종합설계 결과 보고서

**AI 미래가격 예측**

1조

박영민 32141847

유주열 32142942

이명재 32143180

목 차

1. **프로젝트 개요 -------------------- 3**
2. 프로젝트 목표
3. 프로젝트 개발 한계
4. **과제 수행 내용 ------------------- 3**
5. 개발 환경 & 기술 스택
6. 설계도
7. Data Augmentation
8. 텐서플로우
9. 상관계수
10. LSTM
11. 웹 연동 (Flask, AWS)
12. **결과 ------------------------------ ?**
13. **후기 ------------------------------ ?**
14. **출처 ------------------------------ ?**
15. **프로젝트 개요**
16. 프로젝트 목표

농산물의 미래 가격을 예측하고, 한달이내의 미래 가격을 예측함으로써, 머신러닝과 딥러닝의 활용방안을 공부, 이해하고 나아가 미래 가격을 필요로하는 client들 에게 도움을 주고자 제작하게 되었다.

1. 프로젝트 개발 한계

인공지능을 이용한 예측의 문제점으로 근미래는 비교적 정확하게 예측이 가능하지만 먼 미래를 예측 할수록 정확도가 크게 낮아지며, 예측 가능한 변인외의 돌발적인 변인들이 존재하여 정확한 예측이라고 하기 힘들다.

1. **과제 수행 내용**
2. 개발 환경

**JupyterNotebook (python 3.6)**

**PyCharm IDE (python 3.6)**

**Flask Web Server**

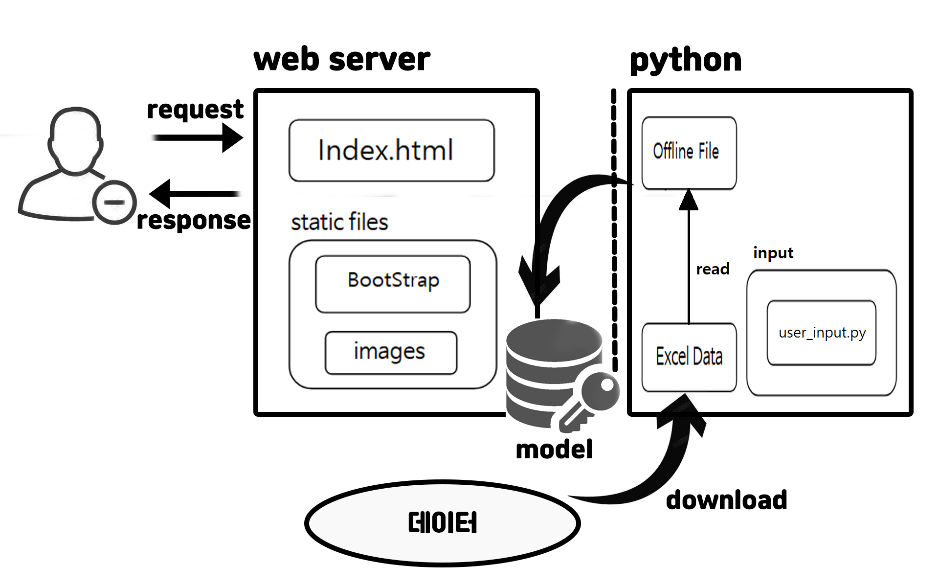
**AWS EC2**

1. 기술 스택

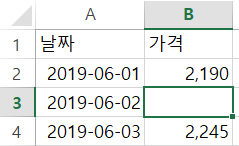
**Tensorflow**

**Keras**

1. 설계도



1. Data Augmentation



Data Augmentation에 대해 황두성 교수님께 질문을 하러 갔었을때, 교수님께서 여러가지 방법들을 가르쳐 주셨었다. 그 중 우리팀은 ‘중간값’을 찾아 결측치를 메꾸는 방법을 선택하였다. (우리 조는 결측치가 있던 곳을 아예 삭제하고 새로운 데이터를 받아오려고 했었는데, 교수님께서는 모든 데이터는 버리는 것이 아니라고 조언을 주셨었다.) 위의 표에서 보이듯이 결측치가 있는곳의 위의 가겨과 아래의 가격의 중간값을 결측치가 있는곳에 메꾸어 놓는 방법으로 Data Augmentation을 수행 하였다.

