# Assignment#1
## 코스피 지수 최저점 예측

국민대학교
소프트웨어학부
20191631
윤민상

# 1. 주제 선정 동기 및 문제 설명

제가 선택한 주제는 LSTM을 사용하여 코스피 지수의 최저점을 예측하는 것입니다. 예전부터 비트코인과 같은 가상화폐, 주식에 관심이 있었고 실제로 많은 투자를 진행했습니다. 최근까지도 투자를 이어오고 있는 상황인데, 코스피시장에 있는 종목들 중 하나에서 큰 손해를 보고 있습니다. 매일 주가를 확인하며 이 종목을 계속 보유하고 있는 것이 맞을지, 아니면 지금이라도 매도하는 것이 맞을지 많은 고민을 합니다. LSTM을 통해 코스피 지수의 최저점을 예측해본다면 관심 분야와 겹쳐 과제를 잘 진행할 수 있을 것 같았고, 신뢰할 수 있는 결과값이 나온다면 제 고민을 해결할 수 있을 것이라는 생각이 들어 주제로 선정했습니다. 실제 과제 제출날 직전의 코스피 지수를 확인할 수 있는 6/2일까지의 코스피 지수 데이터로 학습을 완료한 모델을 통해 다음 날 코스피 지수의 최저점을 예측해보았을 때, 현재의 값보다 높다면 종목을 계속 보유하고 있는 것이 좋다는 판단을 내릴 수 있을 것입니다. 반대로 현재의 값보다 다음 날의 지수가 낮게 나왔을 때는 종목을 지금이라도 매도하는것이 맞다는 판단을 내릴 수 있을 것입니다.

# 2. Code의 작동 원리 및 전체 구조

제가 제출한 Code의 전체 구조는 data의 전처리를 위한 코드인 preprocessing.py, 모델의 학습을 위한 코드인 train.py, 학습에 사용할 data directory가 있습니다. data directory의 내부에는 데이터의 원본값이 담긴 raw.csv 파일과 preprocessing.py를 통해 데이터 전처리를 진행해준 minmax.csv 파일이 있습니다. csv data 파일들에는 그 날의 시장이 열렸을 때와 닫혔을 때의 코스피 지수, 그 날 코스피 지수의 최대값과 최소값, 그 날의 거래량, 변화율 값이 담겨 있습니다. 그리고 사양 문제로 학습이 돌아가지 않을 것

을 대비하여 미리 전처리해준 data 파일과 함께 Colab 환경에서 학습을 진행하기 위한 ipynb 파일이 있습니다.
 우선 preprocessing.py 파일에서는 min-max 방식을 사용하여 데이터 전처리를 진행하여 줍니다. min-max 방식은 feature 값의 범위를 0~1로 조정하는 방법으로, 아래 첨부 사진과 같은 수식으로 구현됩니다.

$$x_{scaled} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

data directory에 있는 raw.csv 파일은 코스피 지수가 처음 시작된 1981년 05월 01일부터 현재까지의 데이터가 들어있습니다. raw.csv를 min-max 방식을 통해 전처리된 데이터는 0과 1사이의 값만으로 구성되어 minmax.csv 파일로 저장됩니다. 이 파일은 이후 train.py 에서 학습을 진행할때 실제로 사용할 데이터입니다.
 train.py 에서는 minmax.csv 데이터와 LSTM 방식을 적용하여 실제 학습을 진행합니다. 30일동안의 데이터를 입력으로 사용하여 31일째의 코스피 지수 저점의 예상값을 출력으로 알려주는 구조입니다. 모델에 학습이 진행될때 매 epoch 마다 loss값이 출력되어 학습이 잘 되어가는지 확인할 수 있도록 하였고, 최종적으로는 오늘의 최저 코스피 지수와 내일의 예측 최저 코스피 지수를 출력해주도록 하였습니다. min-max 방식으로 데이터를 전처리해주었기 때문에 코스피 지수가 0~1 사이의 값으로 출력됩니다. 이를 좀 더 편하게 확인하기 위해 오늘과 내일의 최저 코스피 지수를 float 형으로 동시에 출력하도록 하였고, 사용자는 이 두개의 값을 비교하여 자신의 투자를 유지할지 매도할지에 대해 판단할 때 도움을 받을 수 있습니다.

