사이버 공격 패턴 사전 감지 프로그램

1714490 김예진

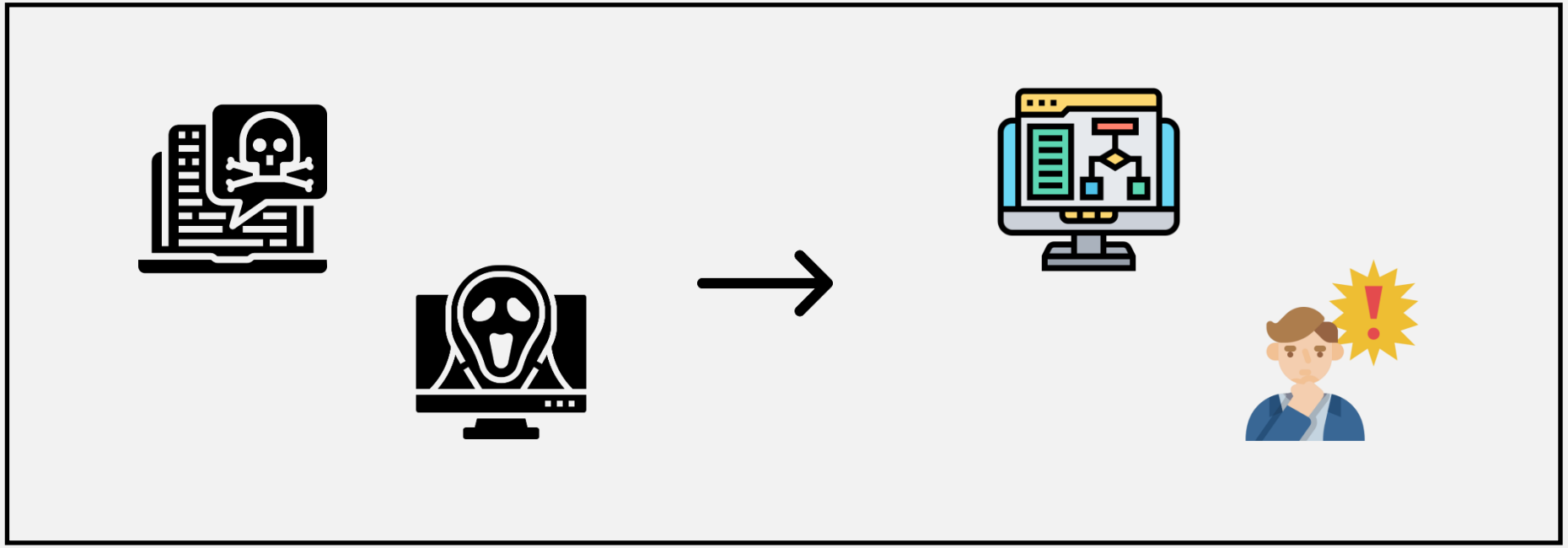
1715237 이혜승

1714573 조수정

CONTENTS

주제 선정 동기

1 주제 선정 동기



- 기술의 발전과 함께 지능화되는 사이버 공격에 대한 대응을 하기 위함
- 대규모 보안 투자가 어려운 1인 기업, 중소 기업을 대상

프로젝트 소개

2 프로젝트 소개

1. 개발 언어 & 환경

[Modeling]

개발 환경:

Windows 10. i5-7200U, RAM 8.0GB

개발 언어: Python 3.8.5

필요 라이브러리: conda 4.8.4, numpy 1.19.1, pandas 1.1.1, scikit-learn 0.23.2, scipy 1.5.2, sklearn 0.0

[Server]

개발 환경:

- Flask 1.1.2
- Werkzeug 0.15.4

개발 언어: Python 3.7.3

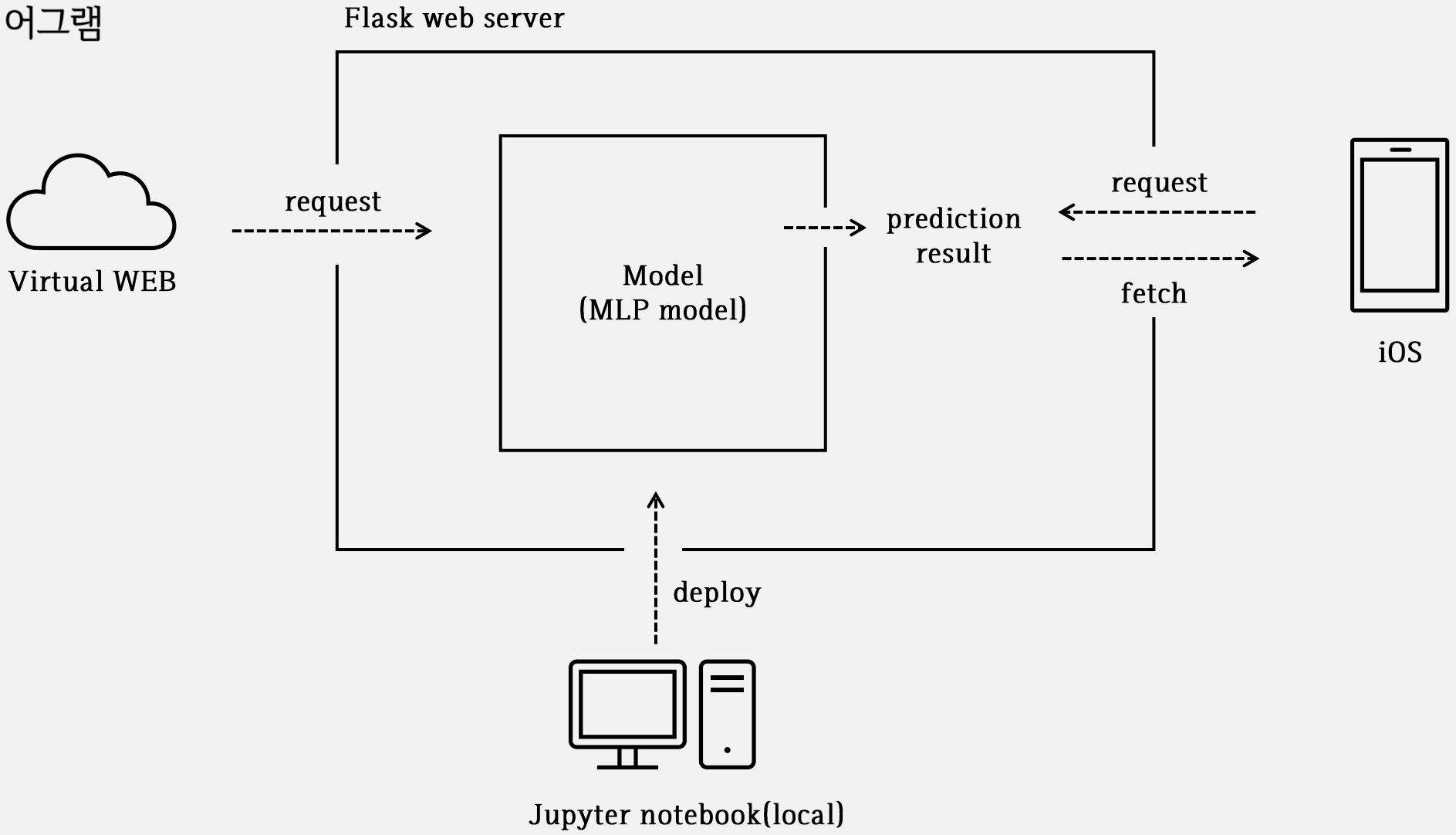
[iOS]

개발 환경:

- macOS Catalina 10.15.5
- Xcode ver. 11.4.1

개발 언어 : Swift 4

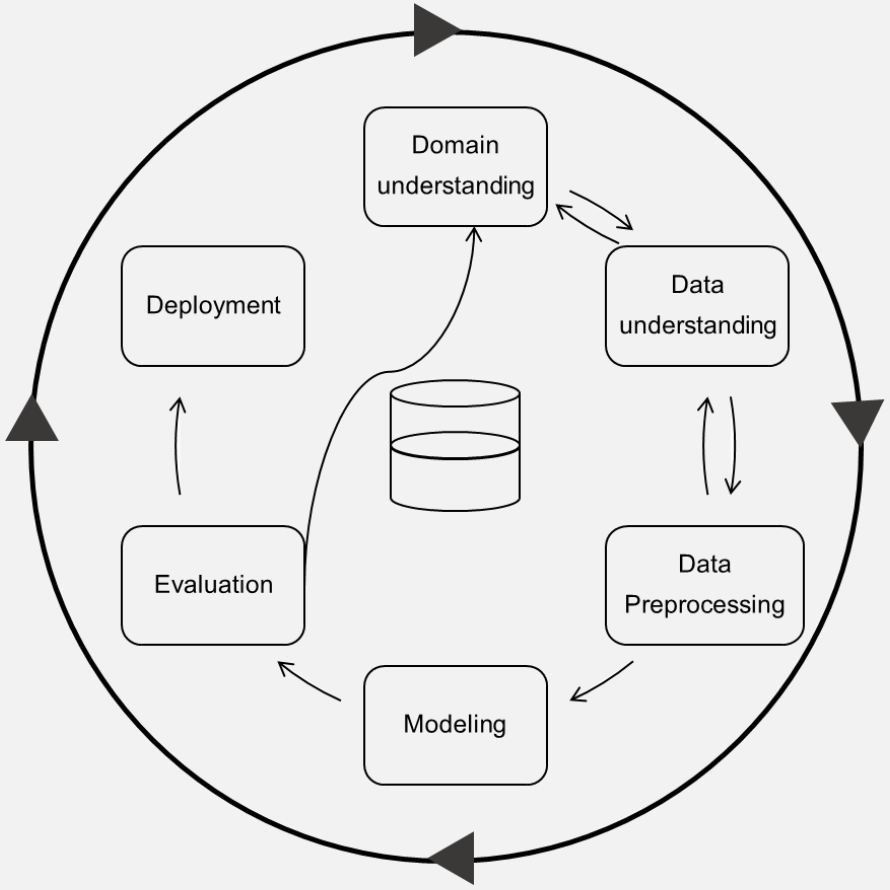
2. 시스템 다이어그램


Flask web server
Virtual WEB
request
Model
(MLP model)
prediction
result
request
fetch
iOS
deploy
Jupyter notebook(local)