그 결과 약 20년간의 데이터(1999년~2019년)인 총 7496개의 데이터를 결측치 없이 얻을 수 있었다.

1. 텐서플로우

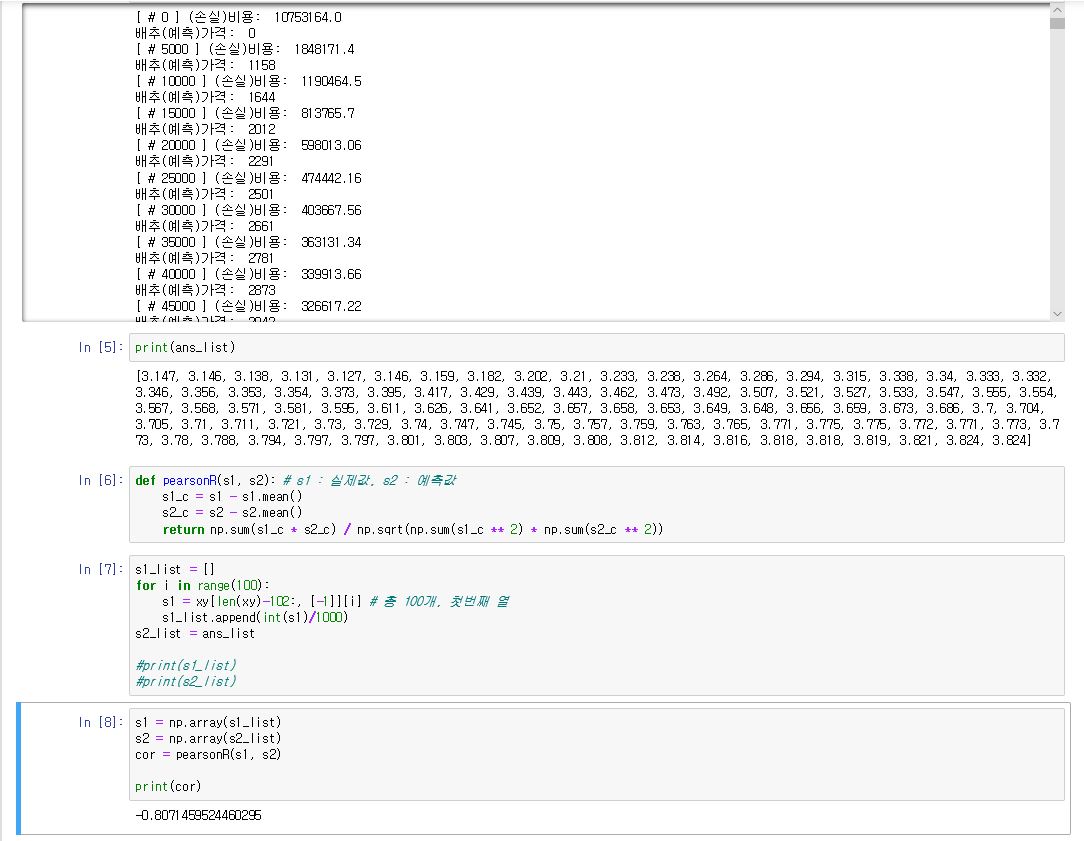
평균온도, 최저온도, 최고온도, 강수량 변인과, 앞서 피어슨 상관계수를 이용해 추출해낸 변인들(운량, 습도, 일조시간) 등을 변인list로 넣고, 경사하강 라이브러리를 이용해 가설을 설정한 뒤 (H = W\*x + b) learning rate와 학습횟수등을 적절히 지정하여 예측가격을 추출해 내는 작업을 하였다. 학습을 시켜줄 때, learning rate 와 학습횟수등은 직접 하나하나 값을 조정해가며 최적의 값을 찾아야만한다. 당연하게도 손실비용은 낮을수록 좋다.

Learning\_rate는 너무 클 경우 **오버슈팅**(cost가 NaN으로 나오는 현상) 이 발생하고, 너무 작을경우, 학습하는데 시간이 너무 많이걸리거나 **로컬미니멈**(cost가 거의 변하지 않는 현상) 이 일어나게 되므로, 여러 번 반복수행을 통해 적절한 값을 찾는 것이 중요하다.

