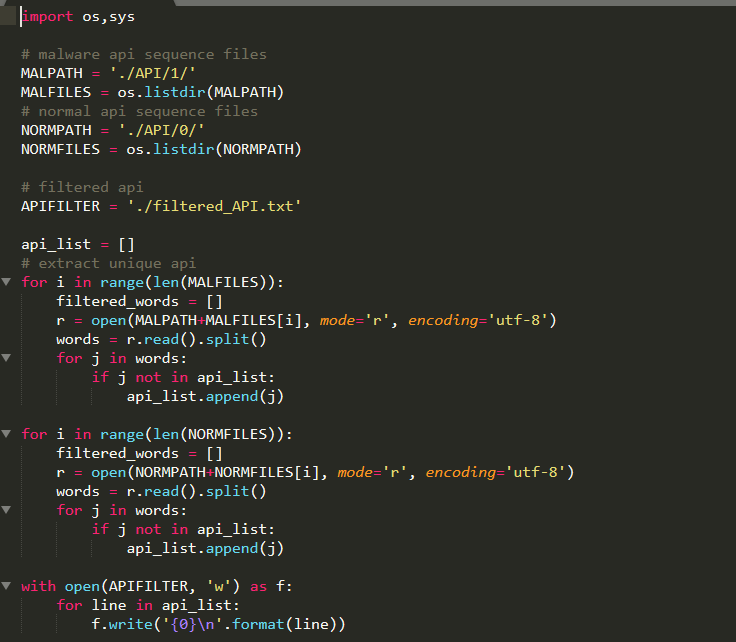
과제 목표, 구현 순서

* 이번 과제 - 동적 분석 결과로 얻은 API sequence 정보를 이용하여 악성 혹은 정상을 분류하는 것을 목표로 삼았습니다.
* Malware Classification을 위해 API sequence 분석을 위한 분류 모델을 설계하였습니다. 첨부로 주신 Native API 정보 분석 논문을 참고하였고, 논문을 읽으며 N-gram 분석과 TF-IDF를 적절히 혼합하여 사용한다면 높은 랜섬웨어 탐지율을 기대할 수 있음을 알게 되었습니다. 논문을 읽고 어떻게 학습을 해야할지 여러 머신러닝에 관련된 유명한 예제와 모델을 참고하여 아래와 같은 Naive한 분류 모델을 설계하였습니다.
* 현재 듣고 있는 인공지능 수업에서 배운 여러가지의 모델 중 뉴럴렛을 사용하여 multi-layer perceptron 알고리즘으로 0,1 이진분류를 위한 logistic regression sigmoid function을 사용하여 학습 후 분류를 생각하였습니다.
  1. 전체 sequence에서 input이 될 수 있는 API를 뽑아내기. X의 기준이 됨.
  2. 각 sequence 별로 API 빈도수를 카운트하여 악성,정상여부 라벨링하고, 학습 data 생성. 가중치는 따로 주지 않고 1로 count
  3. 사이킷런의 mlp(multilayer perceptron)를 활용하여 학습 후 정상, 악성 분류.
  4. 혹시나 학습이 잘 안된다면 2-gram으로 패턴을 추출해보거나 가중치 값을 다르게 주는 것 까지 고려해볼 것.



코드 구현 및 과정

1. 전체 데이터에서 input이 될 수 있는 API를 뽑아내기. 정상, 악성여부에 상관없이 API를 뽑아내어 학습할 input을 위한 text file을 따로 뽑아내었습니다. 겹치는 api가 없도록 중복을 제거하며 unique한 API들만 전체 파일들에서 추출. 전체파일에서 단일 api 만 추려내는데 20분 이상 소요되었던 것 같습니다.(깃에는 api 파일은 올리지 않았습니다.)

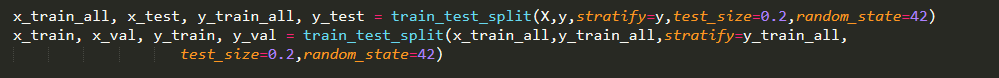


필터링한 API의 개수는 301개 였습니다.

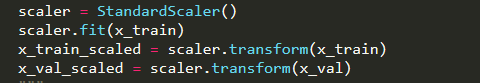
1. 그 후 악성과 정상파일을 각각 1000개씩 불러와 추출한 api list에 해당하는 api들을 읽으며 파일별로 count하여 만든 list를 y값(0 – 정상, 1 – 악성)과 라벨링하여 X, y data를 전처리 하였습니다.



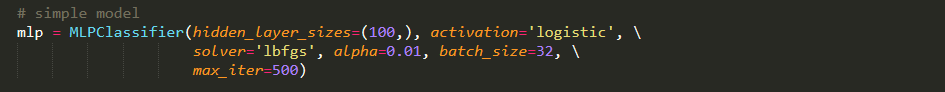
1. test를 위한 data set을 0.2, 즉 20프로 정도의 비율로 전체 데이터 셋에서 테스트 데이터를 분리합니다. X train, y train, x test, y test의 data set을 뽑아낼 수 있고, X,y 의 training data에서 validation을 위한 validation data를 20프로의 비율로 또 추려냅니다. Mlp를 다른 옵션을 주어 사용했을 때 성능평가를 하기 위함 입니다. 데이터셋의 셔플링과 분할을 위해 random\_state, stratify, test\_size를 파라미터를 지정해줍니다.



1. 분리해낸 data set에서 학습할 데이터를 표준화 처리합니다. 이진 분류를 하기 위해 linear regression, logistic regression의 가정을 하고 구현된 경우 표준화를 적용하는 것이 예측 성능 향상에 중요하기 때문입니다.



1. 사이킷 런에서 사용하는 MLPClassifier에서 default 값이 hidden layer의 사이즈를 100(default)로, hidden layer는 logistic sigmoid 함수를, solver에서 quasi-Newton 방법을 적용하여 학습합니다. 주로 사이킷 런 api에서 사용하는 default 예제와 방법을 참고하였습니다.



1. 테스트 셋에서 분리한 데이터를 학습 후, validation data set으로 성능을 평가합니다.





0.996875%의 정확도로 성능예측 결과가 나왔습니다.

구현 중 한계나 어려웠던 점

* 논문을 읽고 n-gram을 적용해서 해볼 수 있지 않을까 했었는데, 시작부터 2-gram으로 api 패턴을 추출해내려는데 시간이 오래 걸렸고, 학습을 하기에 자원과 시간의 한계를 많이 느꼈던 것 같습니다. 그 후 api 출현 빈도로 다시 data를 처리하여 학습할 계획을 세우게 되었습니다.