프로젝트 실현 과정

모델링

3 프로젝트 실현 과정

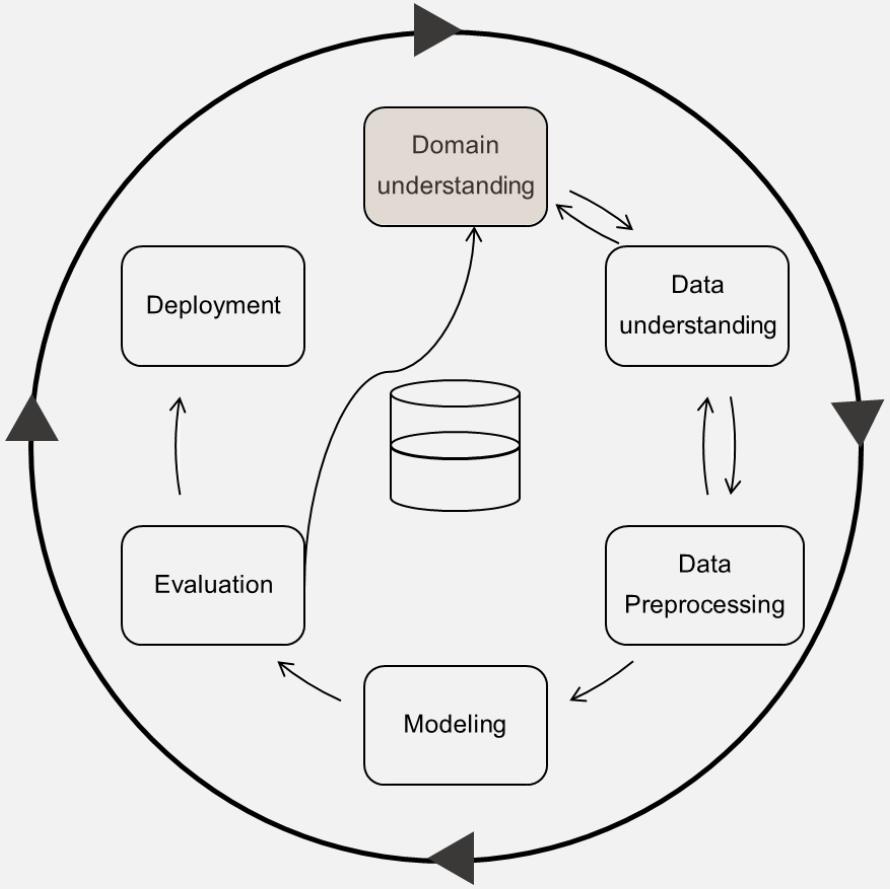


Cross-Industry Standard Process for Data Mining

[CRISP-DM] 분석 방법론

1. 비즈니스 이해(도메인 이해와 대응됨)
2. 데이터 이해
3. 데이터 준비
4. 모델링
5. 평가

3-1 프로젝트 실현 과정: 모델링

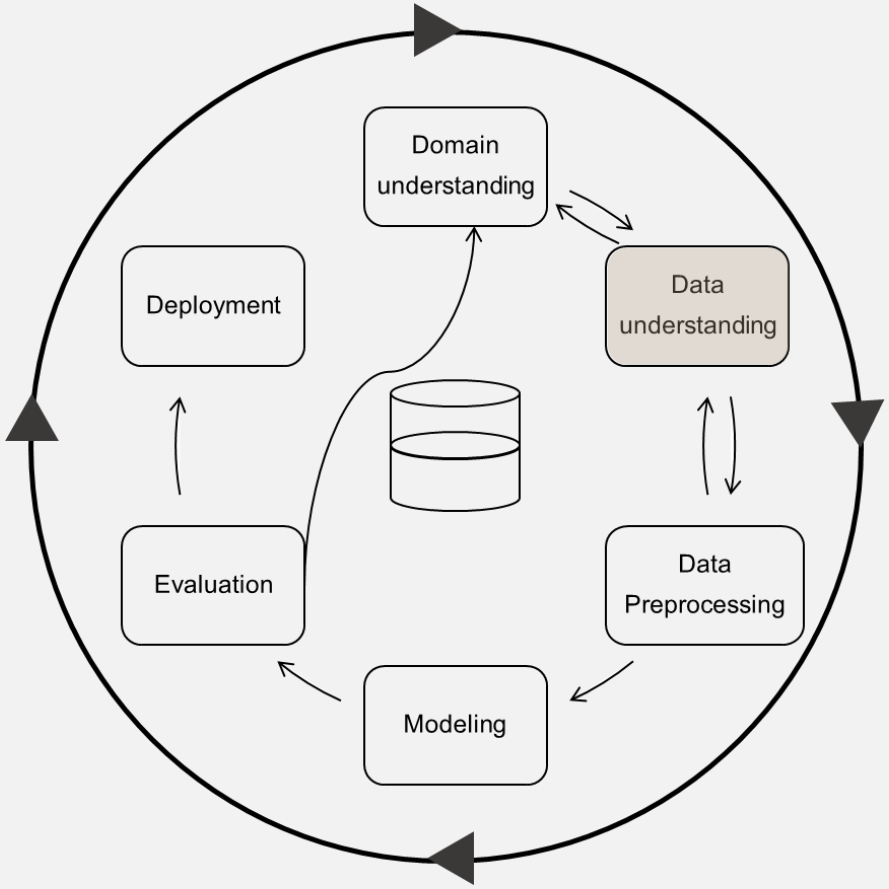


Cross-Industry Standard Process for Data Mining

[CRISP-DM] 분석 방법론

1. 비즈니스 이해(도메인 이해와 대응됨)
2. 데이터 이해
3. 데이터 준비
4. 모델링
5. 평가

3-2 프로젝트 실현 과정: 모델링



Cross-Industry Standard Process for Data Mining

