CPS 7조

**스마트 홈 사용자 편의 증진을 위한 기기별 데이터 수집 모델**

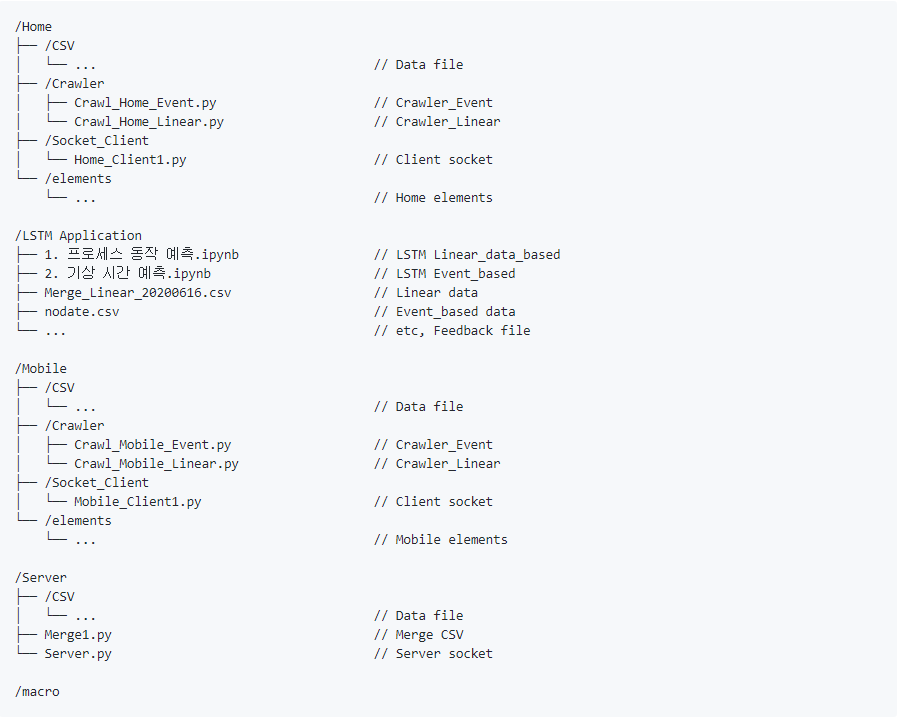
7조 팀원

김진욱 양인선 정영주

# 프로젝트 목적

* 스마트 홈에 있는 기기별 프로세스 데이터를 수집하는 모델 구축
* 수집된 데이터를 학습 데이터로 활용하여 일정 루틴 생성

# 구조



# 프로젝트 결과

## 환경 구성

### 구조도

스크린샷이(가) 표시된 사진

자동 생성된 설명

### 실제 구현 디자인

스크린샷이(가) 표시된 사진

자동 생성된 설명

## 크롤러 개발

### 구조도

스크린샷이(가) 표시된 사진

자동 생성된 설명

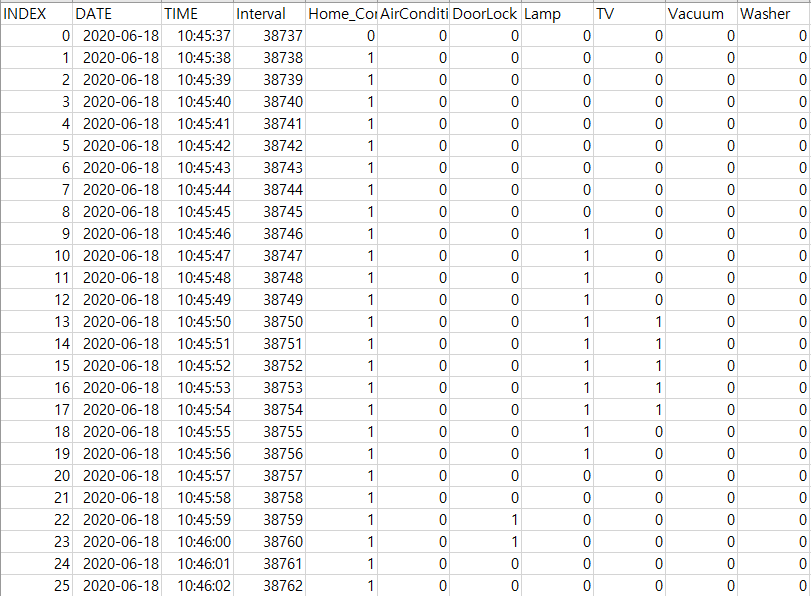
스크린샷, 표지판이(가) 표시된 사진

자동 생성된 설명

### 실제 출력 CSV

1. Linear

Home\_data\_Linear\_2020xxxx.csv



1. Event

Home\_data\_event\_2020xxxx.csv

검은색, 목재의, 캐비닛, 하얀색이(가) 표시된 사진

자동 생성된 설명

## 클라이언트 소켓 개발, 서버 프로그램 개발

### 구조도

스크린샷이(가) 표시된 사진

자동 생성된 설명

### 실제 출력 CSV

Merge\_Linear\_2020xxxx.csv

비이(가) 표시된 사진

자동 생성된 설명

## 매크로 개발

* 데이터 수집의 용이성을 위해 매크로를 제작
* 이후 매크로를 활용하여 연속적인 데이터 수집 완료

## 프로세스 동작 예측

### 개요

1초 단위의 프로세스 별 플래그 데이터를 이용하여 LSTM 알고리즘으로 향후 프로세스 동작을 예측한다. 이때 각 프로세스 별 플래그를 문자열(이진수 형태)로 결합한 데이터를 사용하여 학습한다.

