정보보호 R&D 데이터 챌린지 2018
AI 기반 악성코드 탐지

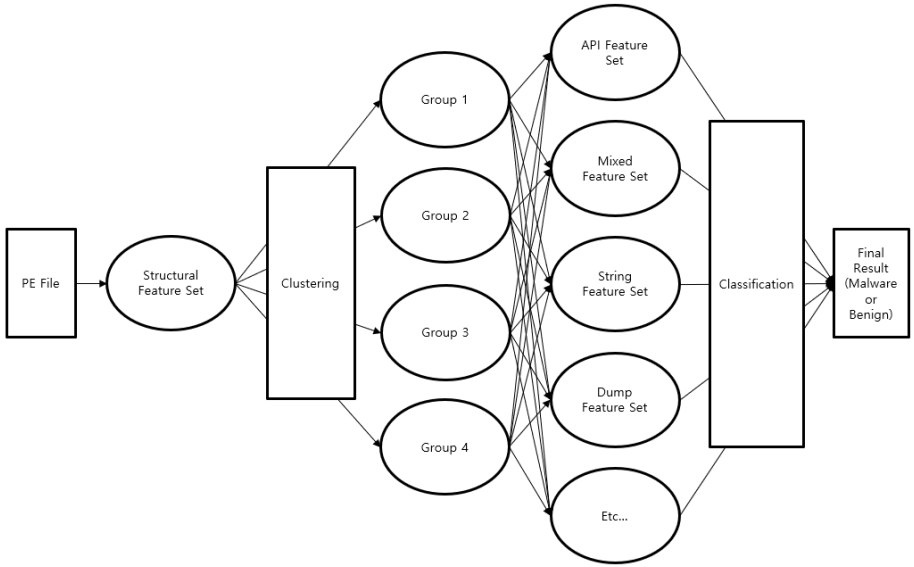
정성균 개인팀
Sungkyun Jung

목차

- **Background**
- **Modeling Process**
  - Data Preparation
  - Feature Engineering
  - Model Selection & Tuning with Cross-Validation
- **Conclusion**
  - Personal Opinion

Background

- **정보보호 R&D 데이터 챌린지 2017**
  - 참여 경험을 토대로 부족한 점 보완
    - Overfitting을 유도하는 과도한 Feature 개수 사용
    - 정제되지 않은 상태의 Feature 활용
    - **기본기에 충실**
      - **국영수 위주로 교과서를..**



Background

- **Based on Static Analysis**
  - Reference List

Name	Reference Link	Description
Manalyze	https://github.com/JusticeRage/Manalyze	A static analyzer for PE executables
Objdump	https://en.wikipedia.org/wiki/Objdump	it can be used as a disassembler to view an executable in assembly form
UPX	https://github.com/upx/upx	the Ultimate Packer for eXecutables
LightGBM	https://lightgbm.readthedocs.io/en/latest	LightGBM is a gradient boosting framework that uses tree based learning algorithms.
XGBoost	https://xgboost.readthedocs.io/en/latest	XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable
Scikit-learn	http://scikit-learn.org/stable/	Simple and efficient tools for data mining and data analysis

Modeling Process

- **Data Preparation**
  - Modeling Process 中 가장 중요한 단계
    - But, 대회 특성상 검증된 학습 데이터 제공
      - 근소한 차이로 순위 결정
  - UPX Unpack
    - 학습 데이터 10,000개 中 3,879(1,159 Benign / 2,720 Malware)개가 Packing 의심
    - 정적 Unpack이 가능한 UPX를 대상으로만 Unpack 시도
      - 632(118 Benign / 514 Mawlare)개 파일 Unpack 성공

Modeling Process

- **Feature Engineering**
  - 정적으로 분석 가능한 거의 모든 정보를 대상으로 Featurization 수행
    - PE Header
    - Section
    - Resource
    - Byte-1-Gram
    - Assembly Instruction
    - Import/Export Api
    - String
    - Rich Header
    - Version
    - TLS Callback

Modeling Process

- **Feature Engineering**
  - PE Header
    - Dos Header, File Header, Optional Header
    - Additional Information
      - Detected Languages Number
      - Has Debug Information

Modeling Process

- **Feature Engineering**
  - Section, Resource, Assembly Instruction
    - 출현 빈도가 높은 대상 선정 및 활용

Selected Section Name
'.rsrc', '.data' '.text', '.bss', '.crt', '.rdata', '.reloc', '.idata', 'data', '.edata', '.sdata', '.ndata', '.itext', '.tls', '.crt', 'bss', 'code', '.code'

Selected Resource Type
'RT_STRING', 'RT_DIALOG', 'RT_GROUP_ICON', 'RT_VERSION', 'RT_BITMAP', 'RT_RCDATA', 'RT_ICON', 'RT_GROUP_CURSOR'

Selected Opcode Name
'mov', 'lea', 'andl', 'je', 'jmp', 'add', 'sbb', 'sub', 'int3', 'shr', 'or', 'jb', 'dec', 'decl', 'incl', 'fxch', 'fsubr', 'jp', 'fstp', 'not', 'pushf', 'xchg', 'adc', 'in', 'clc', '(bad)', 'lcall', 'aaa', 'fiaddl', 'outsl', 'xlat', 'roll', 'les', 'outsb', 'aam', 'das', 'cld', 'notb', 'iret', 'fstps', 'ss', 'cmc', 'rorb', 'fnsave', 'flds', 'fiadd', 'jno', 'incb', 'cmpw', 'adcl', 'movswl', 'shrl', 'cpuid', 'fimul', 'rorl',