[CRISP-DM] 분석 방법론

1. 비즈니스 이해(도메인 이해와 대응됨)
2. 데이터 이해
3. 데이터 준비
4. 모델링
5. 평가

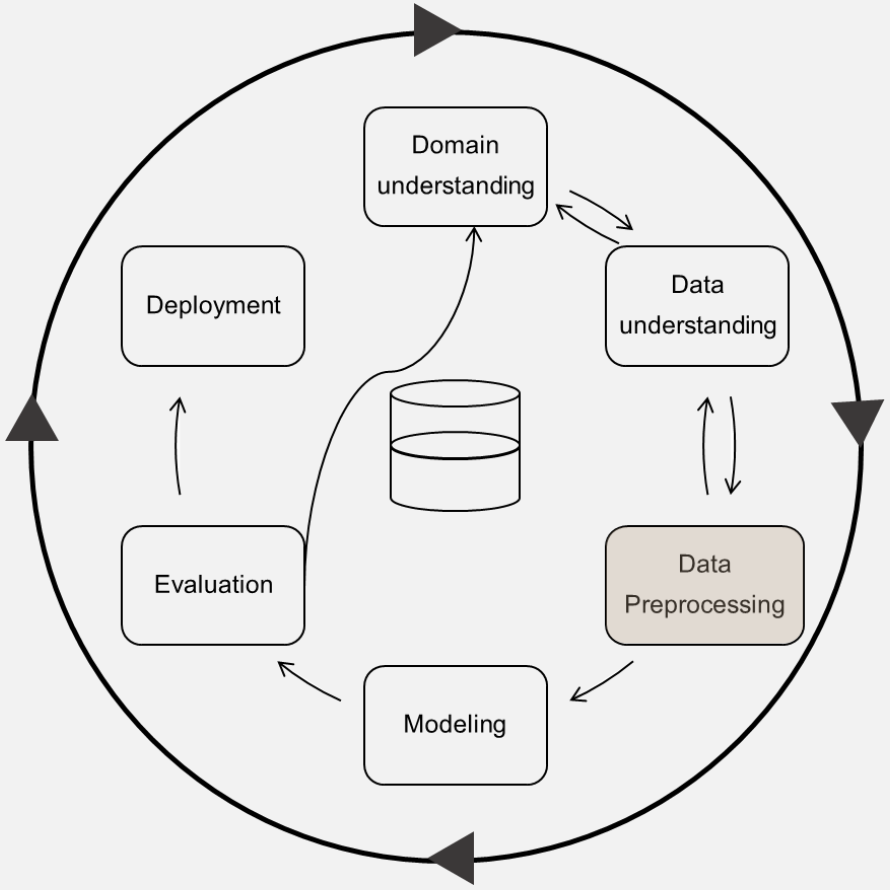
3-2 프로젝트 실현 과정: 모델링

2. 데이터 이해

- Network traffic으로 공격 상황을 제한함
- 가장 많은 선행연구가 진행되었던 KDD CUP'99, NSL-KDD, UNSW-NB15을 비교

	KDD CUP '99	NSL-KDD	UNSW-NB15
데이터셋 설명	군사 네트워크 환경, tcp dump 데이터 가공	KDD Cup을 정제	네트워크 트래픽의 현대적인 정상/공격을 추출
단점	DoS(smurtf, neptune)공격이 test data의 71%	비교적 옛날 데이터	모델링의 성능의 한계가 존재
선택 이유			현대적인 부분에 중점