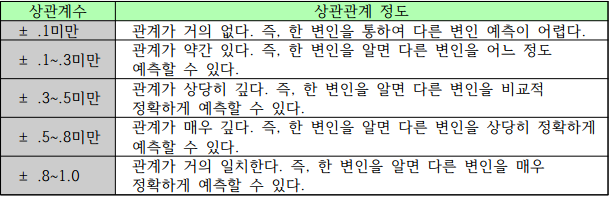
offline.py 코드를 실행한후, 예측결과를 saved.cpkt 의 파일로 저장하게된다. 이렇게 오프라인 model 파일로 저장하는 이유는 매번 온라인으로 몇십만번의 학습을 통해 학습결과를 도출하게 되면 시간과 비용이 많이 들기 때문에 offline으로 checkPoint의 형식으로 저장해두는 것이다. 이로인해 당연히 예측속도 역시 증가하게 될 수 있다.

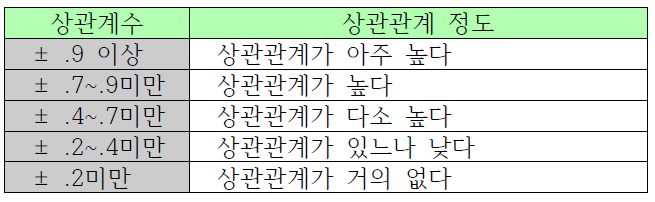
이제 웹에서 Client가 변인들을 입력하면 이 checkPoint파일에서의 결과를 가져와서 웹에 response 해주게 된다.

1. 상관계수



당근의 가격에 영향을 미치는 요인은 매우 많다. 최저온도, 최고온도, 평균온도, 습도, 강수량, 운량, 일조시간 등등..이 있다. 이러한 변수들을 피어슨 상관계수에 넣어서 어떤 변인이 가장 가격에 밀접한 영향을 미치는지를 알아보았다.





(논문에서 상관계수의 정도에 따른 결과값을 발췌해온 결과표)

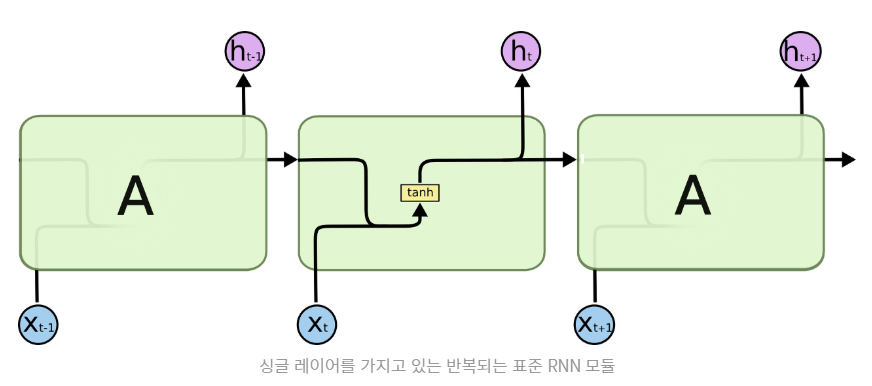
피어슨 상관계수로 변인들을 확인해본 결과, 습도가 0.9가 나왔고, 운량과 일조량은 각각 0.8정도가 나왔다. 이로인해 습도가 가격에 가장 큰 영향을 미치는 것을 알 수 있었다. 나머지 두개의 변인역시 0.8이라는 높은 수치가 나왔기 때문에 3가지 변인을 모두 실제사용할 수 있는 변인list에 넣을것이다. 모든 코드와 진행과정은 깃헙에 정리해 놓았다.