# 3. 결과 분석

Colab 환경에서 epoch을 500으로 설정한 후 코드를 실행한 결과, 처음에는 약 0.0653로 시작했던 loss가 학습을 진행하며 마지막엔 아래 사진과 같이 약 0.0001까지 떨어진 걸 확인하였습니다. 하지만, 오늘의 최저 코스피 지수와 다음날의 예상 최저 코스피 지수의 차이가 너무 크게 나왔습니다. 따라서 overfitting은 발생하지 않았지만 아직은 학습이 충분히 진행되지 않았다 생각하여 epoch를 5000으로 대폭 상승시켜 보았습니다. 그 결과 오히려 overfitting이 발생하여 loss값이 비정상적으로 출력되는 현상이 발생하였습니다. 그 후 적절한 GPU 사용량을 유지하면서 overfitting이 발생하지 않고, 원하는 정도의 결과를 얻을 수 있는 epoch을 찾기 위해 epoch 값을 바꿔가며 실험을 진행하였습니다. 최종적으로 epoch을 1500으로 설정하여 학습을 진행하였을때 가장 좋은 결과를 얻을 수 있었습니다. epoch이 1500일 때는 overfitting이 발생하지 않았으며 아래 사진과 같이 loss도 약 0.2616에서 시작하여 0.0001까지 떨어졌기 때문에 정상적인 학습이 진행되었다 판단하였습니다.

```
[  1  /  1500 ] loss :  0.261665940284729
[  2  /  1500 ] loss :  0.2555510997772217
[  3  /  1500 ] loss :  0.2494833916425705
[  4  /  1500 ] loss :  0.243445947766304 02
[  1497  /  1500 ] loss :  0.00010408003436168656
[  1498  /  1500 ] loss :  0.0001040502538671717
[  1499  /  1500 ] loss :  0.00010402048792457208
[  1500  /  1500 ] loss :  0.0001039907438098453
```

예측 최저 코스피 지수또한 유의미한 결과값이 출력되었습니다. 아래 사진과 같이 오늘의 최저값은 0.9897, 내일의 예측 최저값은 0.9677이 출력되었습니다.

```
오늘의 최저값 :  tensor([0.9897])  / 내일의 예측 최저값 :  tensor([[0.9677]]
```

결과값 같은 경우에는 원래 역변환을 하여 실제 코스피 지수의 값으로 돌려 출력할 생각이였으나, 계속 code에 오류가 발생하였습니다. 제 실력의 문제라고 생각되어 다른 방법을 생각하던 중, 차라리 오늘의 최저값과 내일의 예측 최저값을 역변환을 하지 않은 상태로 함께 출력해준다면 하루 사이에 어떻게 값이 변화하였는지 좀 더 확인하기 편할 것이라 생각하였습니다. 또한, 오늘의 최저값과 내일의 예측 최저값, 오늘의 실제 최저 코스피 지수를 알고있기 때문에 직접 비례식을 사용하면 내일의 예측 최저값을 구할 수 있을 것이라 생각하여 이런 형식으로 결과를 가시화 하였습니다. raw.csv 파일에 담겨있는 오늘의 실제 최저 코스피 지수는 3210.31이였습니다. 따라서

0.9897 : 0.9677 = 3210.31 : 예측 최저값

이라는 비례식을 유추할 수 있었습니다. 내항의 곱과 외항의 곱의 값이 같다는 원리를 사용한다면 예측 최저값은 약 3138.95가 나옵니다. 어제의 최저값보다 71.36 낮은 값으로, 현재 가지고 있는 종목을 매도하는 것이 맞다는 판단을 도출해낼 수 있습니다. 하지만 하루 사이에 코스피 지수의 최소값이 71.36이나 감소하는 상황이 사실상 없기 때문에, 학습은 잘 진행되었지만 믿을만한 결과를 도출해내진 못했다고 판단했습니다.

처음에 LSTM을 적용하여 무슨 문제를 해결할까에 대한 고민을 많이 하였습니다. 이왕 문제를 찾는다면 제가 실제로 겪고 있는 문제들 중에서 찾고 싶었고, 그렇게 선정하게 된 주제가 최저 코스피 지수 예측이였습니다. 1981년부터 코스피 지수가 시작되었기 때문에 오랜 기간에 걸친 충분한 양의 데이터를 얻을 수 있어 LSTM을 적용하기에도 적당한 주제라고 생각하였습니다.

하지만 개발을 진행하고 필요한 자료를 찾아보며 제가 선정한 주제가 좋은 주제는 아니였다는 생각 또한 들었습니다. 코스피 지수라는 값 자체가 단순히 정형화된 패턴이 있을 수도 있지만 COVID-19 혹은 전쟁 발생, 사회적 현상 같은 외부 요인이나 경제적 상황이 많이 반영되는 값이기 때문에 LSTM

을 사용한 모델 학습으로는 좋은 예측값을 얻기가 힘들 것 같습니다. 그래도 직접 겪고있는 문제를 주제로 선정하여 이에 대해 알아보고, 이를 해결하기 위해 모델을 학습시켜 결과를 도출해내는 과정 자체가 상당히 흥미롭고 재미있었습니다. 비록 좋은 결과값을 얻진 못한 것 같지만, 다음에는 좀 더 열심히 공부하여 다른 주제에 LSTM을 적용해 의미있는 결과를 얻어보고 싶습니다.