### 데이터 전처리

‘Alarm’, ‘Lamp’, ’TV’ 칼럼만 추출한다.

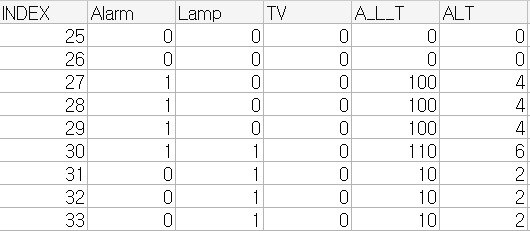
결측 값 처리: 1초전 플래그와 1초후 플래그가 0인 경우에만 결측 값을 0으로 대체하고, 그 외의 경우에는 1로 대체한다.

동일 시간대의 Alarm 플래그, Lamp 플래그, TV 플래그 값을 문자열 형태로 결합하여 ‘A\_L\_T’ 칼럼을 생성한다.

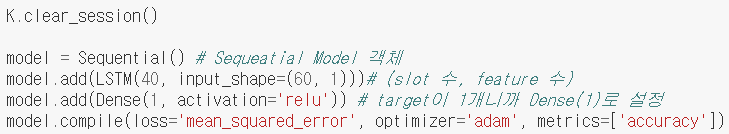
‘A\_L\_T’ 칼럼의 값(이진수로 취급)을 십진수로 변환하여 ‘ALT’ 칼럼을 생성한다.

인덱스를 기준으로 하여 ~3000: 학습 데이터, 3001~4500: 검증 데이터, 4500~5800: 테스트 데이터로 분리한다.

window 크기를 60으로 설정하여 1분단위로 학습할 수 있게 한다.



### LSTM 모델 소개



Keras를 사용하여 Sequential model 객체를 만든 후 LSTM 레이어와 Dense 레이어를 추가한다.

input shape는 (slot수, feature수) 즉, (60,1)로 설정하고 activation 함수는 ‘relu’를 사용한다.

컴파일 과정에서 손실함수는 mse로, optimizer는 ‘adam’으로 설정해주었다.

metrics=’accuracy’로 설정하여 정확도를 측정하였다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

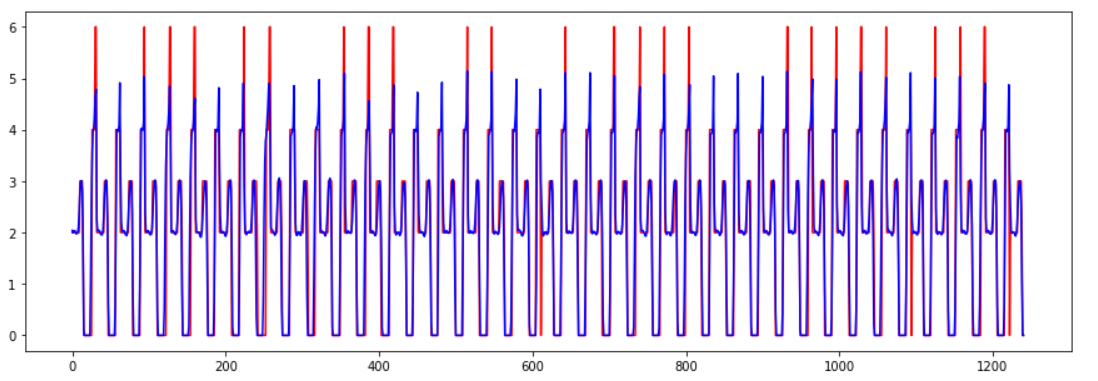
fit 과정에서 epochs=300, batch\_size=120으로 설정하였고, 검증데이터를 입력 값으로 함께 넣어주었다.