(<https://github.com/pym7857/Price_Change_Prediction>)

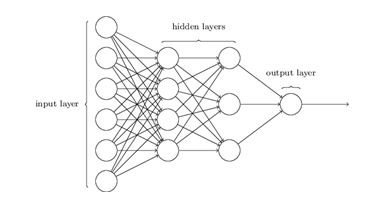
1. LSTM

LSTM의 구현 방식은 해당 데이터의 일정 비율을 이용하여(window size) 다음 데이터를 예측하고 이를 반복하여 학습데이터를 누적시켜 미래를 예측할 수 있게 한다. 예를 들어, 1-50을 이용해 51을 예측하고, 2-51을 이용해 52를 예측하는 것을 반복하여 미래의 값을 예측하게 하는 것이다. 여기서 1-50에 해당하는 값과 학습률을 조정하며 최대한 정확한 예측을 한다. 먼 미래의 값을 예측하기 위해서 실 데이터 값 이후 예측한 데이터 값을 측정한 데이터 시트에 붙여넣어 더 먼 미래의 값을 예측할 수 있도록 한다.

LSTM은 RNN의 일종으로 인공신경망을 이용한 딥러닝 학습방법이다. 학습방법을 설명하자면 우선 데이터를 여러 개의 시퀀스로 나누고 시퀀스별로 하나의 학습 결과를 도출해 낸다. 이때 시퀀스는 첫번째 시퀀스가 1~60번째의 데이터로 61을 예측하고 두번째 시퀀스가 2~61번째의 데이터로 62를 예측하는 방식으로 이루어진다. 여기까지는 기존의 RNN과 같지만 LSTM은 장기 의존성 문제를 해결했다. 이는 이전에 도출해냈던 결과를 기억함으로써 가능해졌다.



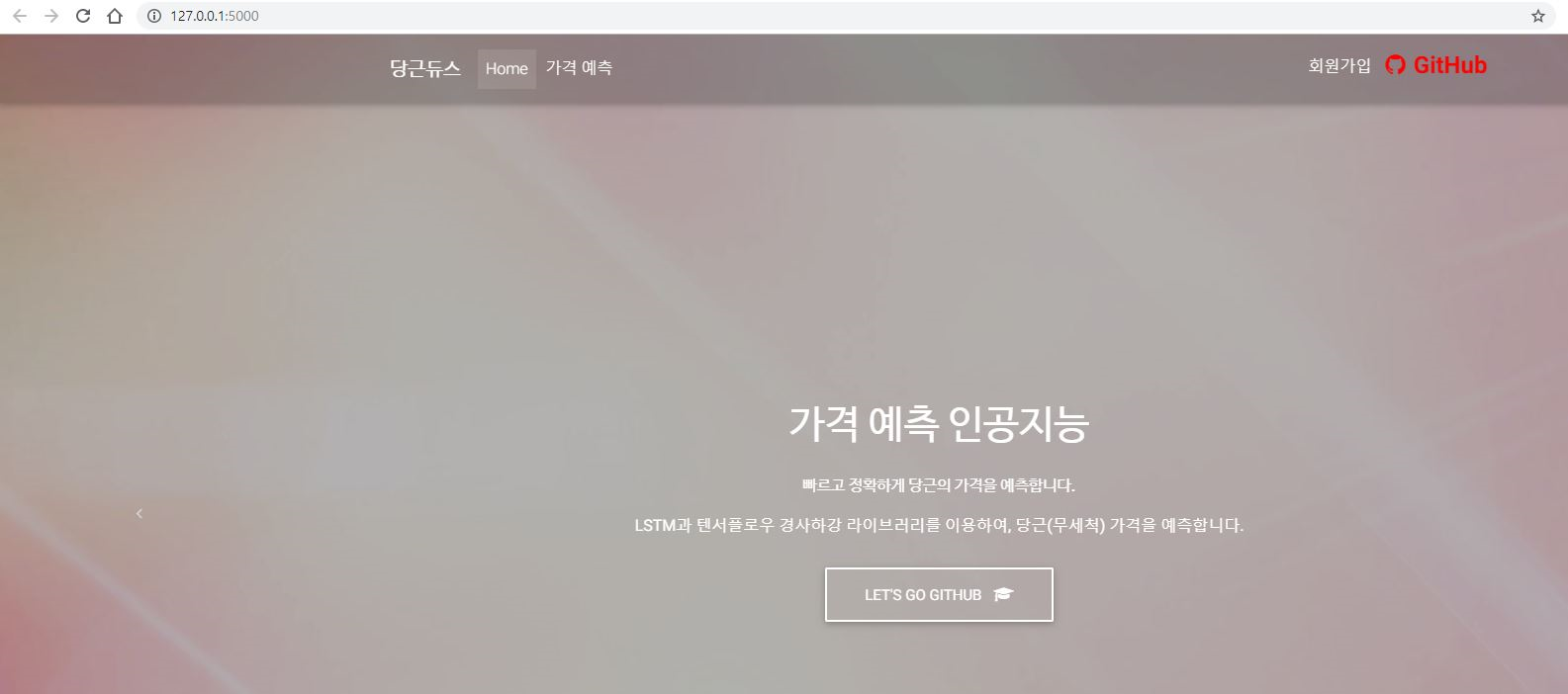
위의 사진 같은 경우에는 단일 레이어를 사용한 경우이다. 이때 레이어에서 나오는 결과의 수는 따로 지정할수 있는데 우리 프로젝트에서는 하나의 결과를 다음레이어로 보내는 방법을 사용하였다. 다음레이어로 보내는 값을 정하는 것을 활성함수를 통해서 결정이 가능한데 기존에 사용하던 방식은 linear function을 사용해서 다음레이어로 값을 넘겼다. 이는 레이어의 수가 2개여서 가능했는데 학습의 정확도를 올리기 위해서 레이어를 추가하였더니 단일레이어에만 어울리는 linear function같은 경우는 사용하기 힘들었고 다층 레이어에 어울리는 sigmoid, tanh, relu function중에 선택해야 했다. 이중 relu는 좀더 깊은 레이어에 어울리는 활성함수여서 제외하고 좀더 성능이 좋은 tanh 함수를 사용했다.