Modeling Process

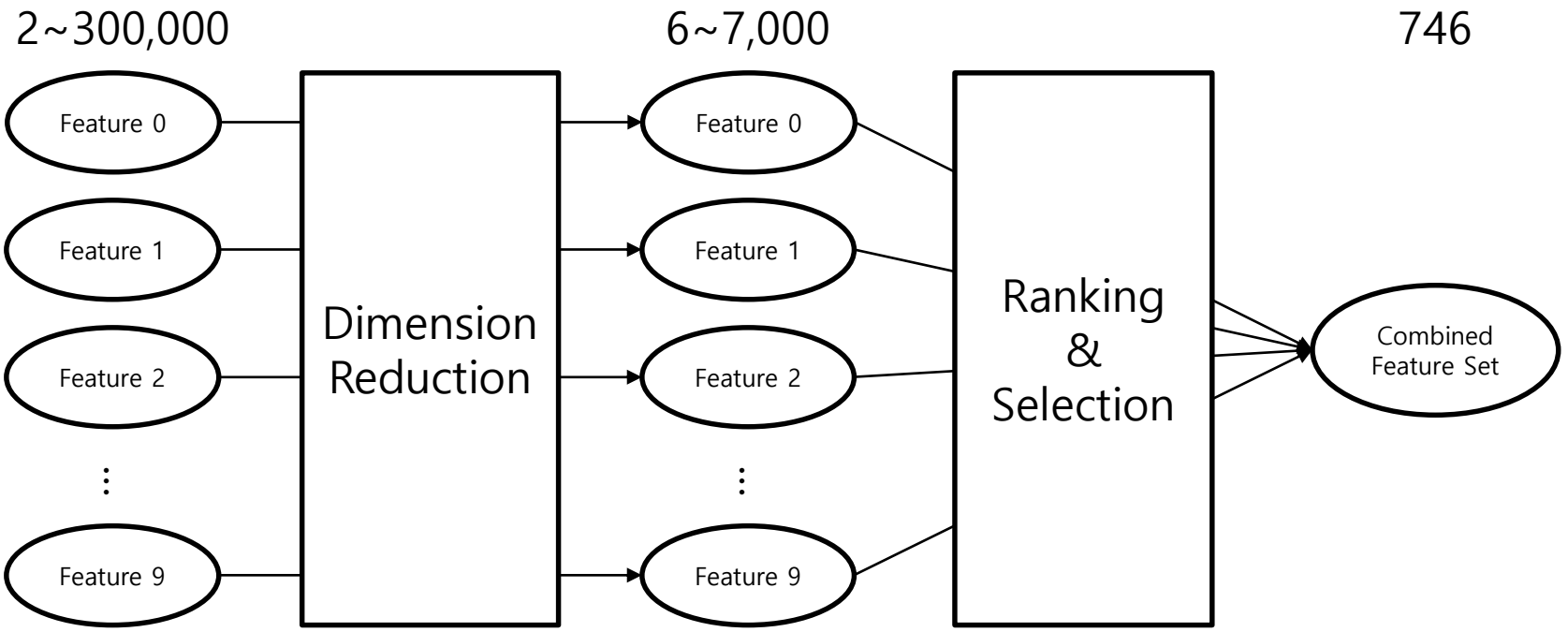
- **Feature Engineering**
  - Byte-1-Gram
    - PE 바이너리 내에서 출현하는 0~255 범위의 바이트 값의 빈도 수 및 비율 정보
    - Microsoft Malware Classification Challenge (BIG 2015)
      - Packing 여부와 관계 없이 악성코드 군('family') 분류에 효과적
      - Benign/Malware Classification에도 긍정적인 효과 기대 가능
  - Rich Header
    - Microsoft로부터 공식적으로 문서화되지 않은 구조 정보
    - PE 바이너리 빌드에 사용된 Compiler를 Detection 가능
    - Malware간의 유사성 분석에 활용한 연구 존재

Modeling Process

- **Feature Engineering**
  - Import/Export Api, String
    - 특정 Api 및 String의 출현 여부를 Feature로 활용
  - Version
    - Version Information Structure
      - LegalCopyright, FileType, FileVersion, FileSubtype, FileOs, FileDescription, CompanyName
  - TLS Callback
    - 프로그램의 실행 진입점(entry point) 이전에 실행되는 서브루틴
      - Malware가 디버거에 Load될 경우, 진입점에 도달하기 이전에 악성 행위 완료 가능

Modeling Process

- **Feature Engineering**
  - Feature Ranking & Selection
    - DataSet 기반 Feature Engineering의 자동화



Modeling Process

- **Model Selection & Tuning with Cross-Validation**
  - Gradient Boosting Algorithm
    - XGBoost & LightGBM
  - Model Tuning With scikit-learn's GridSearchCV
    - Stratified 3-fold-cross validation 수행 및 Best Model 선정
      - XGBoost : 0.9702970297029703
      - LightGBM : 0.9727972797279728
- **최종 Label 예측을 위한 계산 식**

$$YPred_{Classifier} = prediction\ value(\ 1 \text{ if } Malware\ else\ -1)\ of\ each\ classifier$$
$$YProb_{Classifier} = probability\ value(0.5\ \leq\ x\ \leq\ 1.0)\ of\ each\ classifier$$
$$YScore = YPred_{XGB} \times YProb_{XGB} + YPred_{LGBM} \times YProb_{LGBM}$$
$$Label = Malware \text{ if } YScore\ >\ 0\ else\ Benign$$

Personal Opinion

- **Why so Simple?**
  - Simple is the best

- **Modeling Process ≒ Black Box Problem**

  - 사람의 편견을 최소화
    - Byte-1-Gram
    - Noise-like String

Thank you for listening