오렌지, 쥐고있는, 빨간색이(가) 표시된 사진

자동 생성된 설명

### 결과 분석 및 시행착오

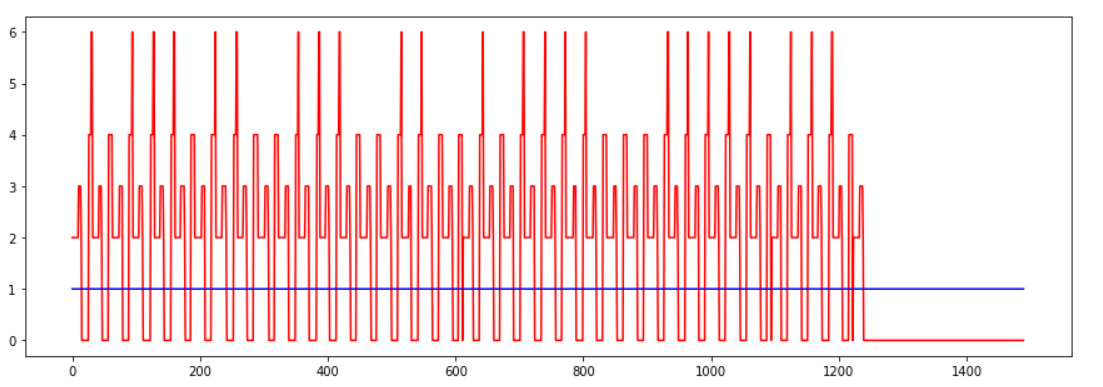
#### 성공 결과 분석

스크린샷이(가) 표시된 사진

자동 생성된 설명

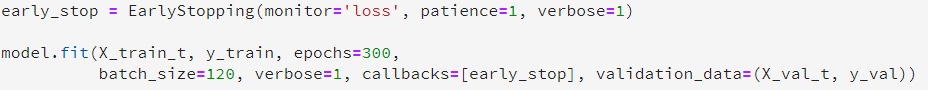
정확도 0.8267로, 잘 예측되었다.

#### 시행착오 사례1 - 활성 함수를 sigmoid로 설정한 경우



예측 값이 1로만 나오는 것을 볼 수 있다.

#### 시행착오 사례2 - early-stopping 지정해준 경우

연필이(가) 표시된 사진

자동 생성된 설명

Epoch 9에서 학습이 중단되었고, 활성 함수로 ‘sigmoid’를 선택했을 때보다 훨씬 잘 예측하는 것을 볼 수 있었다. 하지만 정확도가 0.49 정도로, 높지 않았다

### 활용 방안

각 프로세스의 다음 플래그를 예측함으로써 프로세스 간의 관계를 파악하고, 특정 상황에서 프로세스를 자동으로 동작 시켜주는 자동화 기능을 제공하는 등의 응용 방안을 생각해볼 수 있다.

### 피드백 반영

스크린샷이(가) 표시된 사진

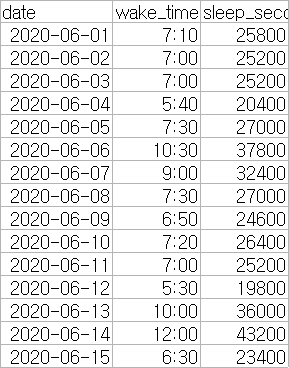
자동 생성된 설명

활성 함수로 sigmoid를 설정했을 때 정확도가 낮은 것을 보완하기 위해 학습 데이터 값을 0~1사이의 값으로 스케일링 해보았다. 그 결과, 정확도가 비교적 상승한 결과를 볼 수 있었다.

## 사용자 기상 시간 예측

### 개요

일 단위의 사용자 기상 시간 데이터를 이용하여 다음 기상 시간을 예측한다. 이때 자정(0시)을 기준으로, 시간 데이터를 ‘초’단위로 변환하여 학습시킨다. 데이터는 수집할 수 없다고 판단하여 임의로 생성하였다.