이런 방식을 사용해서 우리는 현재 존재하는 데이터로 미래의 1일의 가격을 예측할수 있었다. 하지만 좀더 미래의 가격을 예측하기 위해서는 추가적인 코드가 필요했다. 그래서 처음에 사용한 방법은 한번의 학습을 통해 4~5일의 미래가격을 한번에 예측하는 것이였는데 이 방법은 가격의 증,감 비율이 일정하게 변하는 문제가 존재했다. 그래서 한번 학습을 통해 1일의 미래 가격을 예측하고 그 결과를 다시 데이터셋에 넣고 다시 학습해서 결과를 얻어내는 방식을 사용했다. 이런 방법을 통해 얻어낸 결과는 보름 정도는 실제 가격의 동향을 잘 파악해서 따라가고 그 이후는 정확도가 점차 낮아지는 것을 확인 할수 있었다.

1. 웹 연동 (Flask, AWS)
2. Flask Web Server





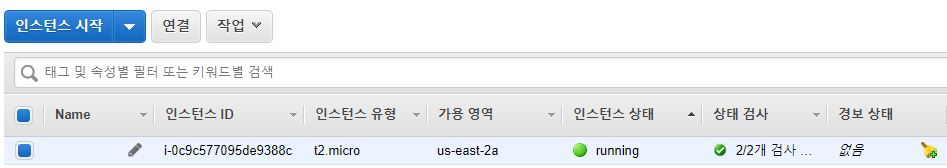
Flask를 이용해 python코드와 웹서버를 연동시킨다. Flask 서버와 연동이 되면, localhost:5000번 포트로 웹서버가 연동된다. (127.0.0.1 : localhost)

이를 호스팅하기 위해 AWS EC2를 이용하여 배포를 할 수 있다.