3-3 프로젝트 실현 과정: 모델링



Cross-Industry Standard Process for Data Mining

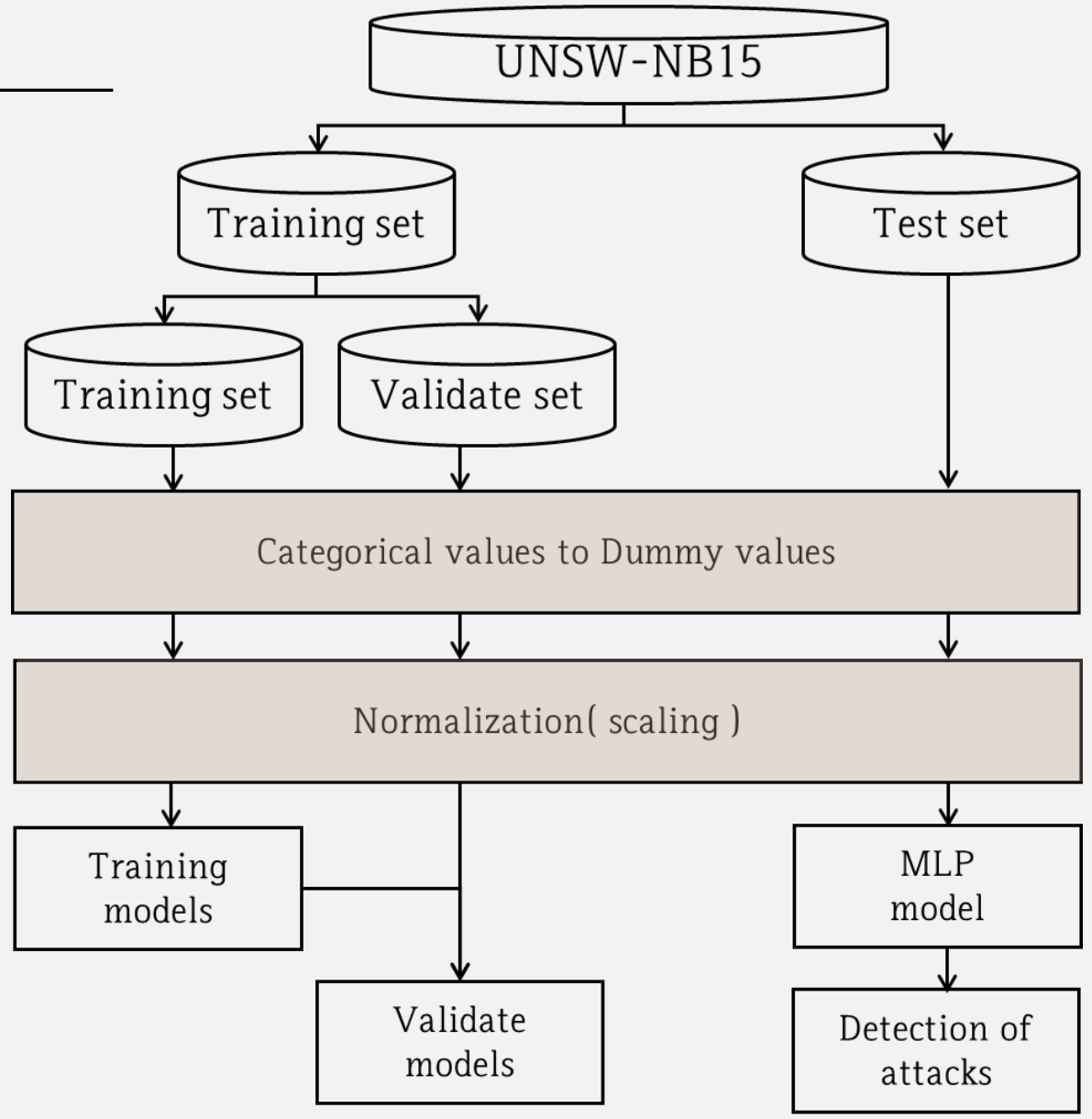
[CRISP-DM] 분석 방법론

1. 비즈니스 이해(도메인 이해와 대응됨)
2. 데이터 이해
3. 데이터 준비
4. 모델링
5. 평가

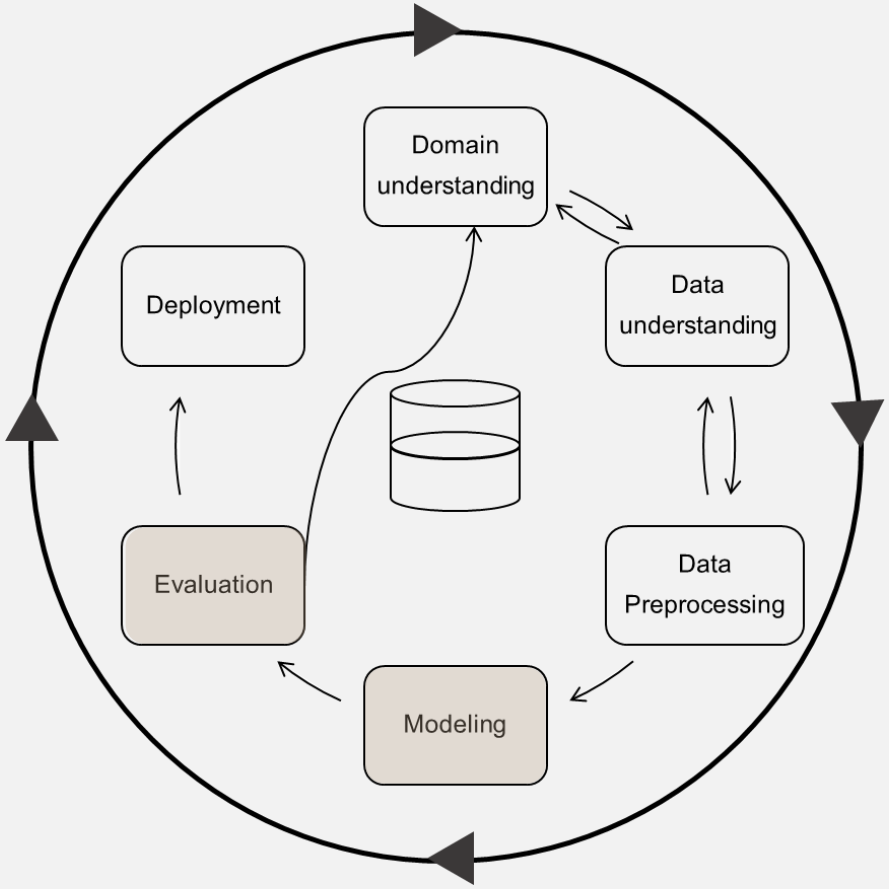
3-3 프로젝트 실현 과정: 모델링

3. 데이터 전처리

pandas NumPy SciPy



3-4 프로젝트 실현 과정: 모델링



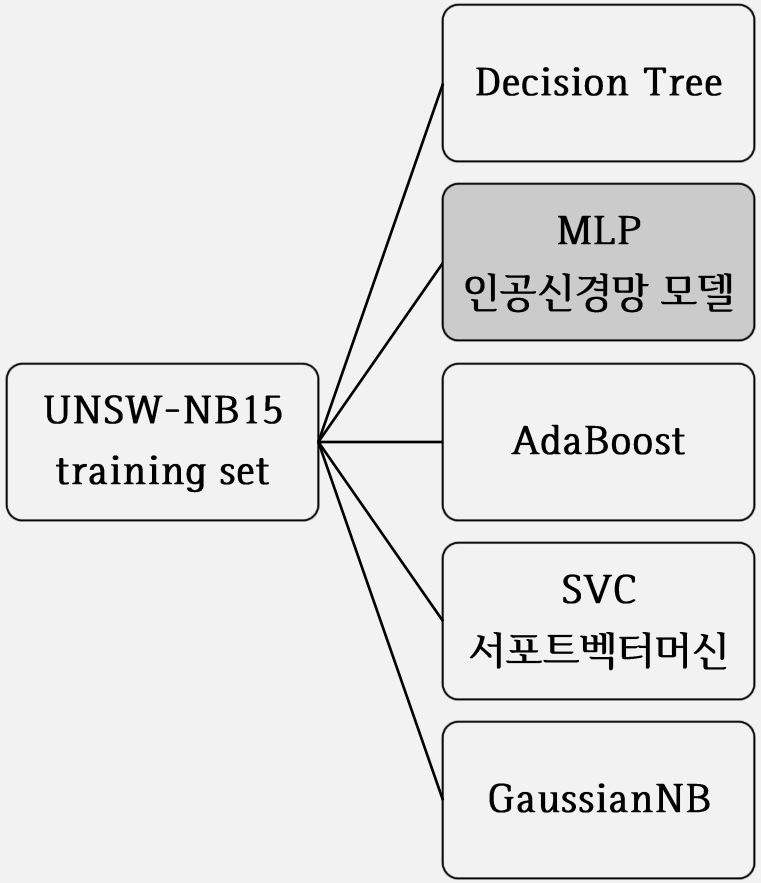
Cross-Industry Standard Process for Data Mining

[CRISP-DM] 분석 방법론

1. 비즈니스 이해(도메인 이해와 대응됨)
2. 데이터 이해
3. 데이터 준비
4. 모델링
5. 평가

3-4 프로젝트 실현 과정: 모델링

4. 모델링 및 평가



- Reference 참고 시, 해당 datasets에 대한 가장 높은 성능 89.134%
- 본 프로젝트에서는 약 82.3%의 성능.

	Decision tree	MLP	AdaBoost	SVC	Gaussian NB
Training accuracy	0.92	0.85	0.535	0.787	0.205
Validate accuracy	0.81	0.823	0.541	0.785	0.203
parameters		activation ='relu' solver = 'adam' alpha = 1e-4 H = (70, 58, 77, 95, 57) max_iter=10000		kernel = 'linear' C = 1	