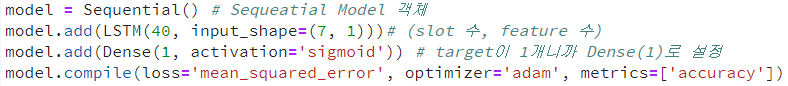
### 데이터 전처리

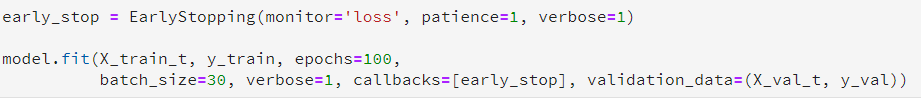
‘MinMaxScaler’를 사용하여 데이터를 0~1사이의 값으로 스케일링한다.

인덱스를 기준으로 하여 ~55: 학습 데이터, 56~69: 검증 데이터, 70~83: 테스트 데이터로 분리한다.

window의 크기를 7로 설정하여 일주일단위로 학습할 수 있게 한다.

### LSTM 모델 소개

스크린샷이(가) 표시된 사진

자동 생성된 설명

Keras를 사용하여 Sequential model 객체를 만든 후 LSTM 레이어와 Dense 레이어를 추가한다.

input shape는 (slot수, feature수) 즉, (7,1)로 설정하고 activation 함수는 ‘sigmoid’를 사용한다.

컴파일 과정에서 손실함수는 mse로, optimizer는 ‘adam’으로 설정해주었다.

metrics=’accuracy’로 설정하여 정확도를 측정하였다.

fit 과정에서 epochs=100, batch\_size=30으로 설정하였고, 검증데이터를 입력 값으로 함께 넣어주었다.

### 결과 분석 및 시행착오

#### 활성 함수를 relu 혹은 tanh로 설정한 경우

라켓볼, 게임이(가) 표시된 사진

자동 생성된 설명

예측에 실패했다.

#### 활성 함수를 sigmoid로 설정한 경우

지도이(가) 표시된 사진

자동 생성된 설명

활성 함수를 sigmoid로 설정한 경우, early\_stopping과는 상관 없이 같은 결과를 보였다. 비교적 나은 그래프 결과가 나왔지만, 0~1의 범위로 스케일링 한 것을 생각해 본다면 0.2의 오차가 작다고 할 수는 없다고 판단하였다. 또한 정확도도 0.02정도로 매우 작게 나왔다.

추가로, epoch 수를 바꾸거나 LSTM 레이어를 추가하는 등의 조치도 취해봤지만 그나마 위의 모델에서 가장 좋은 결과를 보여주었다.

### 피드백 반영

스크린샷이(가) 표시된 사진

자동 생성된 설명

Sigmoid 함수를 사용하기 위해 데이터를 (초 단위 기상시간)/(86400=하루 총 시간)으로 변환해 0~1사이의 값으로 스케일링을 진행했다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

오히려 정확도가 낮아진 것을 볼 수 있다.

# 프로젝트에 활용한 오픈 소스

Python3

psutil 5.7.0

pyautogui 0.9.50

numpy 1.18.1

tensorflow 2.2.0

pandas 1.0.1

matplotlib 3.1.3

keras 2.3.1

# 시행 착오

## 주제 선정 과정에서의 시행 착오

처음에는 Fuzzer를 구성해 볼 계획을 가졌으나, 프로젝트 기간 대비 난이도가 너무 높아서 포기했다.

IoT 시스템 후킹을 생각해서 진행하다가 프로젝트의 의미가 없을 것 같다는 조교님들 피드백을 바탕으로 IoT 기기별 데이터 수집 모델로 변경하여 프로젝트를 진행하게 되었다.

데이터를 크롤링해 와서 중앙 서버로 전송하는 것까지를 목표로 했는데, 수집한 데이터를 활용해 보는게 어떻냐는 피드백을 바탕으로 LSTM 알고리즘 적용까지로 목표를 확대하게 되었다.