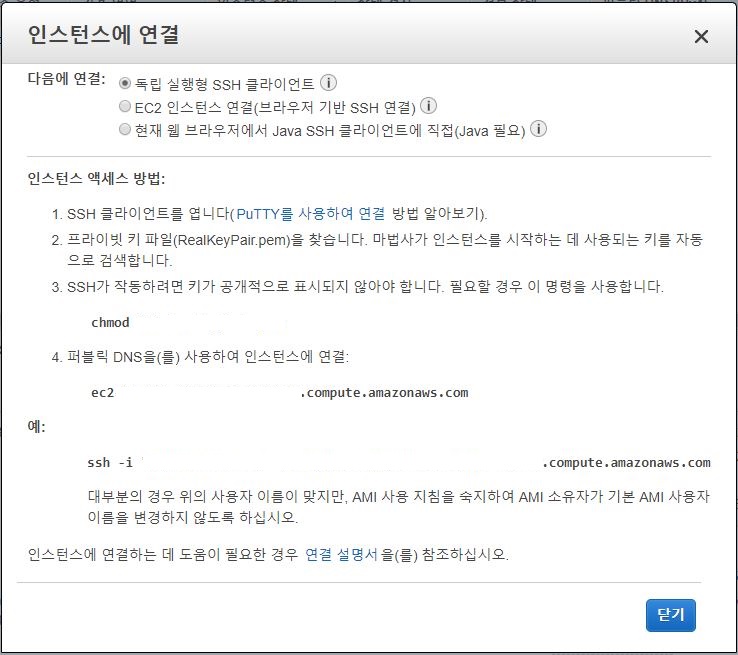
1. AWS EC2

AWS EC2는 아마존에서 제공하는 무료 클라우드 웹서비스로 가상 서버를 제공받아 웹서비스를 배포하도록 해준다.

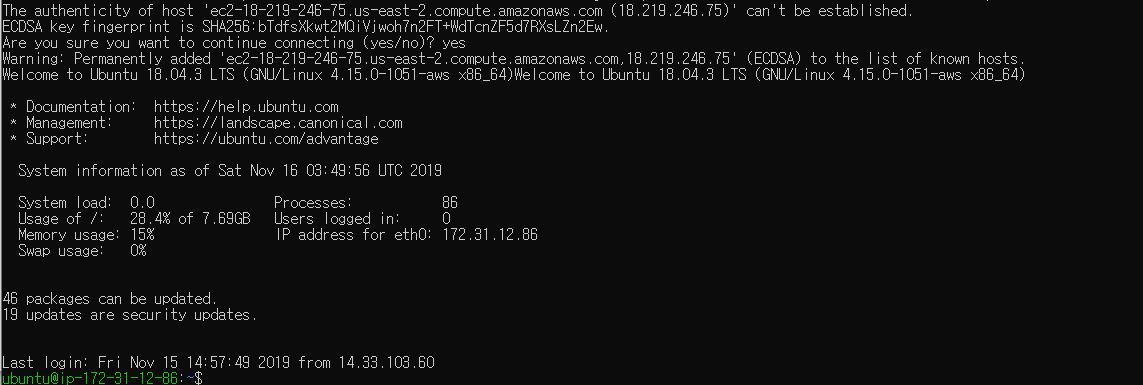
AWS EC2를 이용해 인스턴스(가상 컴퓨팅 환경)를 생성하면 한쌍의 키를 제공 받고 이 키를 이용하여 인스턴스를 관리 할 수 있다.



EC2에 접속하기 위해서는 SSH명령어를 사용한다.

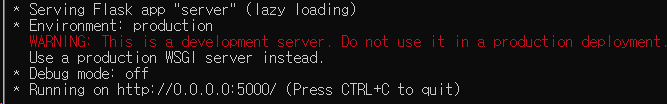


윈도우 환경에서 제공받은 키를 이용하기위해서는 속성을 설정해서 상속관계를 변경하고 접속할 수 있다.