3-5 프로젝트 실현 과정: 모델링

5. deploy

1 생성한 모델들의 최종 평가까지 마친 후, 결론적으로 사용할 예측 모델을 선택

2 선택된 예측 모델을 Flask server에 배포

3 웹서버에 '로그 정보를 알고 싶은 날'을 입력
→ 입력된 날의 로그 데이터를 전처리 → 예측 모델에 넣어서 결과를 반환

프로젝트 실현 과정

iOS

3-1 프로젝트 실현 과정 : 주요 기능

[iOS] 주요 기능

1 서버에게 로그 데이터 request & fetch

NetworkHelper 구현하여 Flask server와 통신

2 정상 데이터 및 비정상 데이터 개수를 카테고리 별로 표시

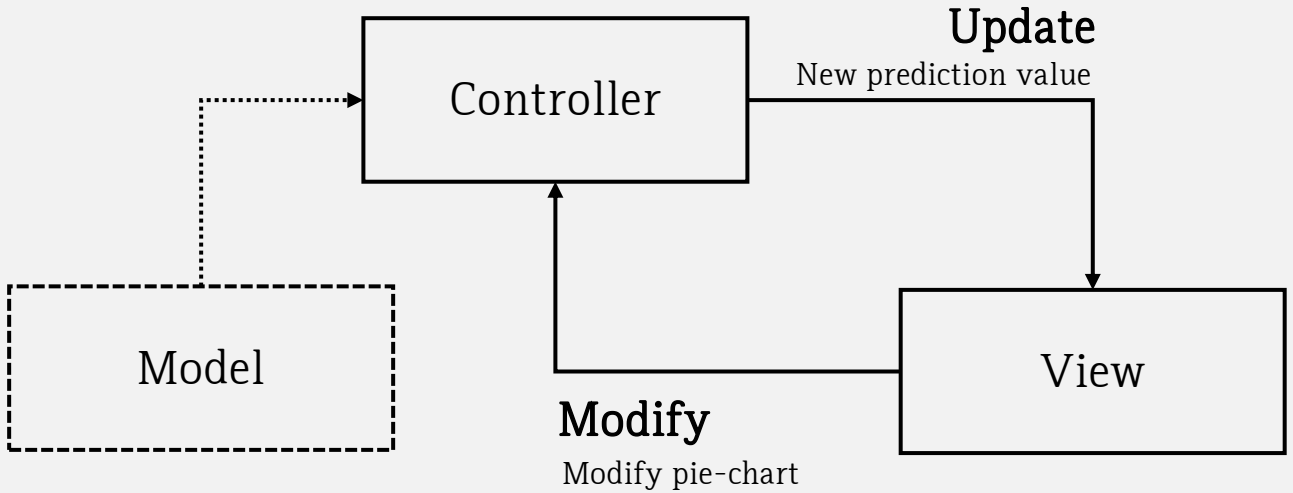
받아온 데이터들을 filter하여 비동기적으로 표현

3 Animated pie chart view 생성

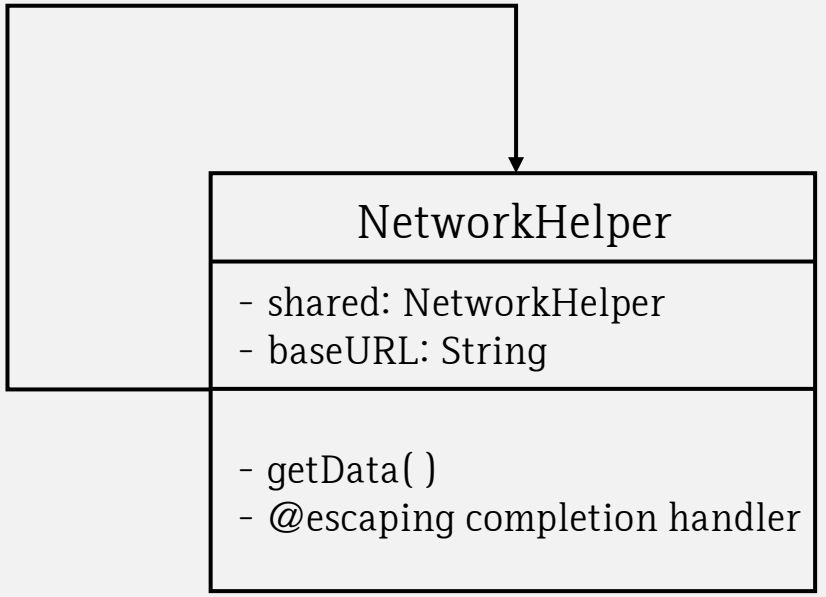
서버 통신 완료를 콜백 함수를 통해 completion 때 받은 데이터로 파이차트 생성

3-2 프로젝트 실현 과정 : Design Pattern

[iOS] 디자인 패턴 : MVC Pattern, Single-ton Pattern



- iOS의 자체 비즈니스 로직이 없으므로 Model 의 역할 제외
- Controller : 서버로 부터 받아온 데이터 정보를 view에게 전달
- View : 새로운 정보를 통해 pie chart를 재표현(re-draw)



- 네트워크 세션관리는 singleton pattern으로 구현
- task를 resume하여 재사용

3-3 프로젝트 실현 과정

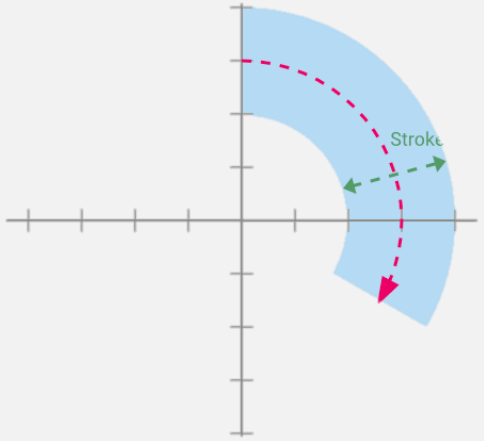
[iOS] 구현

Network Helper

- singleton 객체를 사용하여 네트워크 관련 task를 진행하도록 구현
- request를 서버에게 보내고 정보를 받아오는 작업은 비동기적으로 실행
- fetch해오는 데이터는 json 형태이며, NetworkHelper class에서 decoding 진행