## 구현 과정에서의 시행 착오

### 환경 개발

서브프로세스를 실행할 시 포크가 일어나지 않고 일방적으로 하위 프로세스를 실행하는 방식으로 실행되어, 부모 프로세스가 죽는 현상이 발생했다. 이는 subprocess.run 함수의 문제였으며, 이를 subprocess.Popen 함수를 이용하여 정상적인 포크를 통해 기존의 부모 프로세스도 잘 실행될 수 있도록 수정할 수 있었다

### 크롤러 구현 과정

크롤링하는 방법을 찾는 과정에서 처음에는 PID와 포크를 통해 부모 프로세스와 자식 프로세스 탐색을 기반으로 탐색을 진행해보려고 했지만, PID의 경우 프로세스가 재실행되면 초기화되어 랜덤한 값으로 다시 생성된다는 문제가 있었고, 그래서 파이썬의 클래스를 통해 일련의 딕셔너리를 생성하여 플래그 값을 기반으로 프로세스가 실행되었는지 종료되었는지 확인할 수 있는 모델을 개발했다. 그 과정에서 PID로 프로세스를 탐색할 수 없었기 때문에, 모든 PID를 갖고 있는 프로세스 중 프로세스의 이름을 기반으로 탐색하여 이를 딕셔너리에 넣어 활용했다.

### 크롤러의 형태

딥러닝에 넣을 데이터를 생각하면서, 크롤러의 형태를 이벤트 발생 시에 기록하는 형태와, 선형으로 1초에 한 번씩 데이터를 기록하는 두가지 형태로 나누어 구현했다.

### 연속적인 데이터 수집

1초에 한번씩 크롤러를 실행하고자 하였다. 이때 하나의 프로세스를 실행하는데 시간이 1초 이상 걸리기 때문에, 이를 해결하기 위해 스레드를 이용하고자 하여 처음에는 Threading 모듈을 이용하여 진행했다. 그런데 threading 모듈의 경우 Low level 스레딩이 아닌 타임쉐어링을 이용하는 High level의 스레딩을 사용하기 때문에, 프로세스를 무조건 1초마다 실행할 수 없다는 문제가 발생했다. 그래서 Multiprocessing 모듈로 변경하려고 했으나, 이때 데이터를 기록하는 CSV 파일이 이중으로 open되어 충돌로 인해 권한 문제가 발생했다. 그래서 하나의 프로그램을 타임쉐어링을 통해 사용하는 threading 모듈로 다시 변경했다. 이후 타임쉐어링 이슈로 인해 중간에 1초씩 빈 공간이 발생하는 오류 문제는 이후 데이터 전처리를 통해 해결하기로 결정했다

## 딥러닝 예측 과정에서의 시행 착오

프로젝트 결과 내용에 포함되어 있음.

# 향후 발전 방향

* 현재는 LSTM 알고리즘과 개발된 크롤러에 한정하여 의미 있는 결론을 내는 것에 집중했었다. 하지만, 향후에는 해당 알고리즘 외의 다른 알고리즘을 이용하여 사용자에게 더 유의미한 데이터를 제공할 수 있도록 할 예정이다.
* 크롤러를 추상화된 모델이 아니라 실제 모바일 기기와 홈 IoT 기기 데이터를 수집할 수 있도록 개발해 볼 예정이다.
* 클라이언트 단에서는 크롤러와 서버로 파일을 전송하는 클라이언트 소켓 프로그램을 하나의 프로그램으로 합쳐서 자동으로 하루 한번씩 로그 파일들을 옮길 수 있도록 제작할 예정이고, 서버 단에서는 여러 로그 파일들을 받아오는 서버 소켓 프로그램과, 로그 파일들을 하나로 합쳐 주는 프로그램을 하나의 프로그램으로 합쳐서 개발하거나, 작업 스케줄러를 이용하여 일일 실행될 수 있도록 개발해 볼 예정이다
* 사용자 데이터로부터 찾아낸 패턴을 기반으로 사용자에게 행동 패턴을 추천하는 프로그램을 개발할 예정이다.(어플리케이션 등)
* 예측 모델을 토대로 이를 홈 IoT 기기가 아닌 스마트 카 내부 기기 등 유사한 분야에도 적용 가능하고, 시간에 따른 위치 예측 등 데이터의 형태를 변환하여 다른 분야에도 적용 가능하다.
* 스마트 홈 사용 기기들의 데이터가 수집 되어 수집된 데이터를 바탕으로 일정한 루틴이 만들어질 경우, 그 루틴을 벗어난 행위들에 대해서는 경고 알람을 발생시킨다거나, 스마트 홈 도어락을 이중 잠금처리하는 등의 보안을 적용할 수 있을 것이다. 또 기기들의 데이터를 계속 크롤링해 와 기기의 로그가 서버에 남는다면, 그 로그를 바탕으로 공격 시도가 없었는지도 점검할 수 있다.