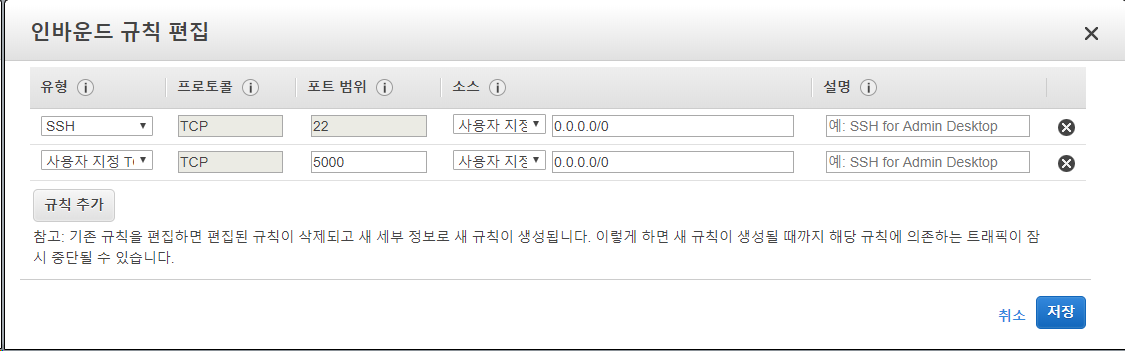


접속에 성공하면 깃 클론을 이용해 플라스크 코드를 불러온 후 관리자 권한으로 실행하면 서버를 구동 할 수 있다

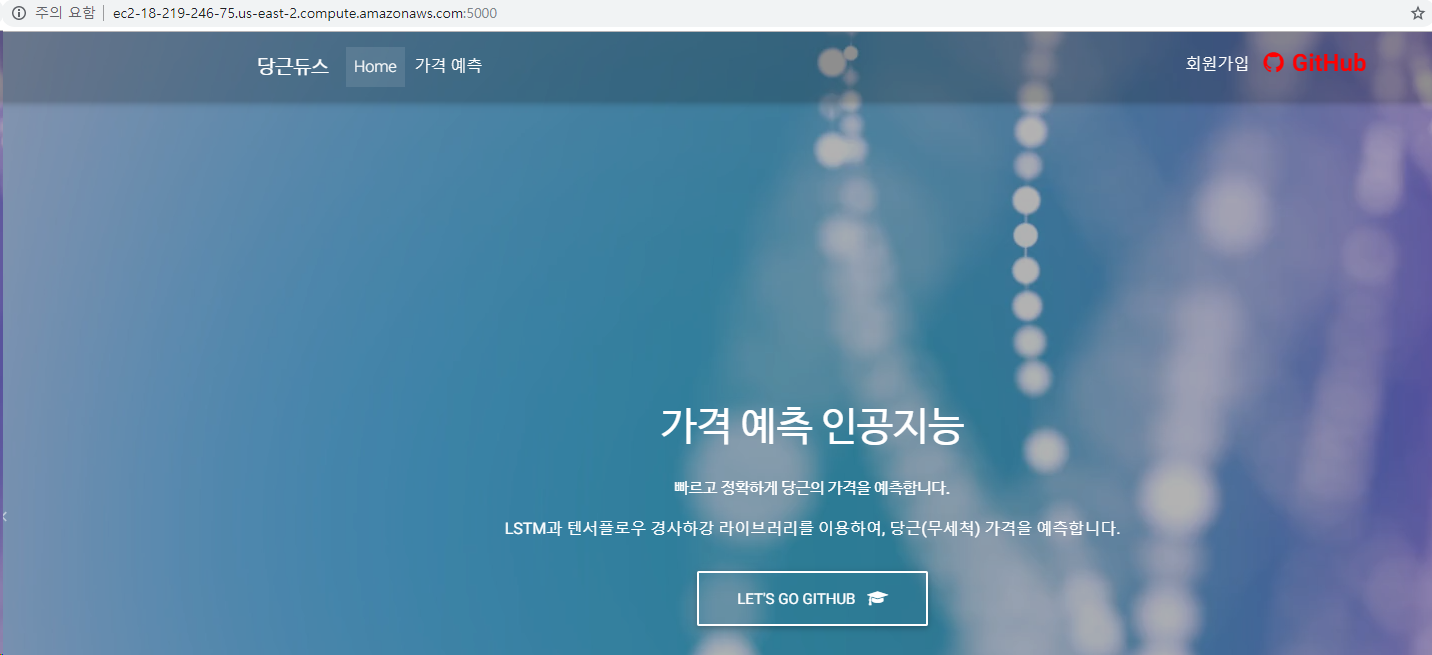




서버를 배포하기 위해 추가적으로 인스턴스의 보안설정을 해줘야 한다. 보안설정의 인바운드 규칙에 플라스크 웹 서버의 포트 번호인 5000번 포트를 열어주면 인스턴스의 public IP/DNS를 이용해 접속 할 수 있게 된다.