3-3 프로젝트 실현 과정

[iOS] 구현



Pie Chart

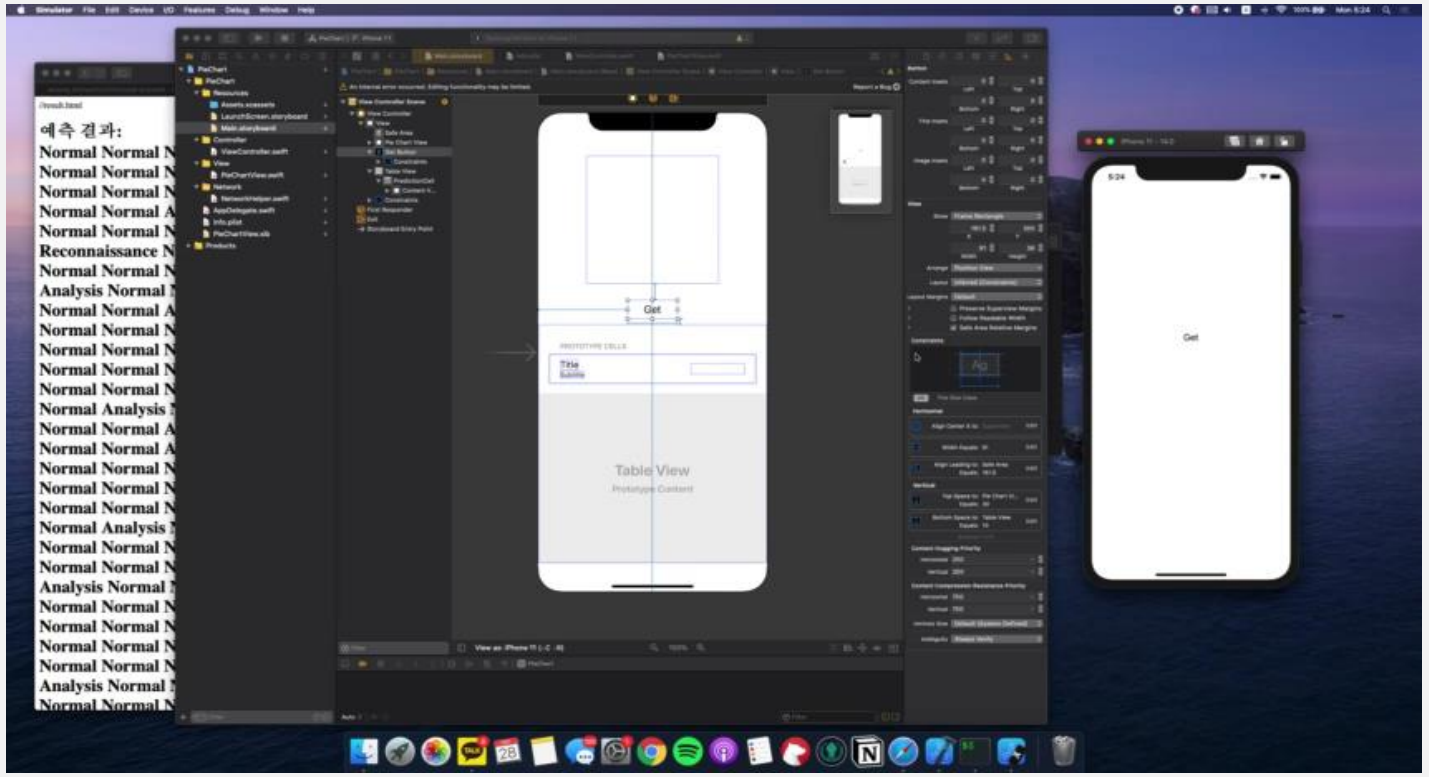
- Bezier curve의 원리를 통해 파이 중심 계산 후
  차트 레이블 생성

  - UIBezierPath와 CAShapeLayer를 통해 구현
  - UIBezier Path : 파이 차트의 레이블을 슬라이스에 중심에 두기 위해 사용
  - CAShapeLayer : Stroke path와 stroke width 속성을 사용해 곡선 구현

- Custom view로 직접 생성

시연 영상

4 시연 영상


예측 결과:
Normal Normal N
Normal Normal N
Normal Normal N
Normal Normal A
Normal Normal N
Reconnaissance N
Normal Normal N
Analysis Normal
Normal Normal A
Normal Normal N
Normal Normal N
Normal Normal N
Normal Normal N
Normal Analysis
Normal Normal A
Normal Normal A
Normal Normal N
Normal Normal N
Normal Normal N
Normal Analysis
Normal Normal N
Normal Normal N
Analysis Normal
Normal Normal N
Normal Normal N
Normal Normal N
Normal Normal N
Analysis Normal
Normal Normal N
Get
Title
Table View
Prototype Content
Get

Challenging

5 CHALLENGING

- 실제 로그 데이터 수집이 어려움
- 테스트시, 레코드 하나씩 입력 받아 트레이닝셋과 같은 형태로 전처리해주는 것이 어려움
- 서버에서 데이터를 받아오는 과정이 비동기라 완료 시점에 뷰를 업데이트 하는 것이 어려움
- GPU가 없는 로컬에서 모델링을 진행함

Efforts to overcome Challenges

- 로그에 관련한 open data(UNSW-NB15 dataset)를 사용
- 테스트셋을 5천 개씩 쪼개서 일일 로그로 가정
- Controller에 데이터를 받아오는 함수에 escaping closure를 만들어 놓고 NetworkHelper의 1개 task가 끝나면 view를 업데이트 하게 함

Future Work

6 Future Work

- 실제 로그데이터를 수집하여, 실시간 예측을 하고자 함
- 실시간 예측 시, iOS notification을 구현하고자 함.
- 비지도학습으로 모델을 통해 모르는 사이버 공격의 상황을 탐지하고자 함
- 성능을 더 높이는 시도를 위해, GPU를 활용한 모델링 진행하고자 함.

7 Reference

[1]M. Tavallaee, E. Bagheri, W. Lu, A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set", IEEE Symposium on Computational Intelligence in Security and Defense Application(CISDA 2009)

[2] 조정래, 성행남, 안병혁 "의사 결정트리와 인공 신경망 기법을 이용한 침입탐지 효율성 비교 연구", 디지털산업정보학회 논문지 제11권 제4호 -2015

[3] Mohamed Amine Ferrag, Leandros Maglaras, Sotiris Moschoyiannis, Helge Janicke, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study”, Journal of Information Security and Applications(2020)

Thank you