# 느낀점

## 김진욱

* 처음으로 하나의 완성된 프로그램을 구현하는 프로젝트를 진행했는데, 이로 인해 분업이 어떤 방식으로 진행되어야 하는지에 대해 알 수 있었고 나에게 특화된 강점을 키우는 편이 좋을 것 같다는 생각을 했다.
* 또한 이번 프로젝트를 진행하면서 여러 개념들을 실질적으로 코딩으로 풀어내는 과정을 경험할 수 있었는데, 그중 특히 재밌었던 점은 부모 프로세스와 자식 프로세스의 구조와 서버와 클라이언트의 소켓 통신이었다. 개인적으로는 파이썬보다 C언어가 더 손에 잘 맞는 편인데, 나중에 C언어를 이용해서 포크 프로세스와 소켓 통신에 대해서도 프로그래밍 프로젝트를 진행해 보고 싶다.
* 특히 가장 많이 공부한 내용은 CSV 파일을 파이썬에서 다루는 방식인데, 이는 개인적으로 가장 큰 수확이었던 것 같다. 프로젝트에서 파이썬을 처음 사용해 봤는데, 데이터 Handling이 매우 간편했던 점이 눈에 띄었다. 현재 빅 데이터 수집과 처리 프로세스에도 관심을 갖고 공부해보는 중인데, 이때 가장 중요한 DATA Handling에 대해 공부할 수 있어 차후 관련 프로젝트를 진행할 때 유용하게 사용할 수 있을 것 같다.
* 나머지 두 팀원들보다 고학번이었기 때문에 팀장으로서 주도해서 프로젝트를 진행했다. 중간중간 어려운 점들과 시행착오도 많았지만 조원들이 바쁘고 어려운 상황에서도 잘 따라와 줘서 프로젝트를 최종 단계까지 이끌어 낼 수 있었던 것 같다. 프로젝트 마무리 단계까지 잘 따라와 줘서 고맙고 필요한 일이 있을 때마다 다들 충실히 잘 수행해 줘서 고맙다. 부족한 조장 만나서 고생한 것 같아 미안하고 고마움을 많이 갖고 가는 프로젝트인 것 같다.

## 양인선

* 두 팀원들에 비해 프로젝트를 진행하는데 있어서 ‘코딩’에 기여한 바가 많이 적은 편이다. 또, 대학에 와서 처음부터 끝까지 프로그램을 구현해보는 팀 프로젝트 자체는 처음이었다. 적극적으로 “그건 제가 한번 해볼게요”라고 말하지 못한 것 같다. 크롤러 만드는 것도 수업을 통해 처음 배웠고, 다른 팀원이 구성한 서버-클라이언트 파일 전송 프로그램도 경험이 없었다. 그럼에도 불구하고 한번 해보겠다고 말할 수 있었을텐데 용기가 많이 부족했다.
* 대학 팀 프로젝트를 하면서 여태까지는 항상 주도적으로, 제일 많이 담당해서 했던 것 같은데 이번 팀플은 개인적으로 내 능력 부족으로 팀플에 도움이 안 되는 것 같아 스트레스를 많이 받았다. 만나서 팀플 하는 시간 동안 내가 여기서 가만히는 있지 말아야지 라는 생각으로 할 수 있는 일을 찾아서 하려고 했다.
* 모르는 것은 검색해 가면서 느리게 코드를 한 줄 씩 짰는데, 결국 직접 짜보고 공부해야 실력이 느는 것 같다. 눈으로만 보지 말고 직접 실행시켜 보고 오류를 해결해야 한다. 매크로 짜면서 발생했던 오류들에 대해서 검색해보고 선배들에게 물어보고 하면서 해결했는데, 그 때 마주쳤던 오류가 또 발생한다면 앞으로는 시행착오를 더 줄여서 해결할 수 있을 것 같다. 관리자 권한이 문제였던 오류인데, 지나고 보면 간단한 오류지만 겪지 않았다면 해결 방법을 몰랐을 것이다. 결국은 이렇게 배워 나가야 하는 것 같다

## 정영주

* 사용하는 알고리즘에 따라 수집하는 데이터의 종류가 정해지는데, 데이터를 수집한 후에 알고리즘을 선택해야 하는 상황이었기 때문에 어려움이 컸던 것 같다. LSTM 모델을 공부하고 구현해볼 수 있는 좋은 기회였지만 잘 알지 못하는 알고리즘이었기 때문에 좀 더 나은 결과를 내지 못한 것에 대한 아쉬움이 있다. 향후 알고리즘을 먼저 선택하고, 해당 알고리즘에 적합한 데이터를 수집할 수 있도록 크롤러를 변형해보는 것도 좋을 것 같다