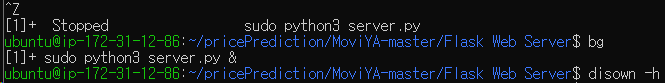




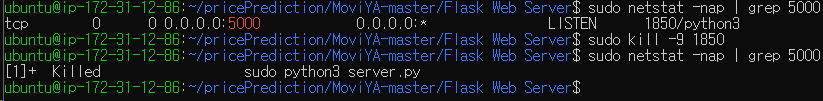


이 경우에 SSH프롬포트를 종료하면 파이썬 프로세스가 죽게되어 서버가 중지되게 된다. 따라서 파이썬 프로세스를 백그라운드에 동작하도록 한 뒤에 소유권을 포기하는 방식으로 프롬프트를 종료해도 서버가 유지되도록 하였다.

bg 명령어를 이용해 백그라운드에 서버를 구동시키고 disown -h를 이용해 소유권을 포기한다.



서버를 중지시키기 위해서는 5000번 포트에서 동작하는 프로세스를 확인 한 후 종료시키는 방법을 사용한다.



1. **결과**
2. 텐서플로우 결과화면
3. LSTM 결과화면
4. **후기**
5. 박영민

처음 프로젝트 주제를 탐색하고 선정하는데에 많은 노력이 들었었다. 초기단계에는 텐서플로우를 이용해서 미래가격을 예측해보기로 주제를 잡고 코딩을 시작하였었다. 그렇게 텐서플로우의 여러 라이브러리들을 이용해보면서 우리의 주제에 맞는 가격예측에 최적화된 라이브러리를 탐색하고, Flask를 이용해서 텐서플로우 코드와 웹서버를 연동했었는데, 결국 한계에 부딪혔었다. 텐서플로우만을 이용해서는 2주~한달 뒤쯤의 가격을 예측하는데에는 무리가 있었던 것이다. 이로인해 우리팀은 딥러닝주제인 LSTM을 도입하기로 했었다. LSTM을 이용하여 결과를 도출해낼때, 추세 그래프와 같은 시각화 툴들을 많이 이용했엇고 자료 데이터의 개수도 훨씬 많이 늘렸었다. 이때 도움을 받으러 황두성 교수님과 여러 TA님들을 만나면서 딥러닝의 원리, 방법과 데이터를 다루는 방법등을 많이 익힐 수 있었던 것 같다. 하지만 LSTM을 이용해 가격예측하는 프로젝트를 진행하면서 역시 수많은 어려움에 봉착할 수 밖에 없었다. 한달보다 더 먼 미래의 가격을 예측하는 것은 가능은 했지만 정확도가 매우 낮았다. 이로인해 우리팀은 한달까지만의 미래가격을 예측하기로 결정하였었다. 이번 프로젝트 경험을 통해 지금껏 공부할 수 없었던 머신러닝과 딥러닝 분야를 공부할 수 있어서 개인적으로 매우 좋았다. 다음학기 종합설계 프로젝트때는 이번에 배운 것을 토대로 좀더 큰 규모의 딥러닝 프로젝트를 해볼 생각이다.

1. 유주열

ㅇㅇ

1. 이명재

처음 시작할때는 텐서플로우를 이용하면 쉽게 가격 예측을 할 수 있을 것이라 생각해서 주제 선정에 고민하지 않았다. 제공되는 API가 사용하기 쉽게 되어있었고 활용하는 것 또한 어렵지 않았기에 텐서플로우를 사용해 근미래의 가격을 예측하는 것은 쉬웠다. 하지만 배포를 비롯한 완성도를 생각했을 때 더 먼 미래의 가격을 예측해야 함을 느꼈고 텐서플로우를 이용한 프로젝트의 한계를 느꼈다. 그렇게 LSTM을 이용하여 프로젝트를 다시 하기로 하였고, 결과가 그래프로 나오는점과 더 정확하고 먼 미래를 예측할 수 있다는 점에서 완성도가 올랐지만 LSTM이 가진 원리와 정확도를 높이는 법 등의 어려움이 많았고 장기적으로 하는 프로젝트가 아닌 비교적 단기적으로 하는 프로젝트이며 학부생인 팀원들의 한계로 어려움이 많았다. 그래도 인공지능을 이용한 첫 프로젝트였기에 흥미로운 주제였고 AWS 또한 처음 사용해보면서 좋은 경험이 되었다.

1. **출처**

(LSTM 이미지)

<https://brunch.co.kr/@chris-song/9>

(깃헙)

<https://github.com/pym7857/Price_Change_Prediction>